0.0.1 Question 1: 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 or didn't work?
3. What was surprising in your search for good features?

1) To find better features, it helped to find strings that were most common in spam emails and least common in ham emails. It was most helpful for me to read the multiple spam emails and understand the topics to narrow down similarly used words in them. Then I would be able to test it out in a model with the training set and see if it would increase the model accuracy.
2) One thing that worked for me in increasing accuracy was using searching/filtering for html tags. Using single words isn't effective in especially if they pertain to really niche topics. It's important to find words or phrases that encompass multiple spam topics and look for the words that were in spam emails. Many of the spam emails seemed to market something whether it'd be a product or explicit content.
3) One thing that surprised me in my search for good features was the commoness of links in spam emails that are likely phishing emails. There also seemed to have a lot of html tags in spam mail. One word that particularly helped me increase accuracy in looking at spam emails was 'death'.

Question 2a Generate your visualization in the cell below.

```
In [1002]: x = words_in_texts(['transfer'], spam['email'])
           len(x)
           sum(x)/len(x)
```

```
Out[1002]: array([0.22367049])
```

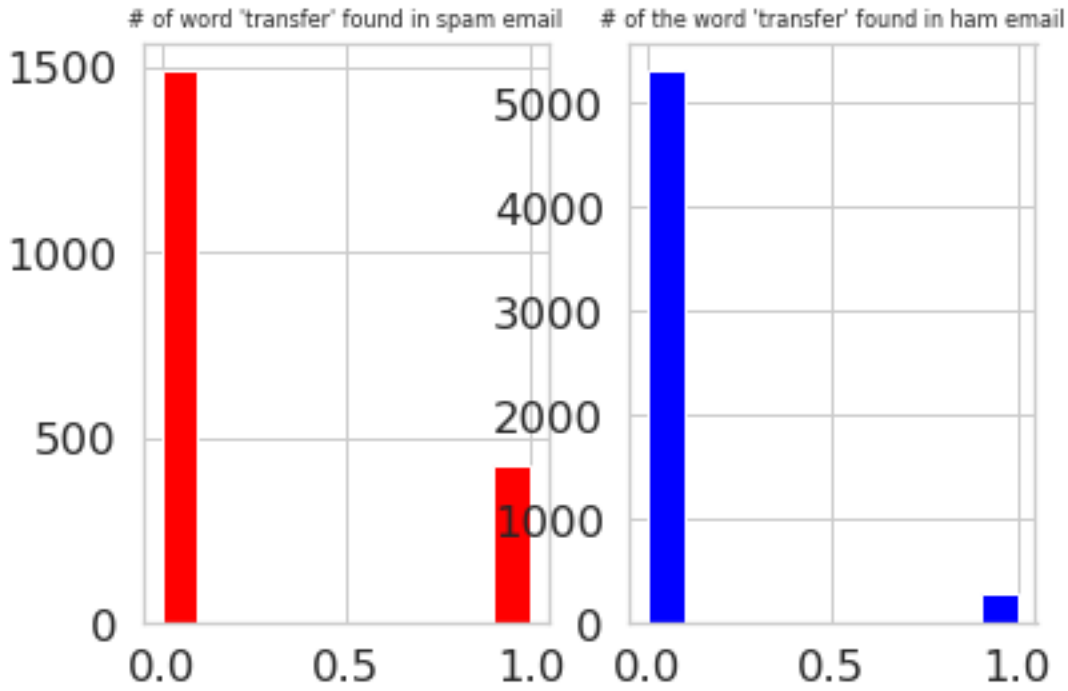
```
In [1000]: y = words_in_texts(['transfer'], ham['email'])
           len(y)
           sum(y)/len(y)
```

```
Out[1000]: array([0.05075961])
```

```
In [1003]: plt.subplot(1,2,1)
           plt.hist(x, color = 'red')
           plt.title('# of word \'transfer\' found in spam email', fontdict = {'fontsize' : 8})

           plt.subplot(1,2,2)
           plt.hist(y, color = 'blue')
           plt.title('# of the word \'transfer\' found in ham email', fontdict = {'fontsize' : 8})
```

```
Out[1003]: Text(0.5, 1.0, "# of the word 'transfer' found in ham email")
```



Question 2b Write your commentary in the cell below.

Looking at the two subplot, we can see that the word 'transfer' is seen much more commonly in spam emails than ham emails. The exact proportion of the ham emails with the word 'transfer' to ham emails without it is 5%. The exact proportion of the spam emails with the word 'transfer' to spam emails without it is 22%. This is a significant difference between spam and ham; almost 4 times the rate in spam emails. This makes 'transfer' a good word to look at and an effective feature to classify between ham and spam emails. As the use of 'transfer' increases in an email, it's more likely that it can be classified as spam.

One reason why transfer could be so prevalent in spam emails is because it is likely referenced in topics such as finance and banking. Transfering money, scams, and phishing emails usually asks the recipient to send some amount of money for a certain cause or blackmail them into doing so. However, we should proceed with cautious because legitimate banking emails may also use the word 'transfer'.

I kept the graphs the same size despite having different x-axis to compare the 'trasnfer' distribution relative to its ham/spam email group.

0.0.2 Question 3: ROC Curve

In most cases we won't be able to get 0 false positives and 0 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 that they have cancer until it's too late, whereas a patient can just receive another screening for a false positive.

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 20 to see how to plot an ROC curve.

Hint: You'll want to use the `.predict_proba` method for your classifier instead of `.predict` so you get probabilities instead of binary predictions.

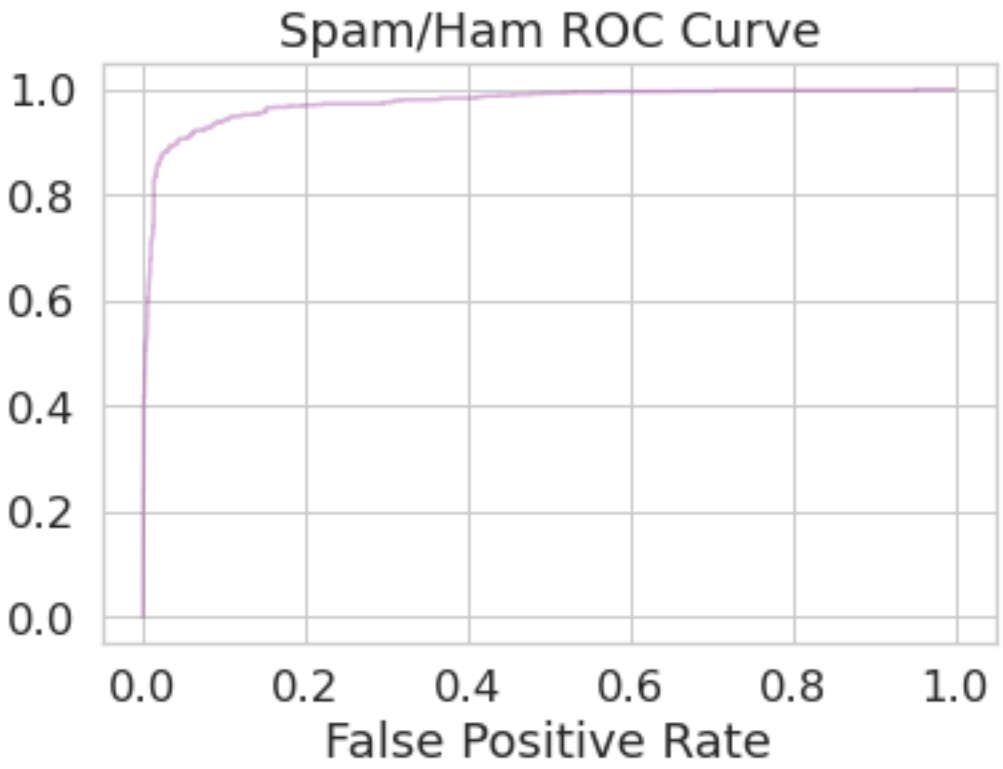
```
In [1023]: from sklearn.metrics import roc_curve

           prob = model.predict_proba(X_train)[:,1]
           false_pos, true_pos, threshold = roc_curve(Y_train, prob, pos_label=1)

           plt.step(false_pos, true_pos, color = 'purple', alpha = 0.3)
           plt.title('Spam/Ham ROC Curve')
           plt.xlabel('False Positive Rate')
           plt.Ylabel('True Positive Rate')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Input In [1023], in <cell line: 9>()
      7 plt.title('Spam/Ham ROC Curve')
      8 plt.xlabel('False Positive Rate')
----> 9 plt.Ylabel('True Positive Rate')

AttributeError: module 'matplotlib.pyplot' has no attribute 'Ylabel'
```


Spam/Ham ROC Curve
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
False Positive Rate