0.1 Question 1

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. How did you find better features for your model? At first, I planned to find the list of words that can distinguish spam from ham emails, so I used the training dataset to find the most common words in spam emails that do not appear in the list of most common words in ham emails. In addition, I wanted to see whether spam emails are longer than ham emails, so that I created a violin plots to visualize the relationship between emails' length and their classification as spam or ham emails.
2. What did you try that worked or didn't work? Before I came up with the idea of using the training dataset to find words as features, I just came up with random words that I personally thought it would come from spam emails. However, this strategy didn't work because it took a lot of time guessing and the words I came up may show up in both spam and ham emails, thus not good to use as featues
3. What was surprising in your search for good features? I think the most surpring is from the my list of words to use as features, most of them are not actually words that makes sense as they mostly a combination of letters and numbers. I'm not sure why it happends that way; perhaps I want to explore what the meaning of these items of the list (e.g: font size eddditing, embedded link, etc.). On the other hand, in the common word list of ham emails, most of the items are English words rather than random symbols.

0.2 Question 2a

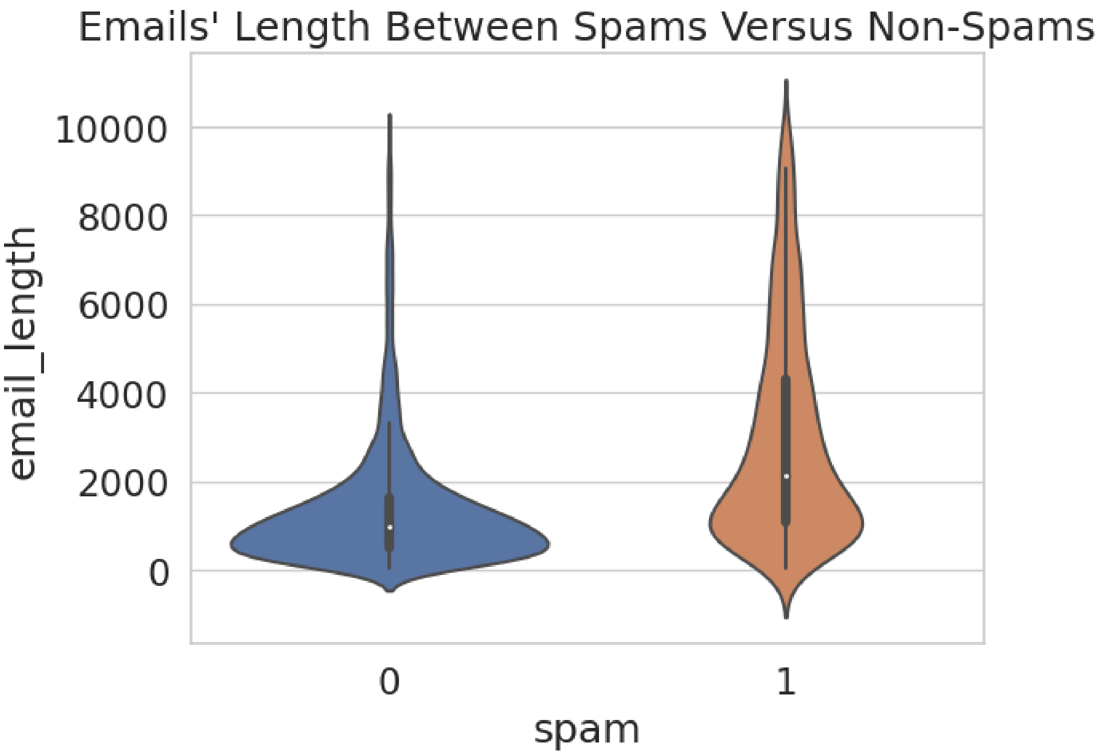
Generate your visualization in the cell below.

```
In [81]: # Visualize length of ham vs spam emails
         eda_1 = train.copy()
         eda_1['email_length'] = eda_1['email'].str.len()


         # sns.boxplot(data = eda_1, x = 'spam', y ='email_length');

         sns.violinplot(data = eda_1[eda_1['email_length'] <=10000], x = 'spam', y ='email_length')
         plt.title("Emails' Length Between Spams Versus Non-Spams"); #Remove outlier for visualization
         eda_1[eda_1['spam'] ==0]['email_length'].describe(),eda_1[eda_1['spam'] ==1]['email_length'].d
```

```
Out[81]: (count      5595.000000
          mean       2986.580518
          std        9104.817075
          min          54.000000
          25%         546.000000
          50%        1043.000000
          75%        1850.000000
          max      303302.000000
          Name: email_length, dtype: float64,
          count      1918.000000
          mean       5528.656413
          std       12532.161763
          min          36.000000
          25%        1140.250000
          50%        2692.000000
          75%        6043.500000
          max      234358.000000
          Name: email_length, dtype: float64)
```


Emails' Length Between Spams Versus Non-Spams
email_length
10000
8000
6000
4000
2000
0
0
1
spam

0.3 Question 2b

Write your commentary in the cell below.

The plot above visualizes how different emails' length vary between spam versus non-spam emails in our dataset. From the plot, we can see that both have normal distribution and large variance with big outliers go up to around 300,000. Those that are classified as spams have higher median and bigger variance while those classified as non-spams have lower median and smaller variance, meaning non-spam emails can have a clear typical length with some rare exception while spam emails can be longer and vary more. Overall, we can say that spam emails tend to be longer than non-spam emails.

0.4 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 threshold*: 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.

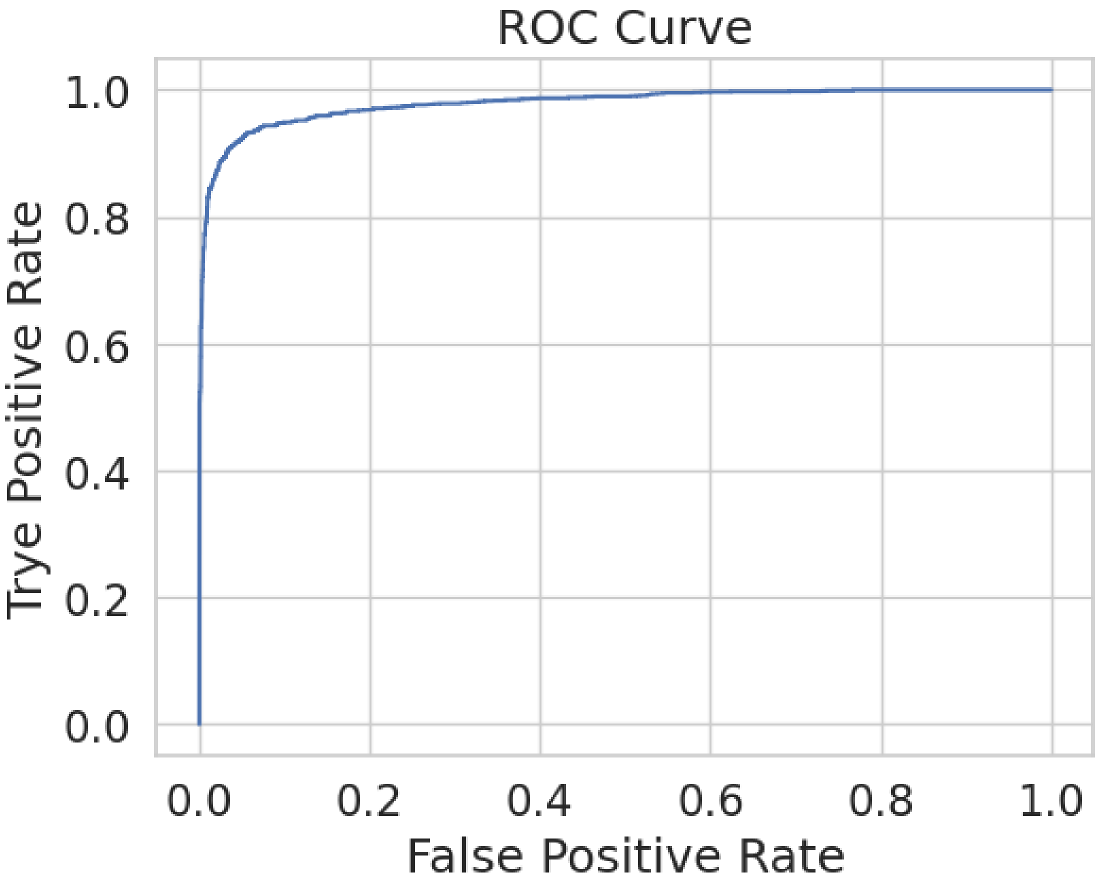
The Receiver Operating Characteristic (ROC) curve shows this trade off for each possible cutoff probability. In the cell below, plot a ROC curve for your final classifier (the one you use to make predictions for Gradescope) on the training data. Refer to Lecture 24 to see how to plot an ROC curve.

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

```
In [95]: from sklearn.metrics import roc_curve


         Y_train_prob = my_model.predict_proba(my_X_train)[:,1]
         FPR, TPR, _ = roc_curve(my_Y_train, Y_train_prob)

         plt.plot(FPR,TPR)
         plt.xlabel('False Positive Rate')
         plt.ylabel('Trye Positive Rate')
         plt.title('ROC Curve');
```


ROC Curve
Trye Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.2
0.4
0.6
0.8
1.0