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.I followed the second advise that provided on previous part, because it is most straitforward to me. In order to find better word as feature, I abstract the words appears mutiplie times on spam email and did't appear in ham email.

2.I first took the first advise that provided on previous part, but I have trouble to process 'E' in first advise, and it took so much time. After thinking about second advise it is easier to me, and I just used second one.

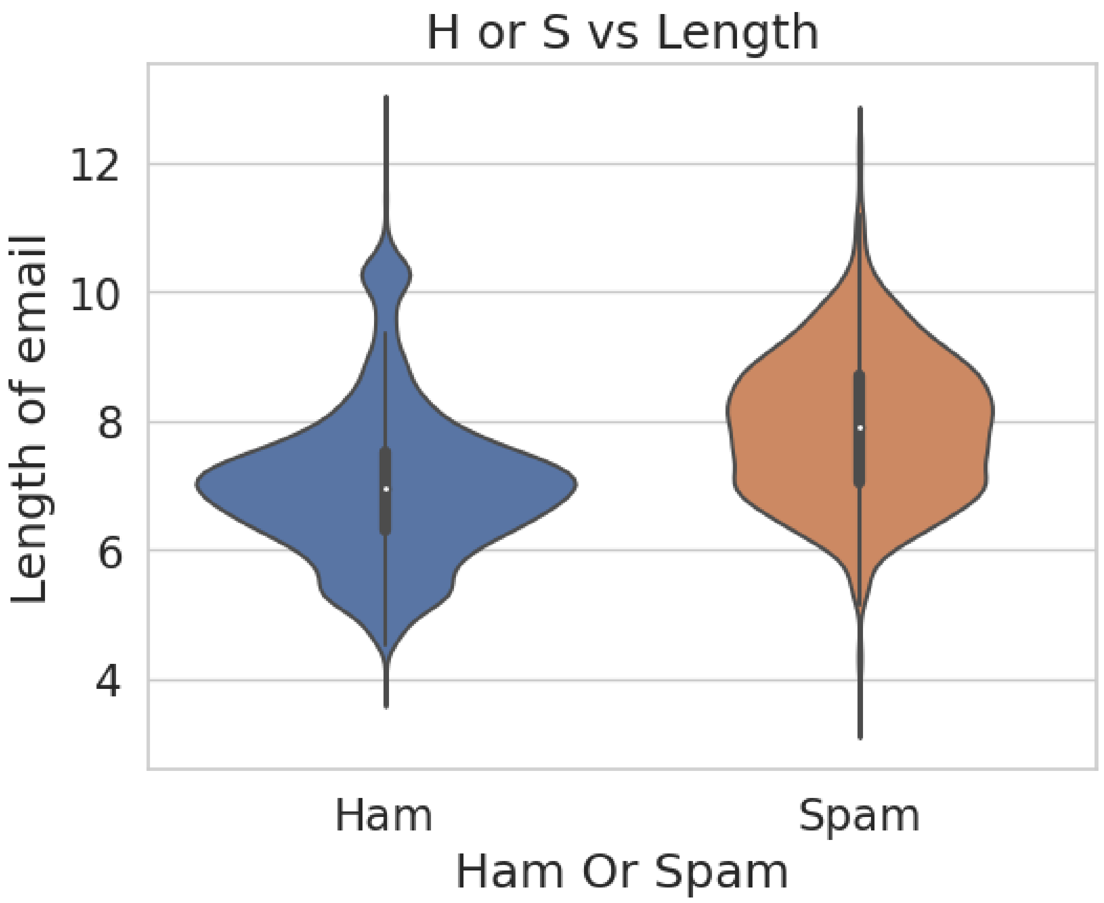
3.Some words appear only in spam are very rare to see like "0000cc" or "vjestika". I thought I may need more features because these rare words may not also appear in other spam email, but it truns out fine.

0.2 Question 2a

Generate your visualization in the cell below.

```
In [14]: train["len"]=np.log(train['email'].str.len())

         sns.violinplot(data=train, x="spam",y="len")
         plt.xticks([0,1],["Ham","Spam"])
         plt.title("H or S vs Length")
         plt.ylabel('Length of email')
         plt.xlabel('Ham Or Spam')
```

```
Out[14]: Text(0.5, 0, 'Ham Or Spam')
```



0.3 Question 2b

Write your commentary in the cell below.

It seems that the average length of spam email is longer than ham email. There are some outlyer for ham email that is long, but majarity of ham emails are shorter than spam email. May because there are long and not understandable words in spam as what we saw when I abstruct powerful features.

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.

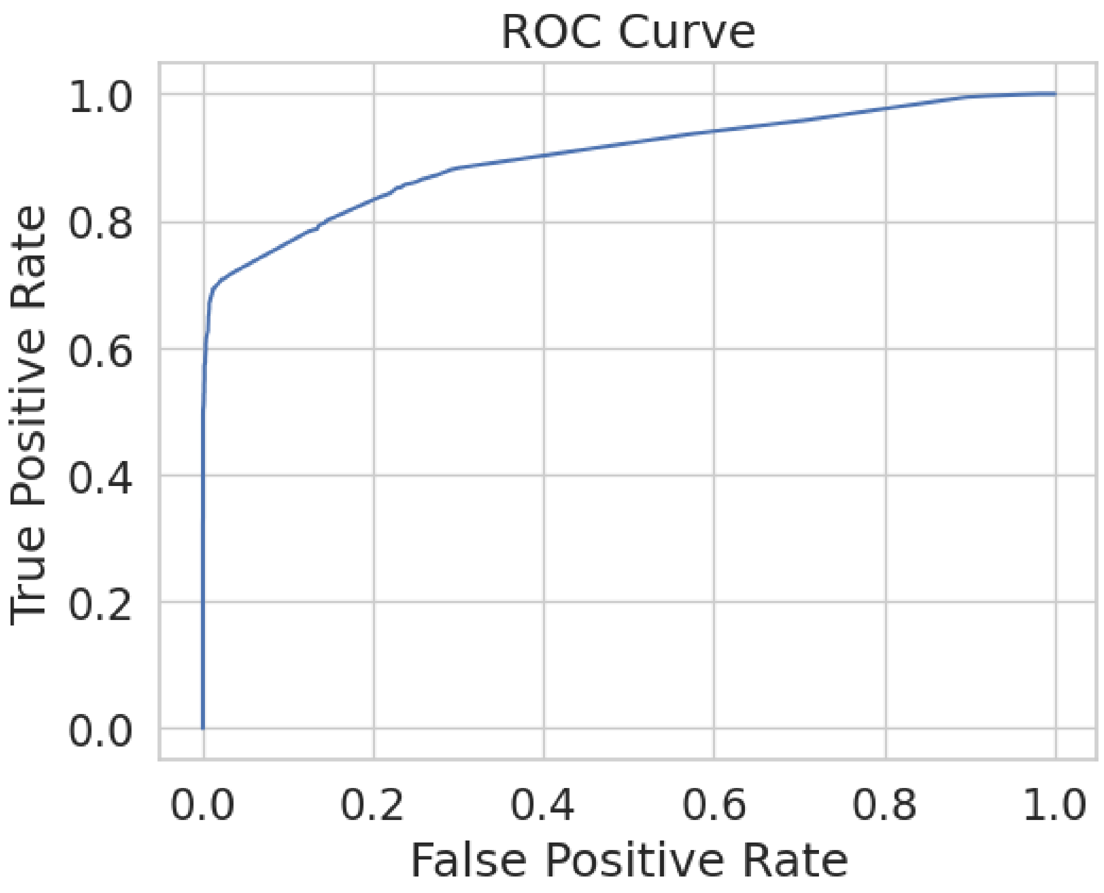
Recall that logistic regression calculates the probability that an example belongs to a certain class. Then, to classify an example we say that an email is spam if our classifier gives it ≥ 0.5 probability of being spam. However, *we can adjust that cutoff threshold*: we can say that an email is spam 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 [15]: from sklearn.metrics import roc_curve


         def predict_threshold(model, X, T):
             prob_one = model.predict_proba(X)[:, 1]
             return (prob_one >= T).astype(int)


         def tpr_threshold(X, Y, T): # This is recall
             Y_hat = predict_threshold(model, X, T)
             return np.sum((Y_hat == 1) & (Y == 1)) / np.sum(Y == 1)

         def fpr_threshold(X, Y, T):
             Y_hat = predict_threshold(model, X, T)
             return np.sum((Y_hat == 1) & (Y == 0)) / np.sum(Y == 0)

         thresholds = np.linspace(0, 1, 100)
         TPR = [tpr_threshold(X_train, Y_train, t) for t in thresholds]
         FPR = [fpr_threshold(X_train, Y_train, t) for t in thresholds]

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

```
Out[15]: Text(0, 0.5, 'True Positive Rate')
```


ROC Curve
True 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