CS 434 Report 1

Team number:8

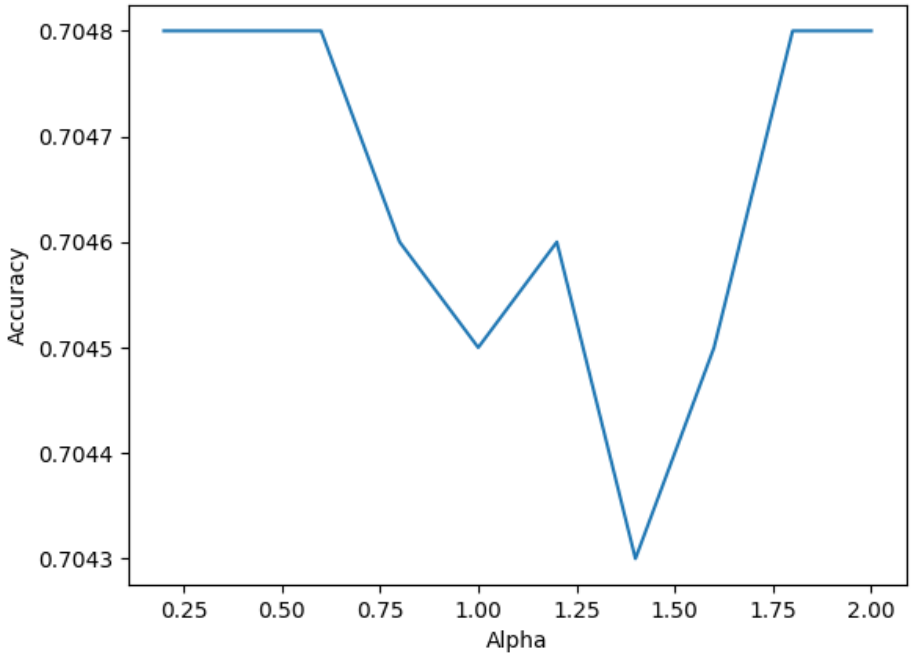
Team member: Adam Stewart, Hao Deng, Yuhang Chen

3. Apply the learned Naive Bayes model to the validation set (the next 10k reviews) and report the validation accuracy of the your model

```
Step3: validation accuracy is 0.7045
```

The validation accuracy of our model is 70.45%

4. Report the results by creating a plot with value of α on the x-axis and the validation accuracy on the y-axis. Comment on how the validation accuracy change as α changes and provide a short explanation for your observation



This graph is quite interesting, by alpha increasing from 0.25 to 2, the accuracy will drop first and then go back.

Multinomial:

$$P(x = w_i|y = 1) = \frac{\#\text{ of times word } i \text{ appeared in spam emails} + \alpha}{\text{total \# words in spam emails} + |V|\alpha}$$

Base on this formula, we can tell that alpha will change the probability of each words. And the change of probability will impact the mode and change the prediction eventually.

5. Please describe your strategy for choosing the value ranges and report the best parameters (as measured by the prediction accuracy on the validation set) and the resulting model's validation accuracy

```
The best combo of (Features,alpha) is:
(700, 0.4)
0.7413
```

When my max_features is 700 and alpha is 0.4, I got the best accuracy which is 74.13%

My strategy is finding the max threshold of max_features first, I tried a big range of numbers between 1,000 to 10,000

```
Step5:Tune max_features
max_features_test: 1000
max_features_test: 2000
max_features_test: 3000
max_features_test: 4000
max_features_test: 5000
max_features_test: 6000
max_features_test: 7000
max_features_test: 8000
max_features_test: 9000
[1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000]
[0.7311, 0.7048, 0.6826, 0.6908, 0.6791, 0.6718, 0.6868, 0.6855, 0.6769]
```

I found the higher max_features is, the worse it is. So I found the test range for max_features is between 100 to 1000.

I did the same thing for Alpha, and I found the range is (0,1)

In the end, I used parallel programming to tune both max_ features and alpha at the same time.

```
print("Step5: Tune both")

alphaTask=[0.2,0.4,0.6,0.8,1]
featuresTask=[i*100 for i in range(1,11)]
bothtasks=[(x,y) for x in featuresTask for y in alphaTask]


result=pool.starmap(tuneBoth,bothtasks)
print(bothtasks)
print(result)
```

Here is the result I got.

```
[(100, 0.2), (100, 0.4), (100, 0.6), (100, 0.8), (100, 1), (200, 0.2), (200, 0.
4), (200, 0.6), (200, 0.8), (200, 1), (300, 0.2), (300, 0.4), (300, 0.6), (300,
 0.8), (300, 1), (400, 0.2), (400, 0.4), (400, 0.6), (400, 0.8), (400, 1), (500
, 0.2), (500, 0.4), (500, 0.6), (500, 0.8), (500, 1), (600, 0.2), (600, 0.4), (
600, 0.6), (600, 0.8), (600, 1), (700, 0.2), (700, 0.4), (700, 0.6), (700, 0.8)
, (700, 1), (800, 0.2), (800, 0.4), (800, 0.6), (800, 0.8), (800, 1), (900, 0.2
), (900, 0.4), (900, 0.6), (900, 0.8), (900, 1), (1000, 0.2), (1000, 0.4), (100
0, 0.6), (1000, 0.8), (1000, 1)]
[0.6457, 0.6456, 0.6458, 0.6456, 0.6455, 0.6731, 0.673, 0.6728, 0.6726, 0.6723,
 0.7129, 0.7132, 0.713, 0.7129, 0.7128, 0.7313, 0.7315, 0.7312, 0.7311, 0.7313,
 0.7395, 0.7392, 0.7392, 0.7391, 0.7393, 0.717, 0.7167, 0.716, 0.7158, 0.7158,
0.7411, 0.7413, 0.7409, 0.7408, 0.7406, 0.7278, 0.7278, 0.7277, 0.728, 0.7276,
0.7192, 0.719, 0.7191, 0.7196, 0.7198, 0.7311, 0.7315, 0.7321, 0.7321, 0.7319]
The best combo of (Features,alpha) is:
(700, 0.4)
0.7413
```

So when max_features=700 and alpha=0.4 is the best combo for the this model. The accuracy is 74.13%