Q1) A vocabulary dictionary is created using both train and test data. I sorted and stored the dictionary in a file and than later sliced it to be the most frequent 5000 words.
The size of the dictionary was **150137** which means there are this many different words in documents.

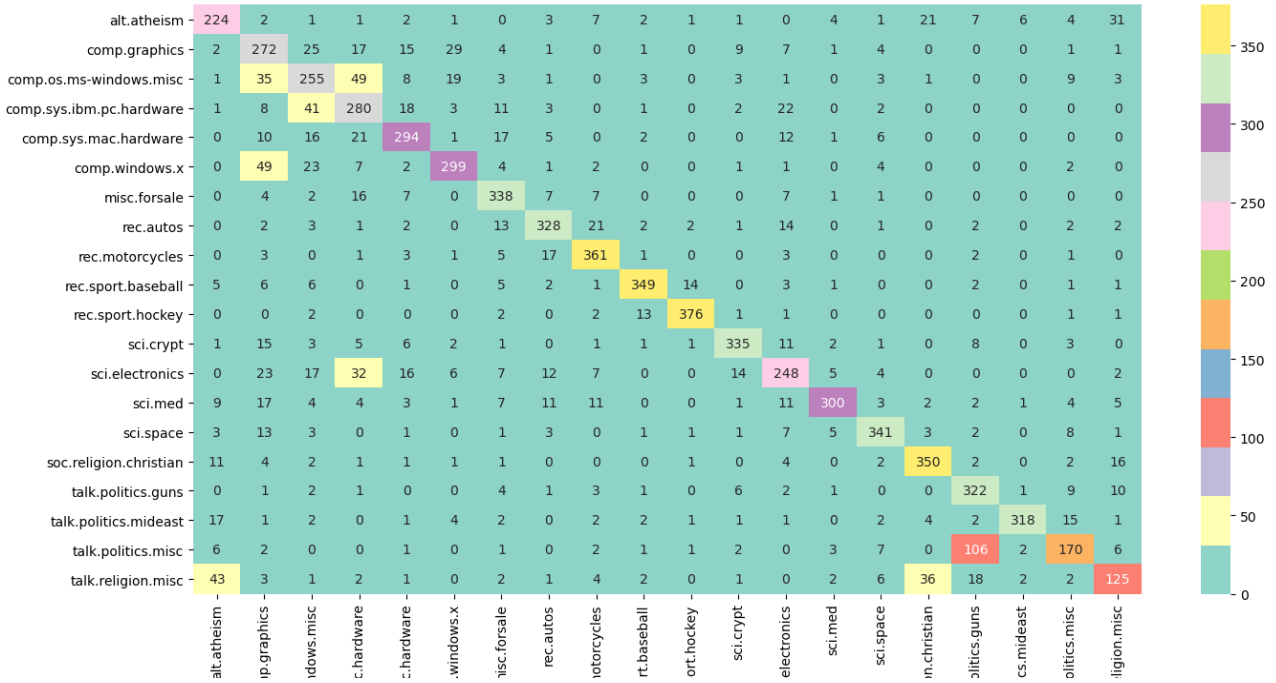
Q2) Class Prior Probabilities are found by using train data.

```
alt.atheism   0.04242531377054976
comp.graphics   0.05161746508750221
comp.os.ms-windows.misc   0.0522361675799894
comp.sys.ibm.pc.hardware   0.05214778150963408
comp.sys.mac.hardware   0.05108714866537034
comp.windows.x   0.05241293972070002
misc.forsale   0.05170585115785752
rec.autos   0.05250132579105533
rec.motorcycles   0.05285487007247658
rec.sport.baseball   0.052766484002121264
rec.sport.hockey   0.0530316422131872
sci.crypt   0.05258971186141064
sci.electronics   0.0522361675799894
sci.med   0.05250132579105533
sci.space   0.05241293972070002
soc.religion.christian   0.052943256142831886
talk.politics.guns   0.048258794414000356
talk.politics.mideast   0.04984974368039597
talk.politics.misc   0.041099522715220084
talk.religion.misc   0.033321548523952624
1.0000000000000002
```

Q3) While implementing the assignment, I both used the in build Multinomial Naive Bayes method of Scikit Learn and my own implementation. In both of these implementations Laplace Smoothing is used but I am not very informed about the details of the implementation of Scikit Learn algorithm. I got similar results as the other algorithm but my accuracy was lower.

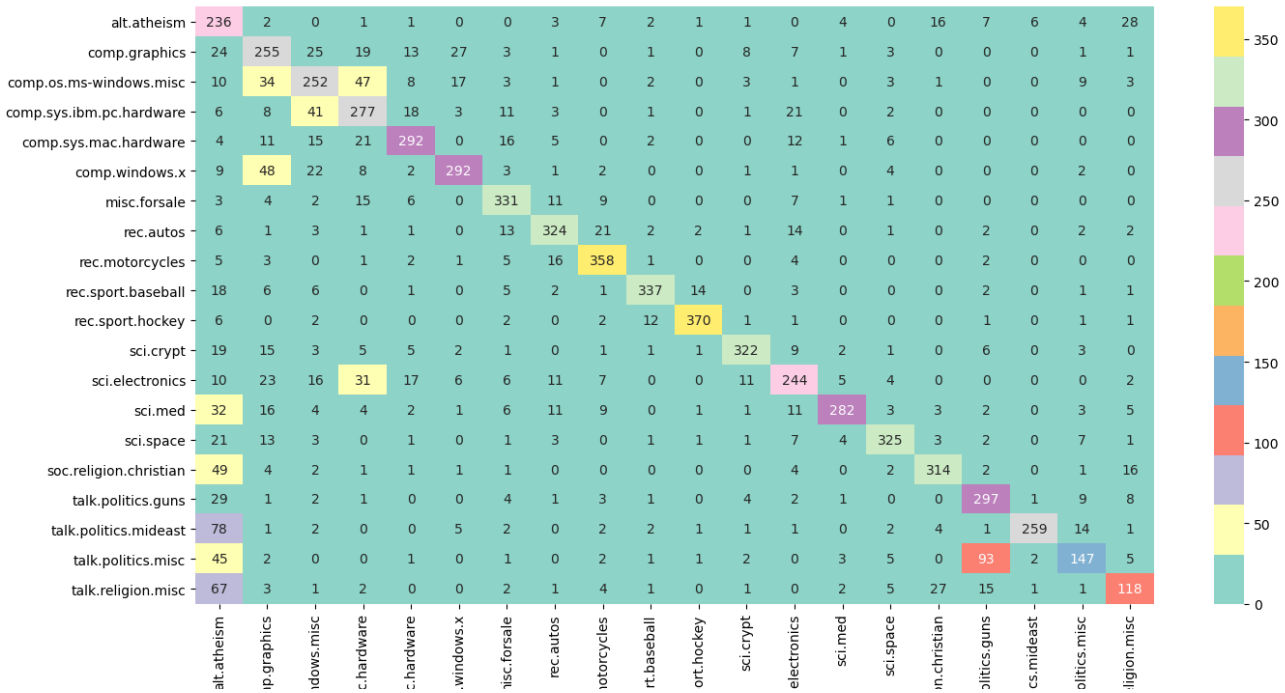
sklearn.naive_bayes.MultinomialNB():

Accuracy = 0.7813329792883696



My implementation:

Accuracy = 0.7477429633563463



Note: X-axis: Predicted, Y-axis: Actual

Q4) In my implementation, my algorithm mistakenly predicts some texts as alt.atheism class. I believe that this is because of the fact that some probability values are smaller than allowed

64 bit float value and as a result of that those probability values occurs as 0,0. It was worse before that I multiplied the values with some constant. When we look at the build-in algorithm, we can see that it mispredicted some of talk.politics.misc documents as talk.politics.guns. I assume that since they are in under the same subcategory, the words that are in those documents are really similar to each other and because of the fact that class prior of guns is bigger than misc, it might tent to has higher probability when multiplied.

Q5) I have chosen the most frequent words that are in these categories. Since they are most common ones they are the indicators of the class but some of the word are in all of the classes.Therefore, I have filtered some of the common ones. Resulting lists are as following:

alt.atheism :
say think god atheist thing could time well make even
comp.graphics :
graphic anyone file need help email program please image use
comp.os.ms-windows.misc :
window file use anyone using problem do driver system distribution
comp.sys.ibm.pc.hardware :
thanks system card drive problem use anyone also work computer
comp.sys.mac.hardware :
mac apple thanks problem anyone distribution use work computer new
comp.windows.x :
x window problem using work distribution application help email system
misc.forsale :
sale offer usa shipping interested price computer sell condition system used asking
rec.autos :
car distribution good think usa much new make engine specs
rec.motorcycles :
dod bike distribution go think time good ride make new
rec.sport.baseball :
game year baseball team last player good time run distribution
rec.sport.hockey :
team game hockey year play nhl player time playoff good
sci.crypt :
key clipper chip encryption system distribution government people use good
sci.electronics :
use anyone need work good distribution want could thanks used
sci.med :
also like time replyto think science year problem good may
sci.space :
space year also distribution think time could first nasa thing
soc.religion.christian :
god christian say think question time believe jesus see may

talk.politics.guns :
gun get think distribution make right time well even say
talk.politics.mideast :
israel israeli way world time say year right state many

talk.politics.misc :
state think get make time say new even government many
talk.religion.misc :
christian say god think time way also well even good

Q6) While calculating the probabilities, the result of the calculation is usually smaller than the allowed 64 bit float value. As a result of that, the first accuracy result was lower than 65 percent. In order to solve this problem, I multiplied all of the probability values by some constant and obtained this result. However, as I have seen it in internet, this problem would be solved using log probability. The other thing would be removing words like “Line”, “Subject” which are used in all of the documents. There are also other advanced techniques for improving the result. We could use n-grams by instead of counting single words, we could count sequences of words or we could use TF-IDF by instead of just counting frequency we could penalizing words that appear frequently in most of the samples.