1B: NAÏVE BAYES CLASSIFIER

Description and Implementation:

Naïve Bayes classifier is based on the principle of bayes theorem in probability.

P(A/B)=P(B/A)\*P(A)/P(B)

Therefore, to find whether a mail is spam or not, we will find probability of the mail being spam or not if we are given a word.

P(spam/word)=P(word/spam)\*P(spam)/P(word)

P(word/spam),P(spam),P(word) will be found through the training data.

P(word)=P(word/spam)\*P(spam)+P(word/nspam)\*P(nspam)

P(word/spam)=no(w/spam)+alpha/n(spam)+alpha\*n(words)

no(w/spam)=no. of word repeats in spam messages

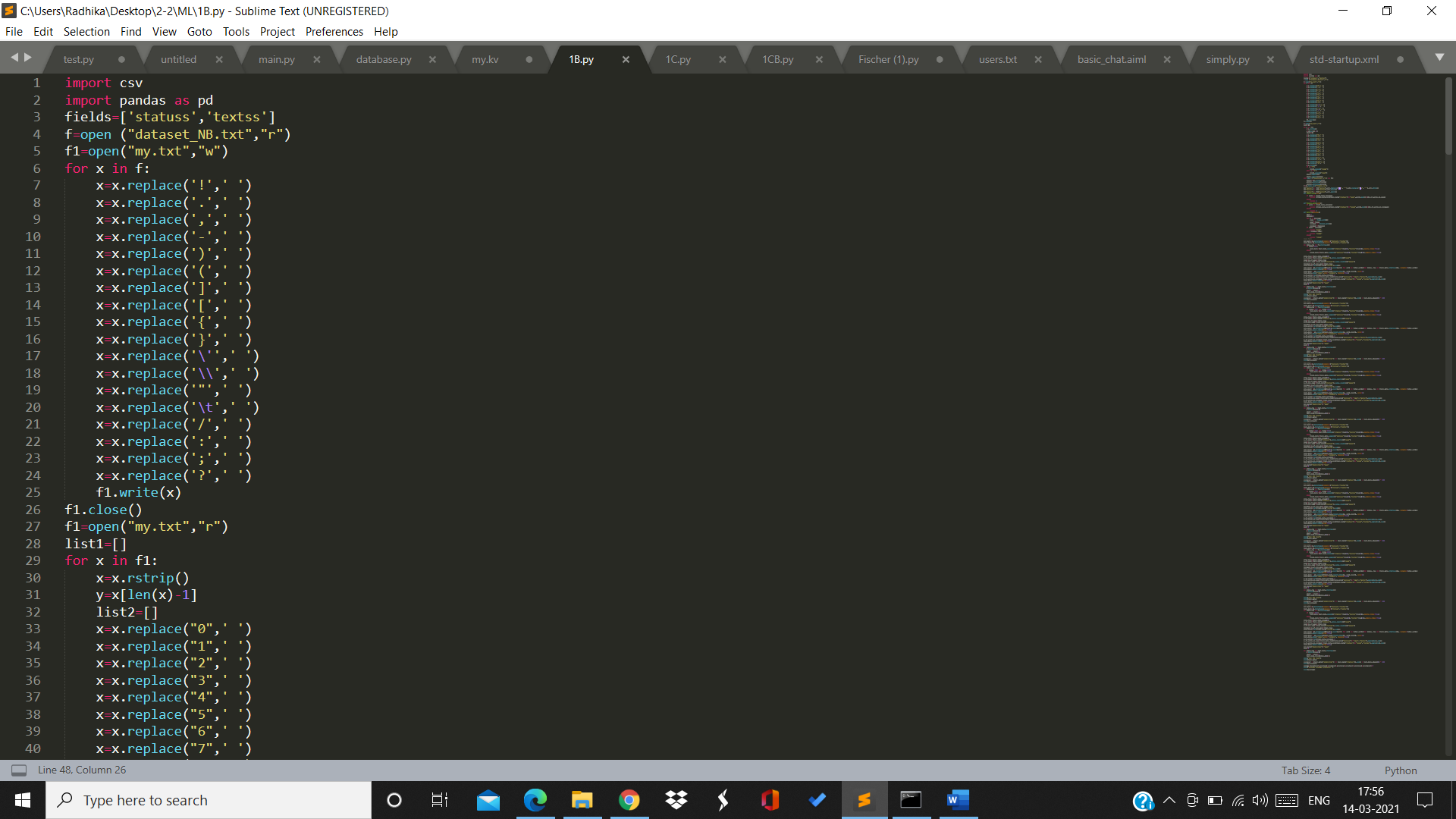
n(spam)=no. of words in spam

n(words)=total no. of words

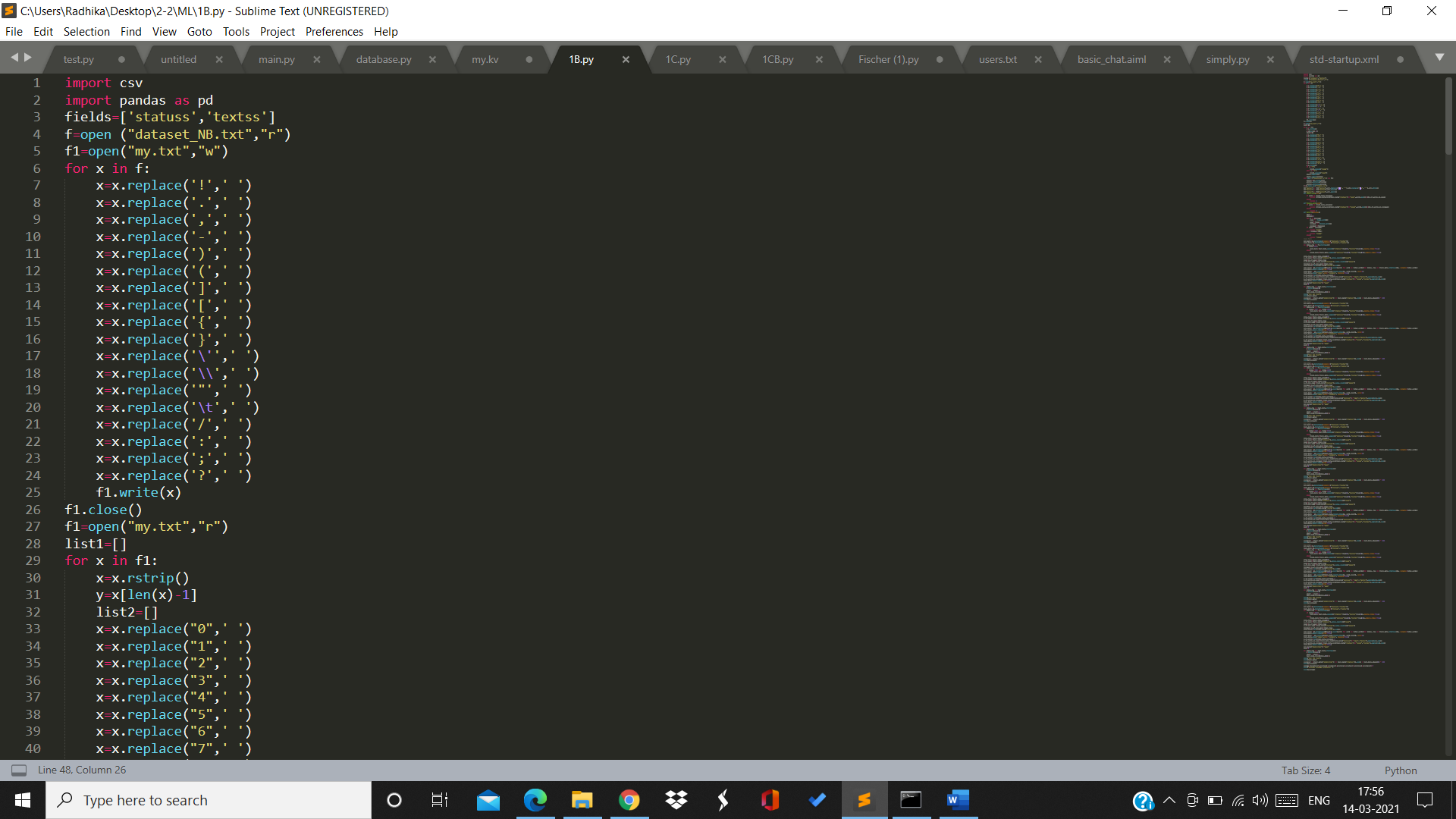
alpha=coefficient when a word is missing in the dataset

Implementation of this model:

We will need the necessary imports of pandas and csv

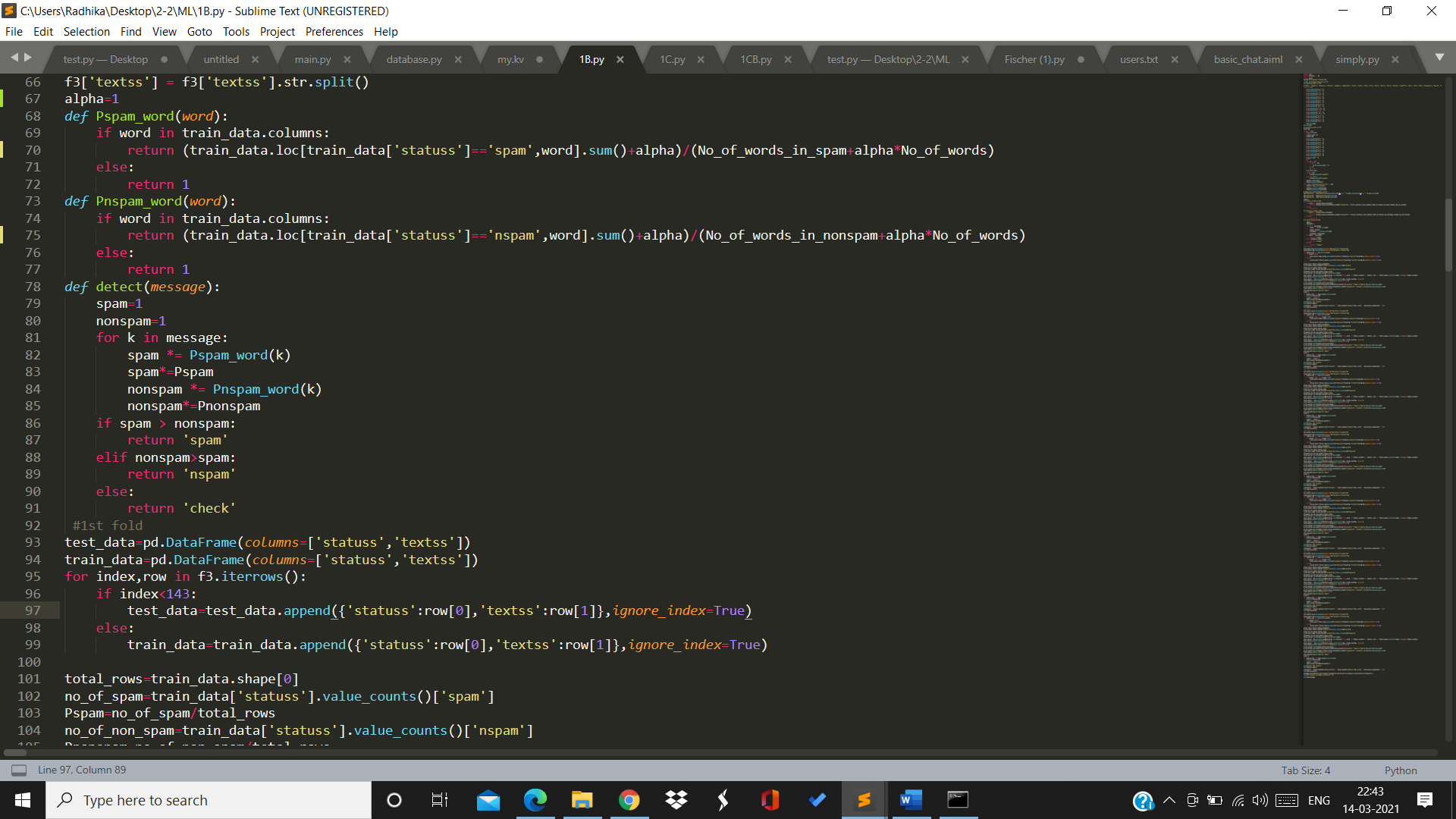


Then, we have to given file in .txt to .csv and remove all the full stops, commas, and unnecessary words that does not affect the sentiment of the text.



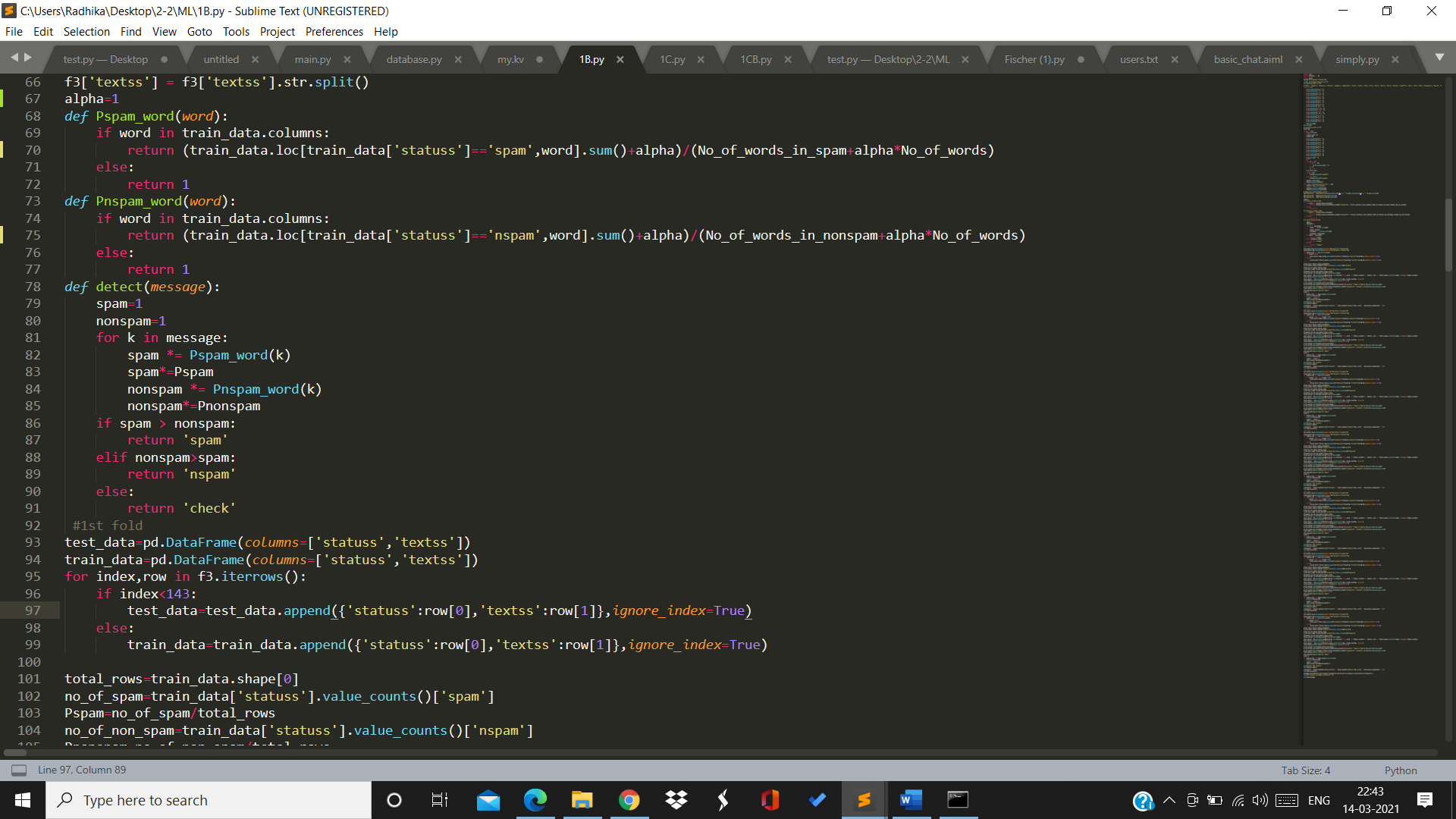
Pspam\_word is a function which will return the probability of a mail being a spam if given a word.

train\_data.loc[train\_data['statuss']=='spam',word].sum() will find the the no. of times that word is repeated in spam messages



Pnspam\_word is a function which will return the probability of a mail being a non spam if a word is given.

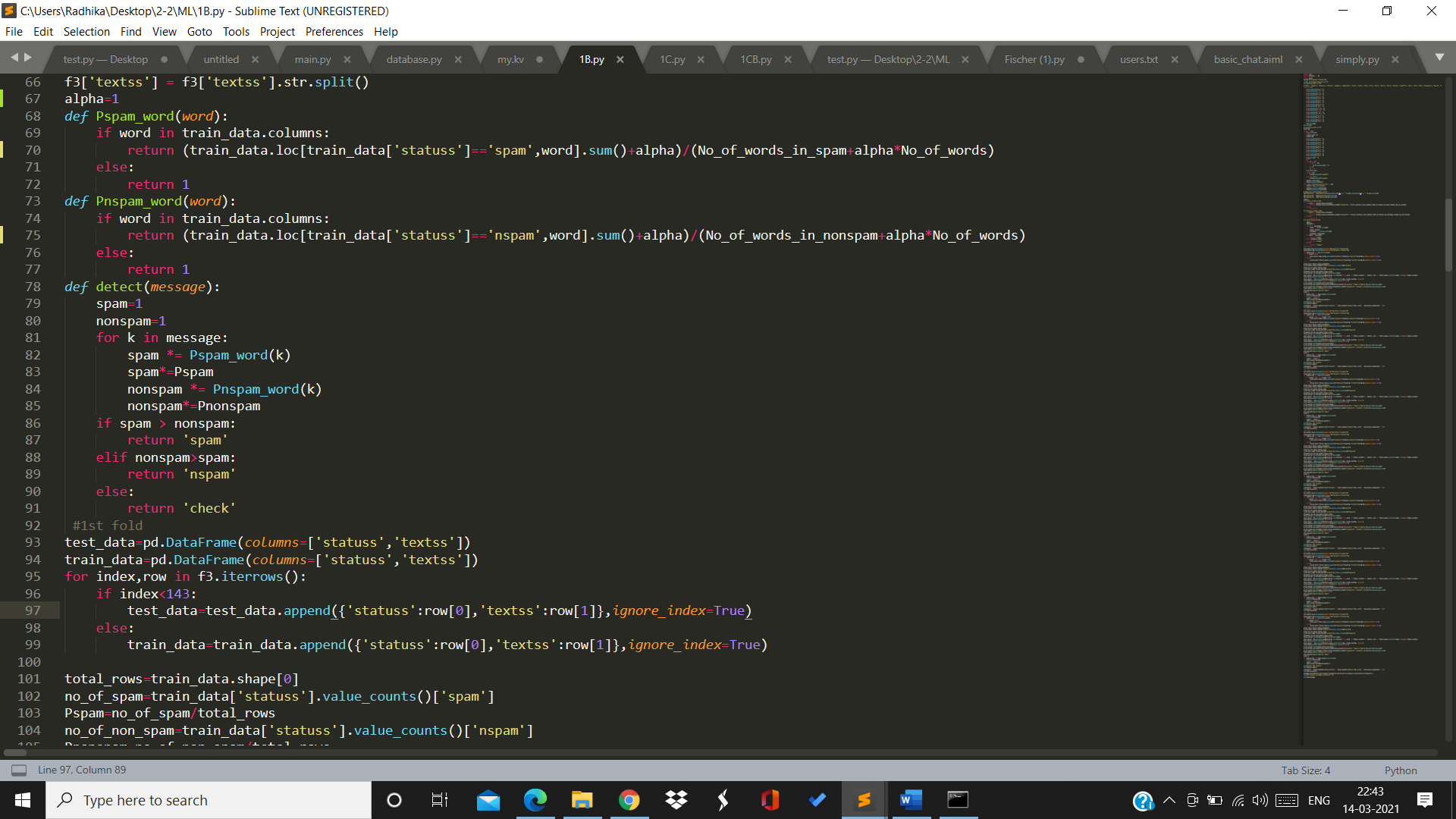
train\_data.loc[train\_data['statuss']=='nspam',word].sum() will find the the no. of times that word is repeated in non spam messages



Detect is a function which will classify if the given message is spam or non spam.

We will find the probability of each word in the given message to be in spam/non spam.

If the probability of the message in spam>probability of message in non spam then the function will return spam. If the probabilities are equal, then the function will return that human intervention is needed because it can’t be classifies as spam/non spam. Otherwise, non spam will be returned.



Because we have to do 7 fold cross validation, we will divide the dataset into 7 parts each fold will have test and train data.

Total\_rows is the total no. of rows in the train data

no\_of\_spam is the total no. of spam messages in the training data

Pspam is the probability of spam messages in the training data

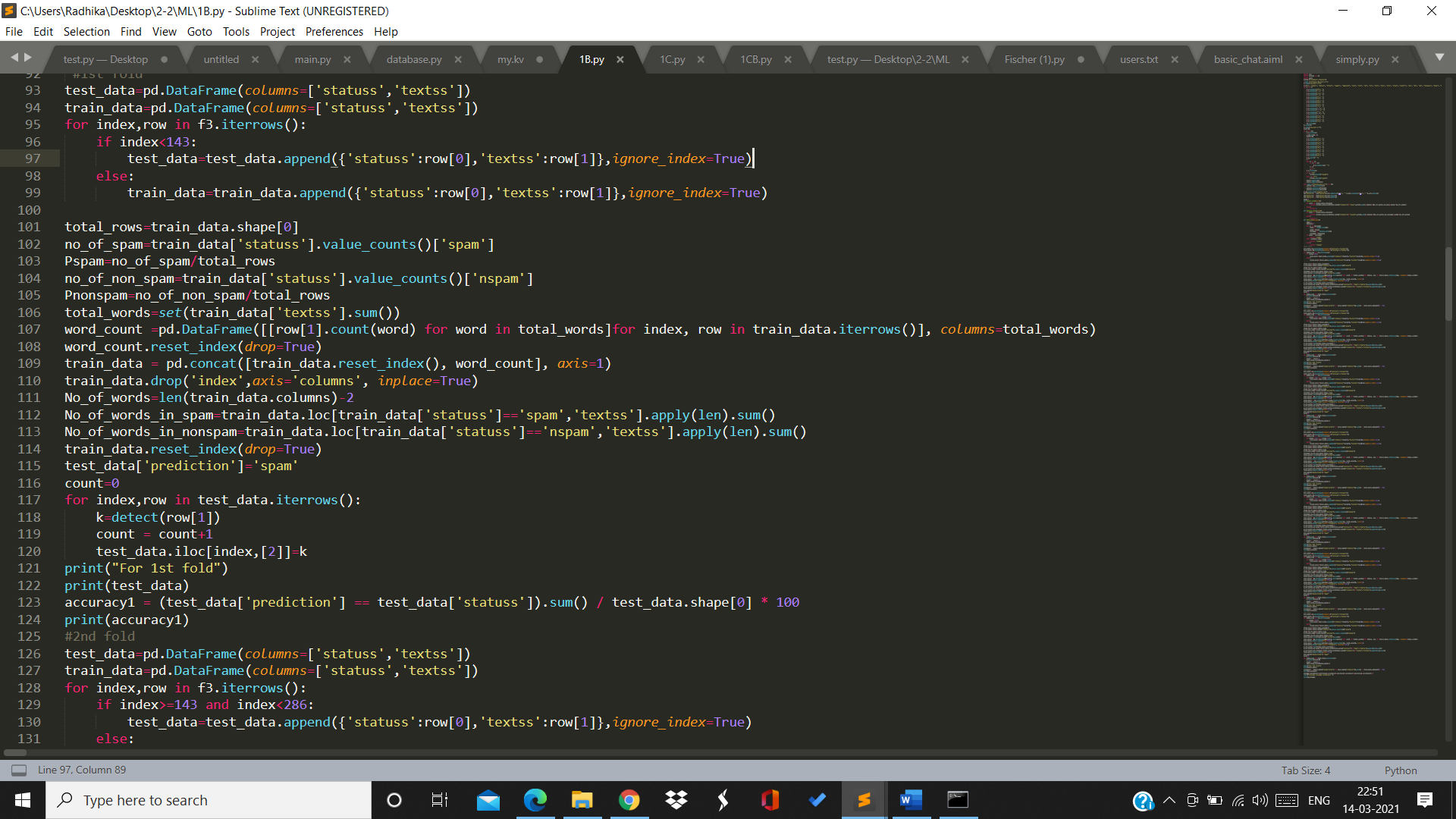
no\_of\_non\_spam is the total no. of non spam messages in the training data

Pnonspam is the probability of non spam messages in training data

Total\_words is the total no. of distinct words in the training set

We will train our model and then classify each message in testing set and verify it with the classification given before.

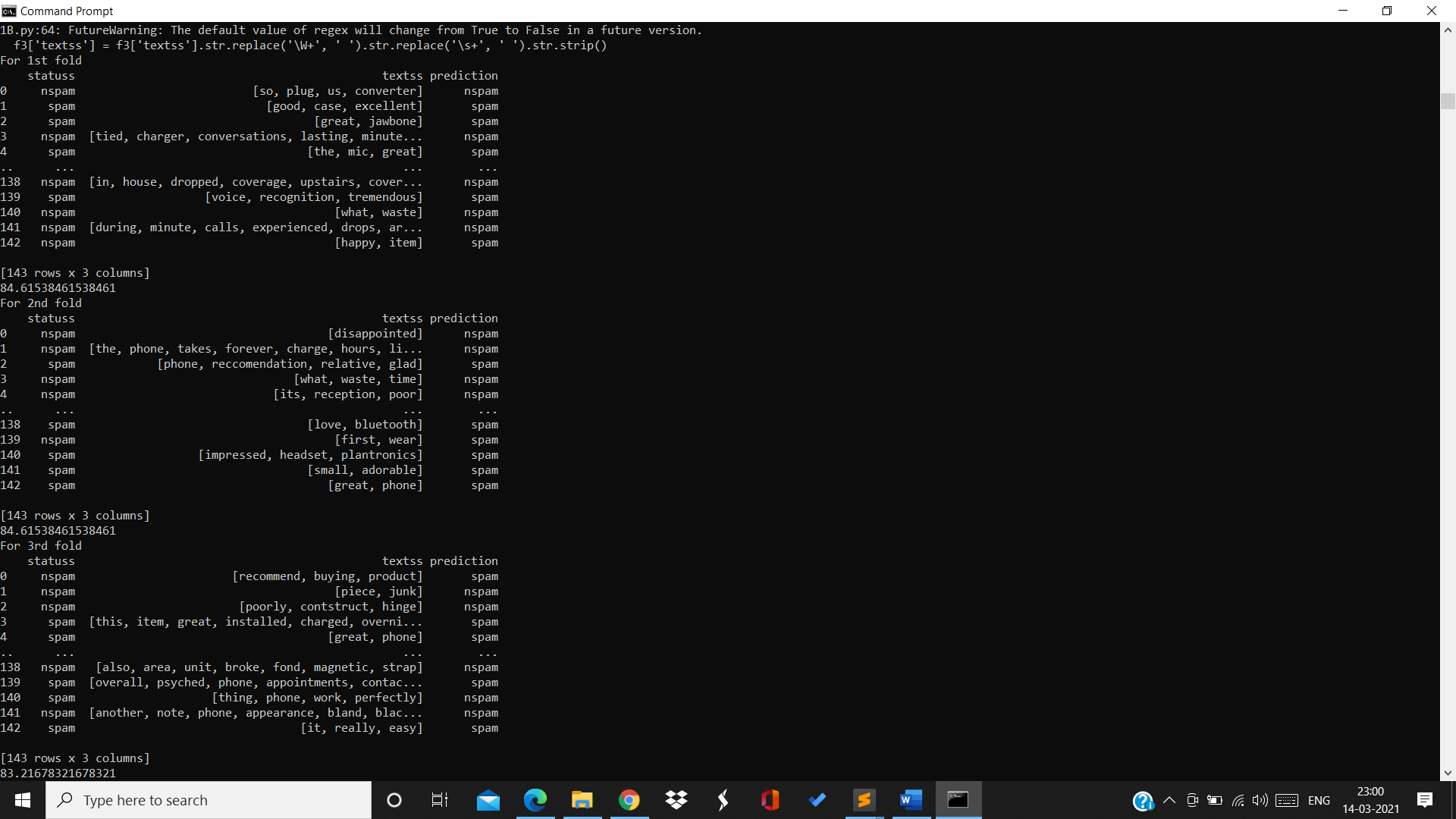
Thus we will find how accurate our model is and repeat this 7 times to find the overall average accuracy of our model.



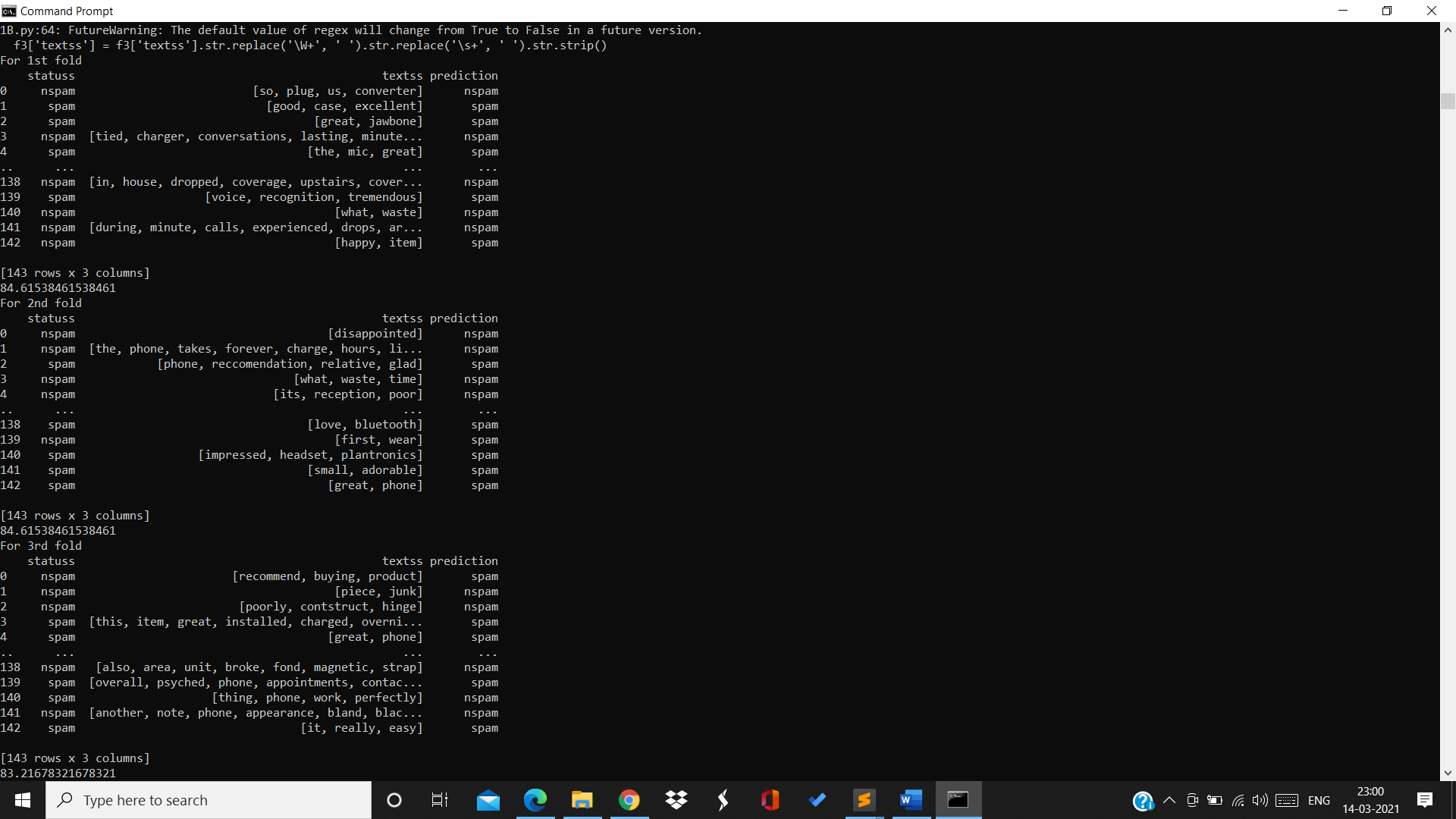
Output:

For 1st fold:

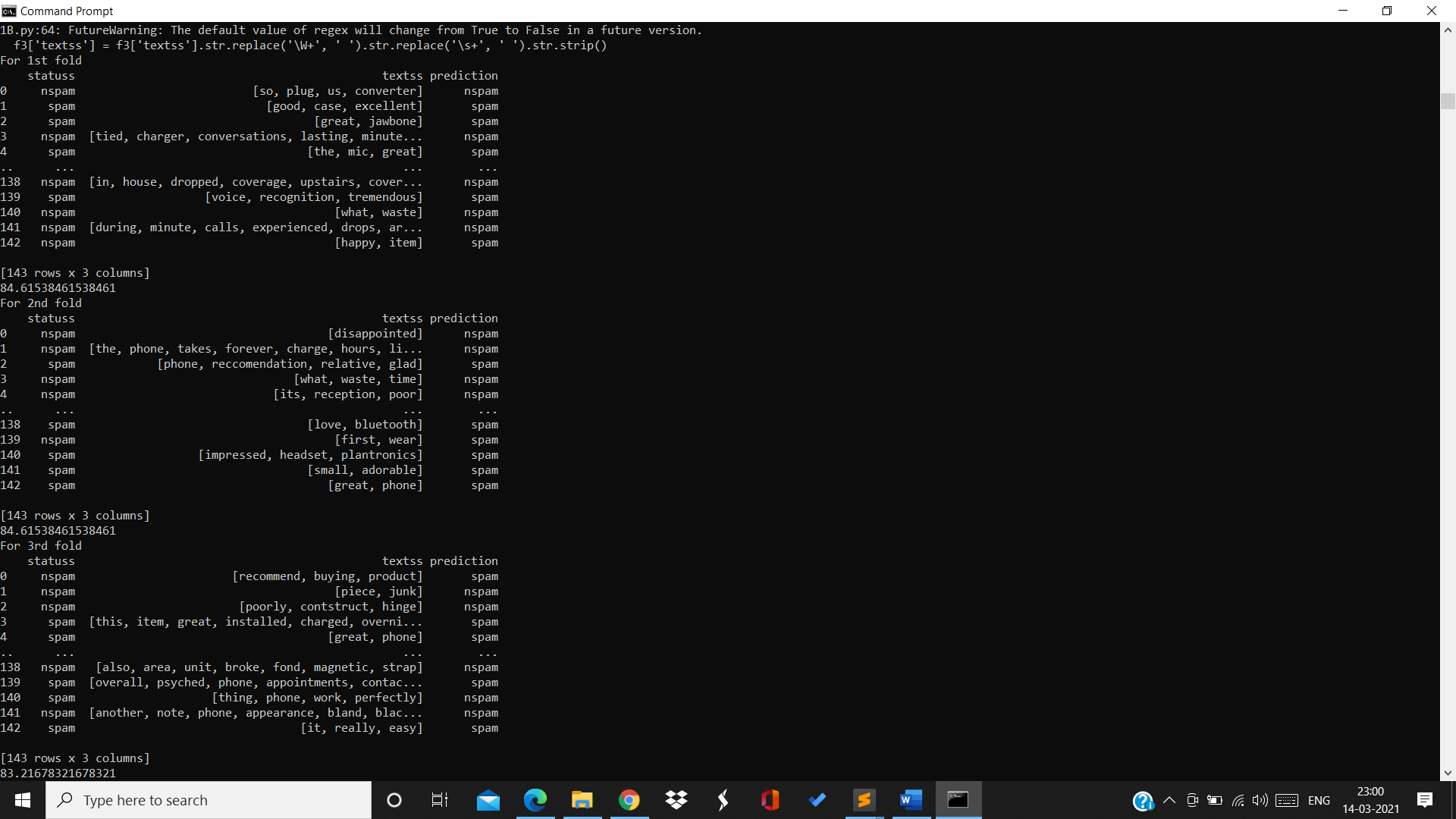
Accuracy=84.6%



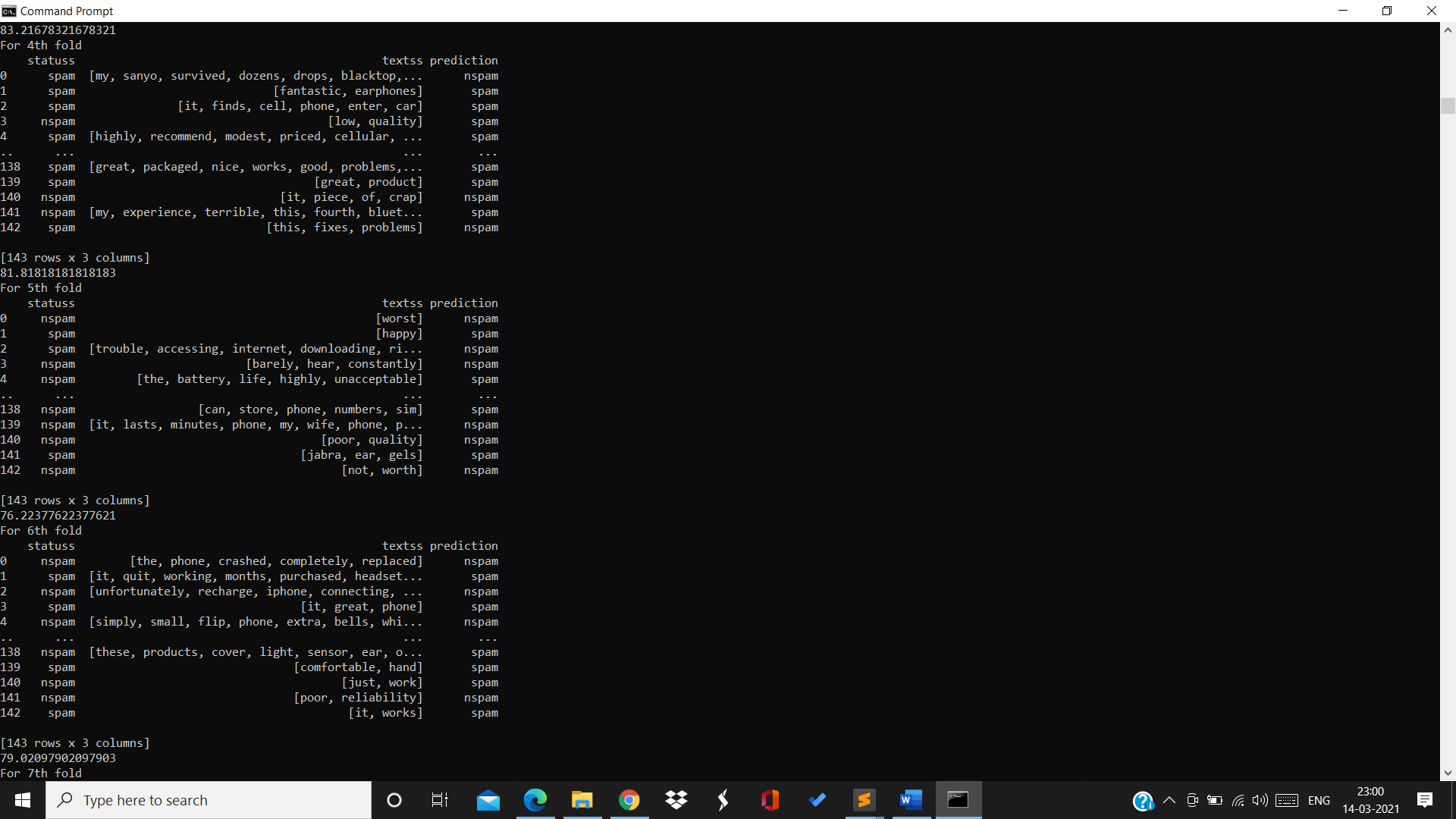
For 2nd fold: Accuracy=84.6%



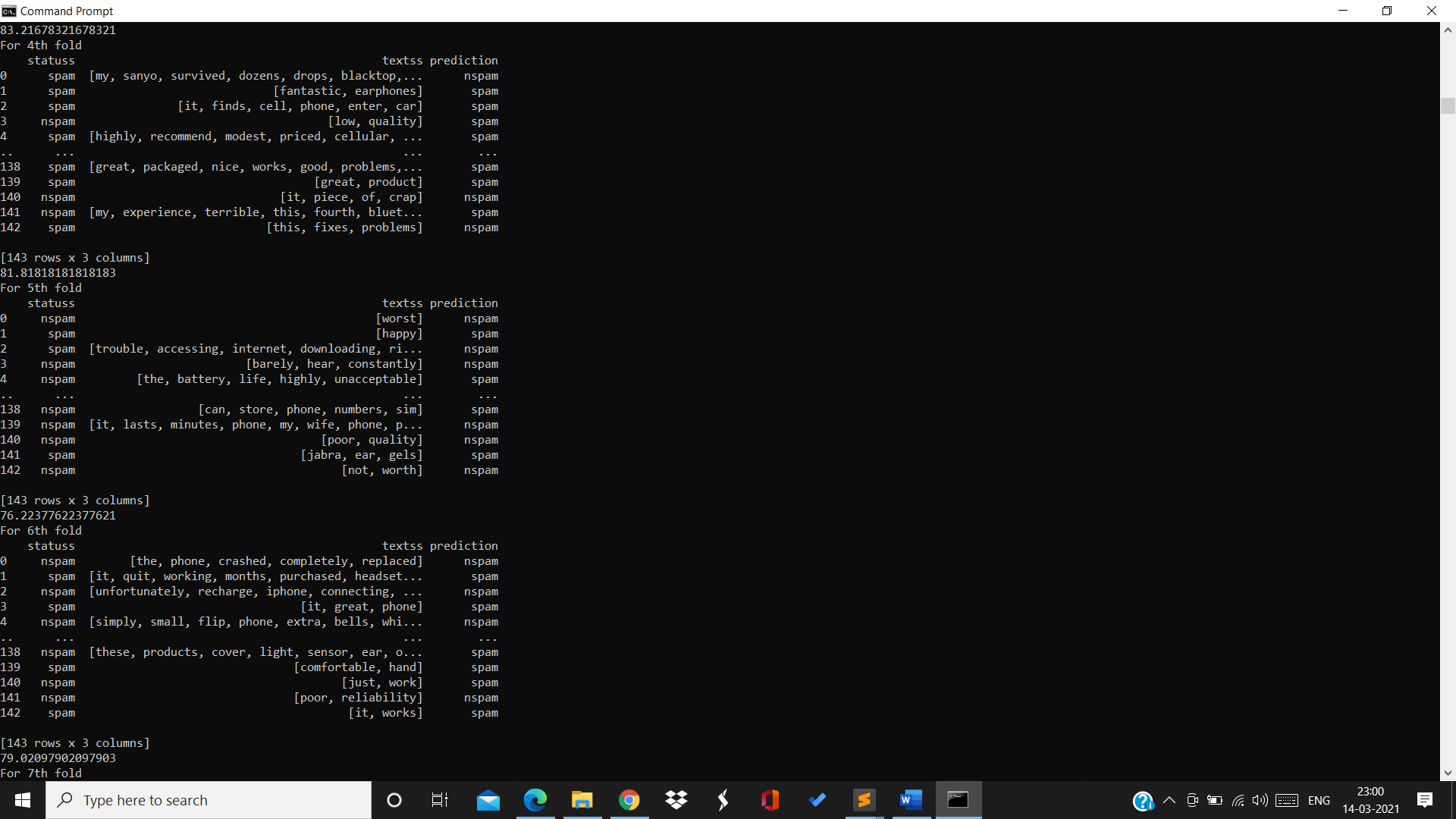
For 3rd fold: Accuracy=83.2%



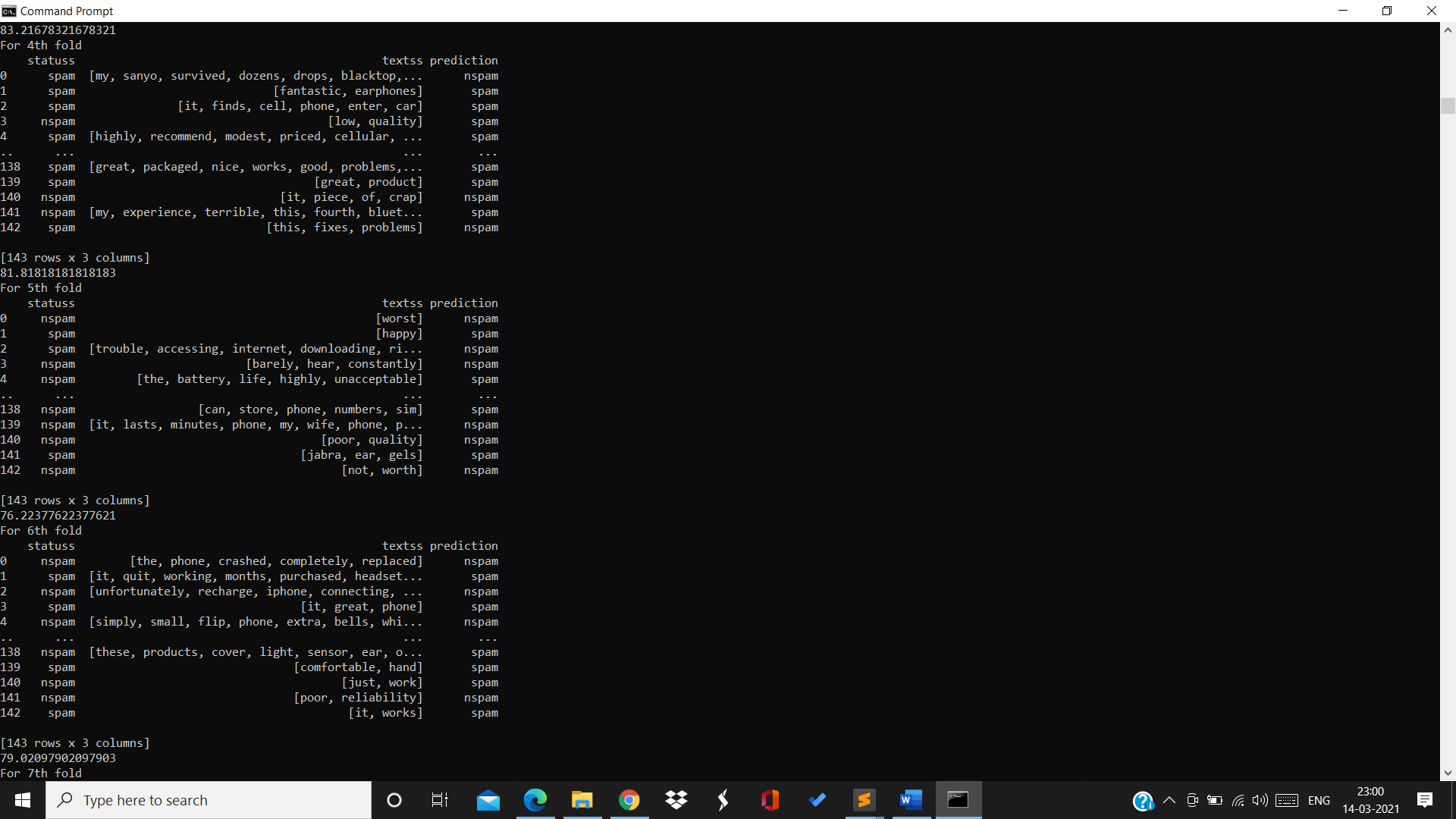
For 4th fold: Accuracy=81.8%



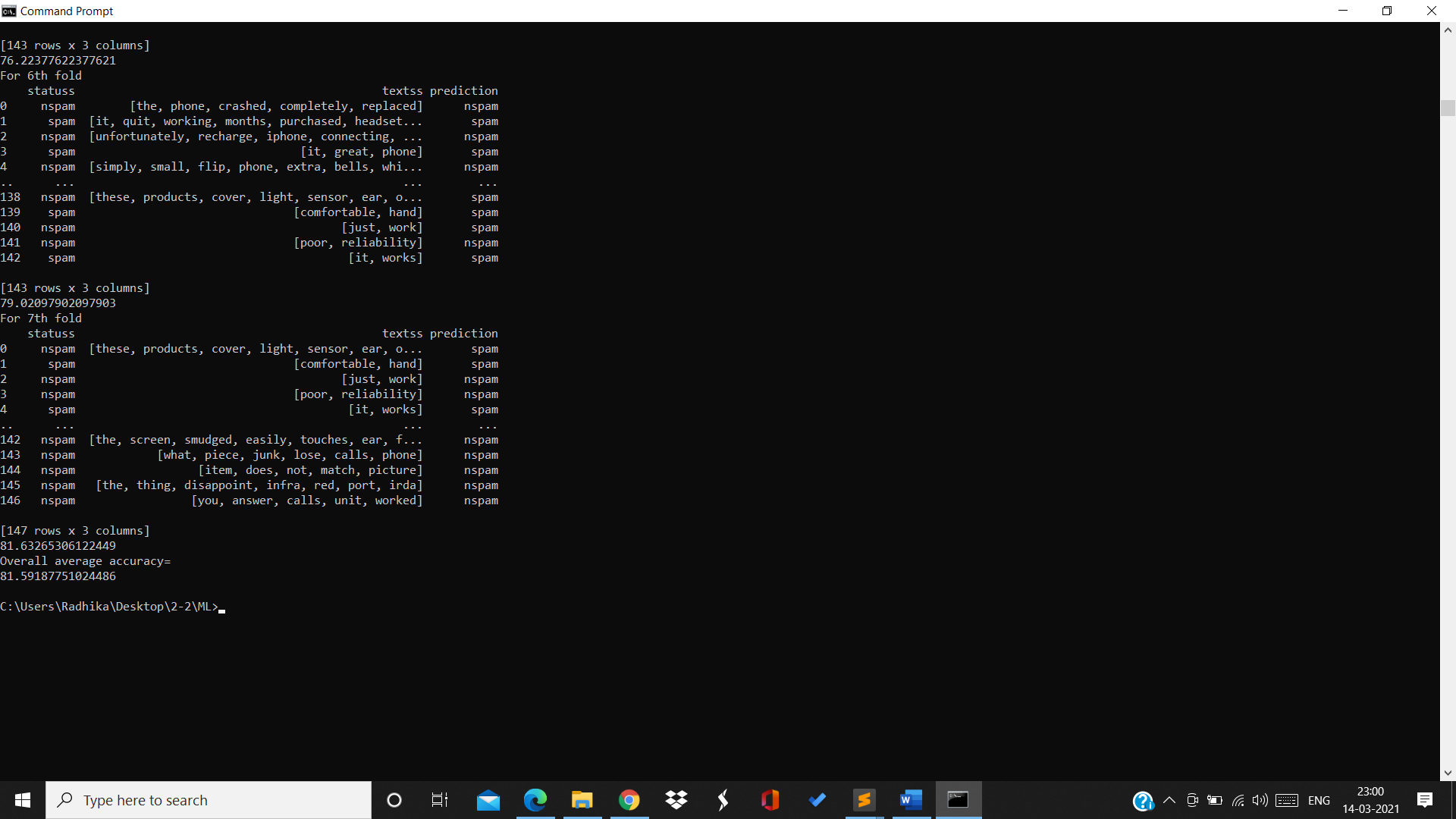
For 5th fold:Accuracy=76.2%



For 6th fold:Accuracy=79%



For 7th fold:Accuracy=81.63



Overall average accuracy=81.59%

DISADVANTAGES

* Naive Bayes assumes that all predictors (or features) are independent, rarely happening in real life. This limits the applicability of this algorithm in real-world use cases.
* This algorithm faces the ‘zero-frequency problem’ where it assigns zero probability to a categorical variable whose category in the test data set wasn’t available in the training dataset.
* Its estimations can be wrong in some cases, so you shouldn’t take its probability outputs very seriously.

By:

Radhika Sigtia

Sharvani

Bathini Akash