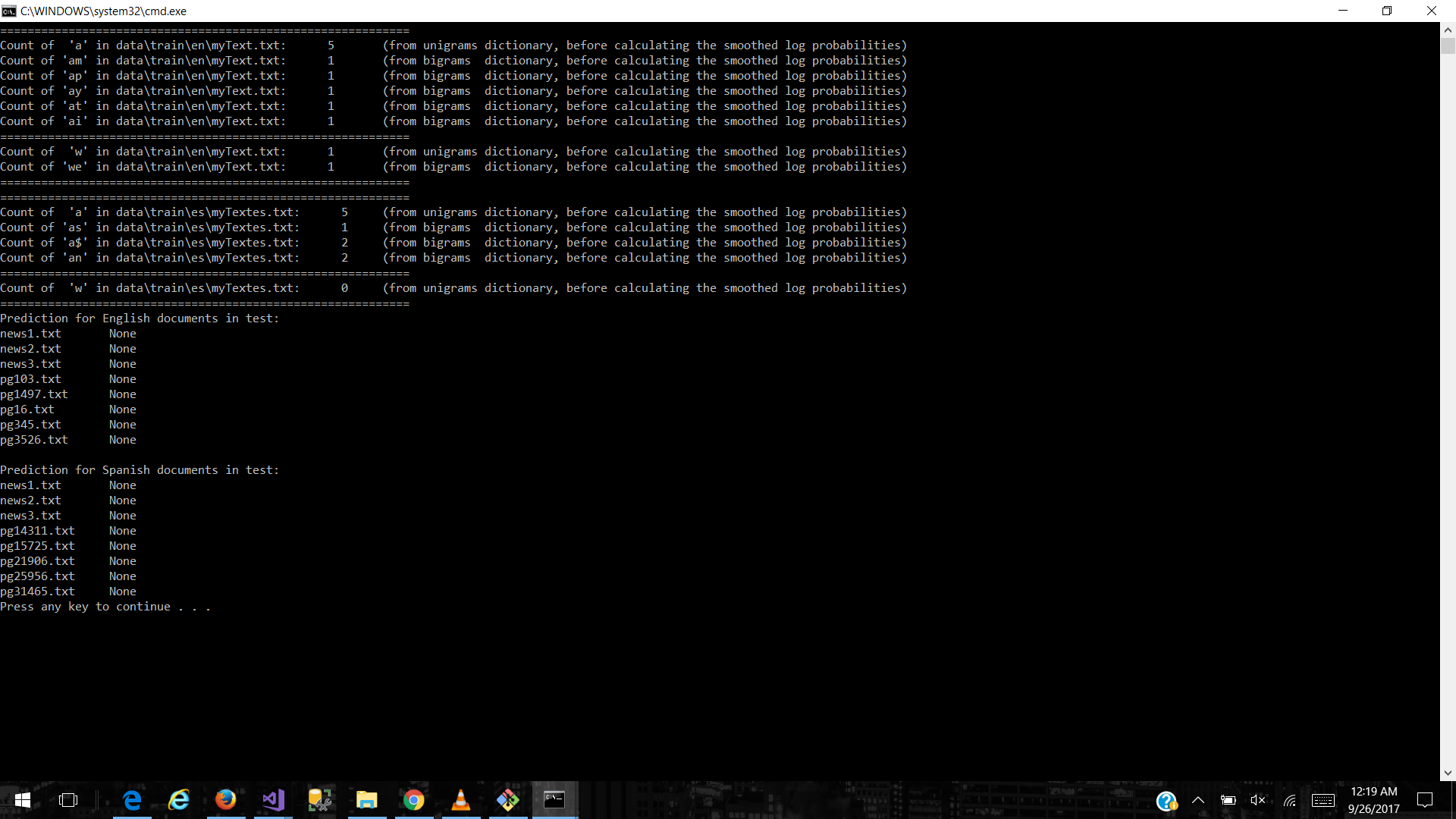
NATURAL LANGUAGE PROCESSING

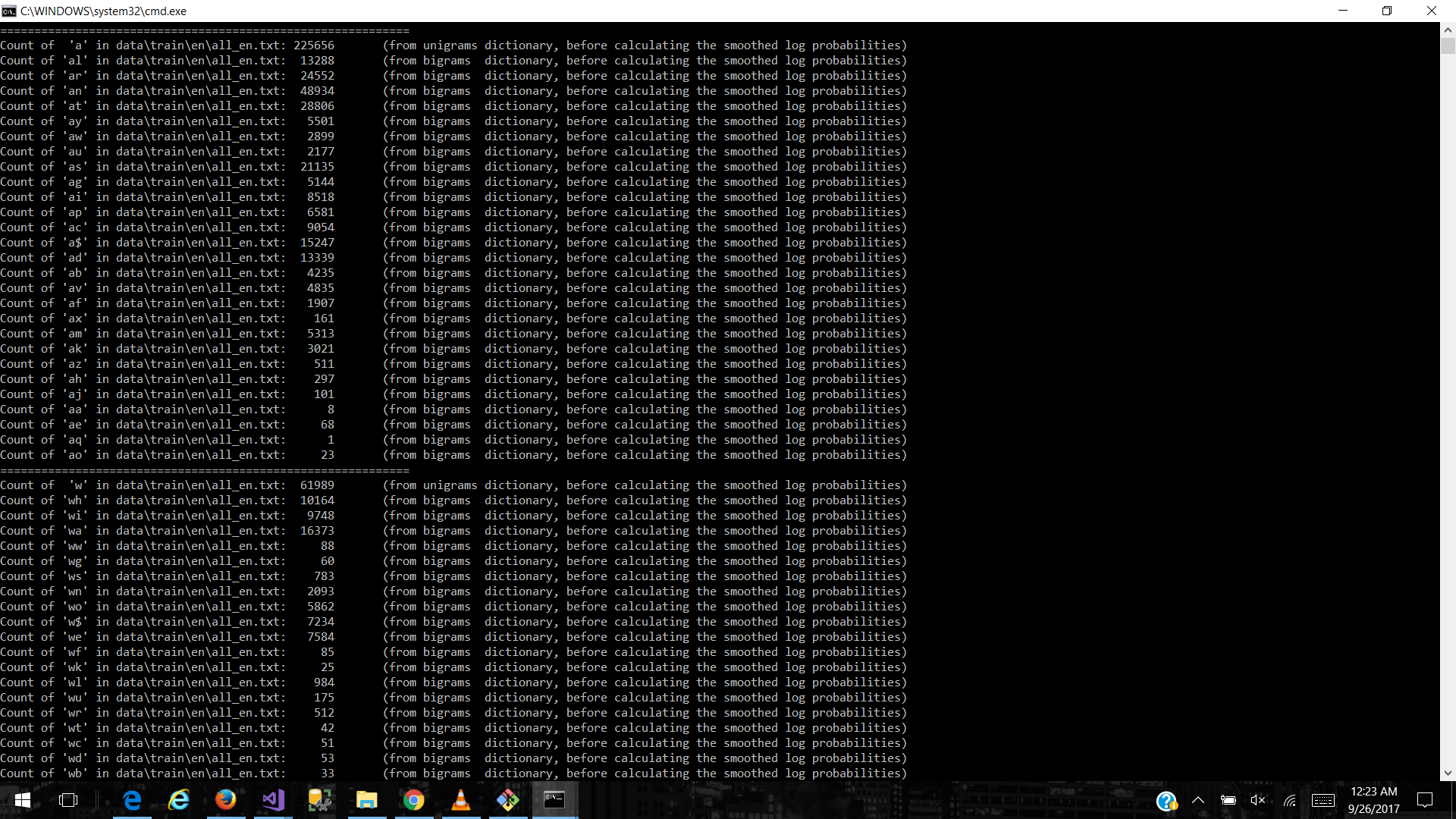
Language Detectors

ZAKIR SHAIK

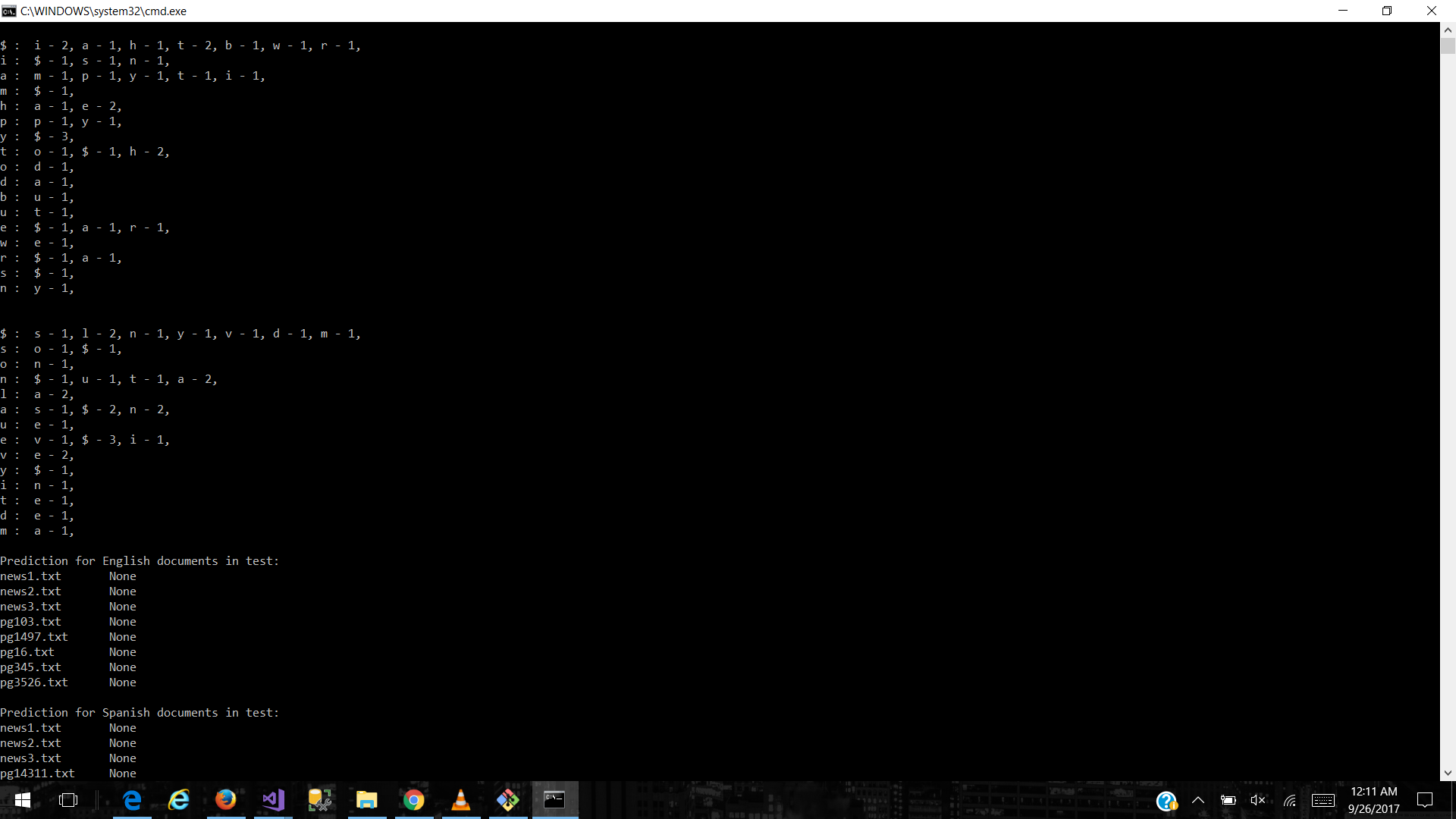
I have implemented language detector using unigrams, bigrams and trigrams in Microsoft Visual Studio. First I wrote code for counting both unigrams and bigrams and tested it on “I am happy today but the weather is rainy”. I got the following output.



Then I executed it on the all\_en.txt and all\_es.txt files and I got the following output.



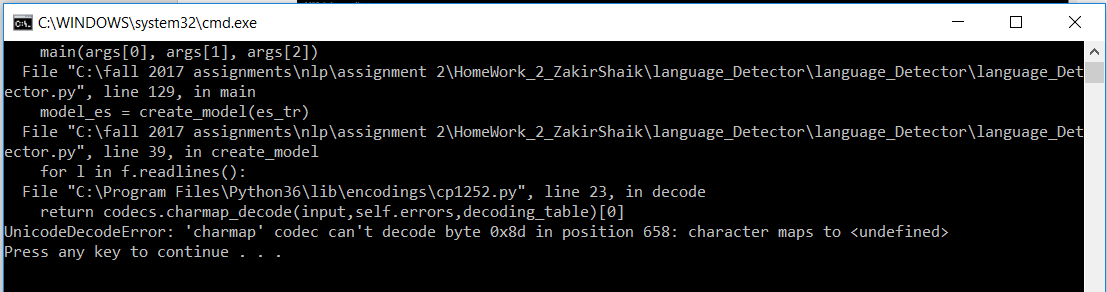
I then tried to see what do lambda:collections.defaultdict(int) do. For that I have written prints and I found that it will assign 0 to value of the key. For example: Consider $i. If $ is not present, then it will assign i to $ and give 0 as its value(count). I got the following outputs for English and espanol.



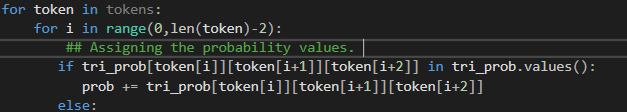
Then I defined the calc\_prob() function which will take token list and model and give the probability. Finally I defined predict() function in such a way that the probabilities are summed up and the one with more probability will be the language in the file.

CHALLENGES FACED:

1. I could not open the file which has other language than English. When I implemented the code without encoding utf-8, my code executed on English files and for the files containing Spanish I was getting the following error.

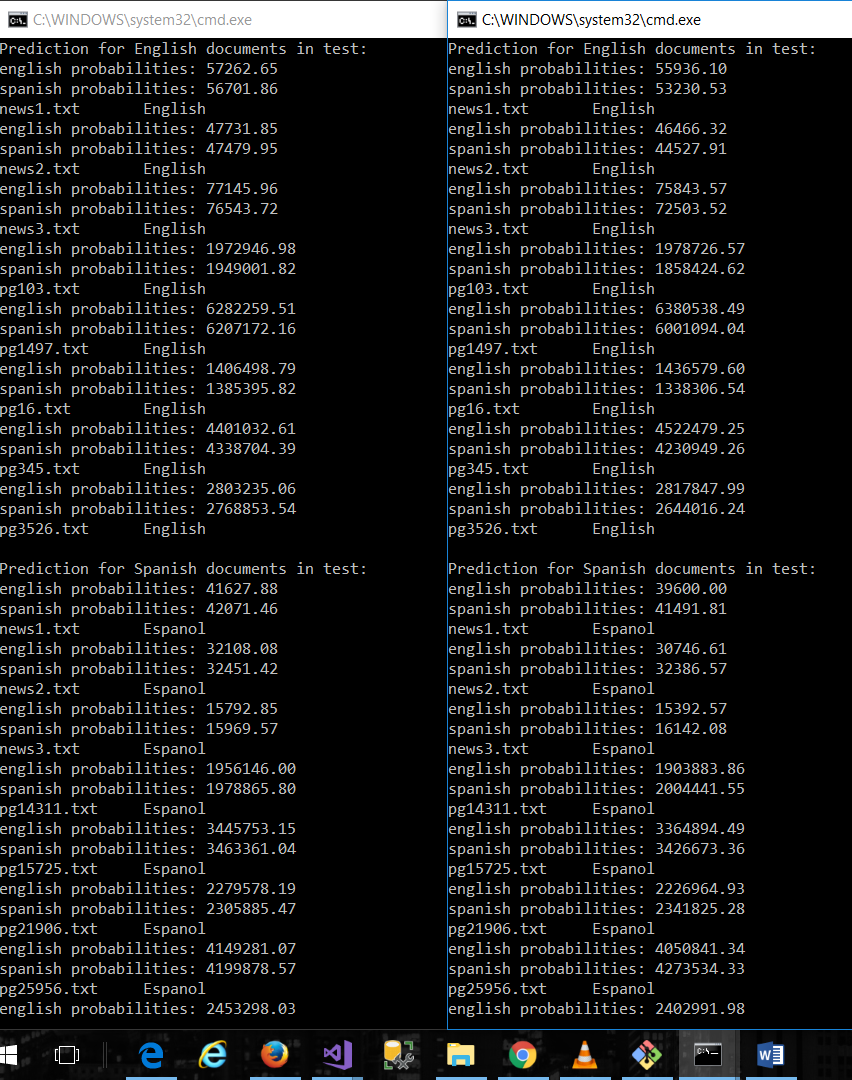


1. I also got string index exceeded error for which I spent more than an hour. The following is the place where I encountered it.



Here I placed len(token)-1 but later I came to realize that at the end we have to leave the last two digits as we can’t start from last but one. Same is with bigrams.

1. I also realized that as most of the alphabets in both English and Spanish are nearly same and enough words are same, the probabilities were on the edge mark. i.e. Close to each other with a little gap.
2. **Do you think it makes sense to create a language model at the character level instead of at the word level for this task? Why?**
3. No, I do not agree with it. If we observe the following results for both bigrams and trigrams we can see that the probability difference between them has changed. For bigrams the sum of probabilities for all letters are very close to each other when compared to trigrams. So, if it were a word, we can predict the language more accurately. THE FOLLOWING IS THE SUM OF PROBABILITIES FOR ALL TOKENS IN THEIR RESPECTIVE FILES.



1. **Take a look at the test documents for English and Spanish. Are the documents written only in one language?**
2. I couldn’t find Spanish words in English documents. But in Spanish documents there are some sentences written in English.
3. **What is the minimum number of tokens you need to process to always make prediction when testing?**

I have done around 7 tests and I found that 2-3 pages of raw data in its own language is enough to predict the testing data. Initially I ran for one single page of raw data. I got English for all. Next I ran for 8-10 pages. I found that I am getting the correct output. Then I decreased and I found that 2-3 pages of raw data is enough. The following are the sample outputs.

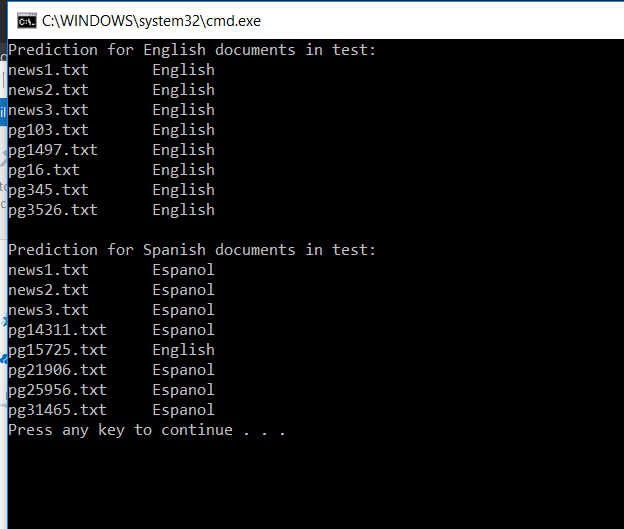
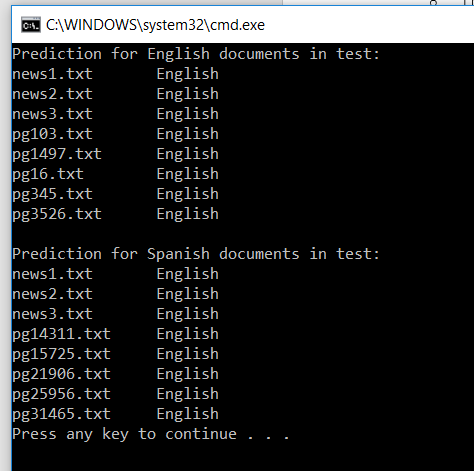


Fig: 2 pages of raw data.

  
Fig: Only one page of raw data.

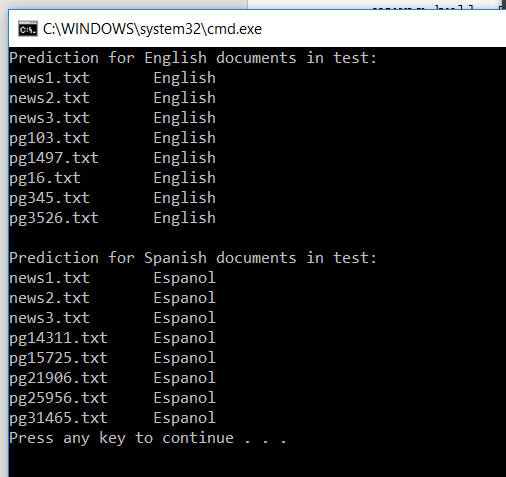


Fig: For 3 pages of raw data on trigram.

1. **If you create several models for English and Spanish (using different training data and different preprocessing), how can you compare them?**
2. I would compare them on the basis of total training data available, the logarithmic probability prediction for files of both types.
3. **Can you train with less training data and still get the right predictions? How does training size affect predictions during testing?**

I have taken less amount of training data in both English and Spanish files. Slowly, I started increasing training data in both the all\_en.txt and all\_es.txt files by a measure of 2 increments and I observed only one file in testing data for each execution ie. News1.txt in both en and es testing data. I observed that as we increase the training data we get more data to compare and test and hence the probabilities also increase. The above graph is for Prediction of English language and “Series 1” here is the “English Probabilities sum”, “Series 2” is the “Spanish Probabilities sum”. The below is the graph for the Prediction of Espanol.

Here “Series 1” represents “English probability sum” for that file News.txt in “es” folder. “Series 2” represents “Spanish Probability sum” for the file News.txt in “es” folder with the increasing in training data gradually.