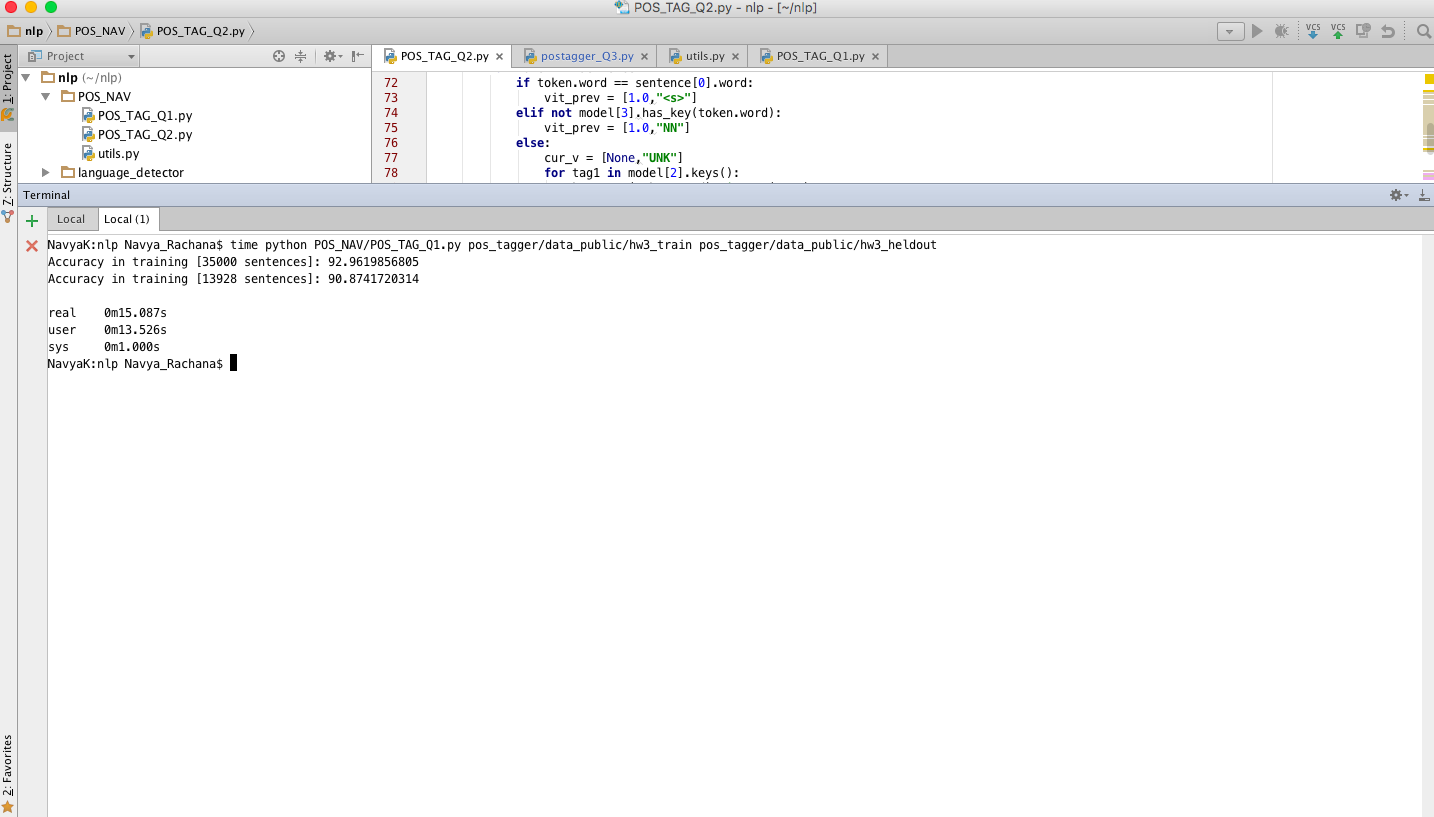
**Question 2: Most Likely tag for POSTAGGER:**

**implementation:**

* Initially the program counts all the tags, then the program finds the most likely tag for each word in the training set
* It then assigns the most likely tag to all the words in the testing set and “NN” for unseen words and then calculates the accuracy

**Results: (LOWERCASING)**



Accuracy in training [35000 sentences]: 92.9619856805

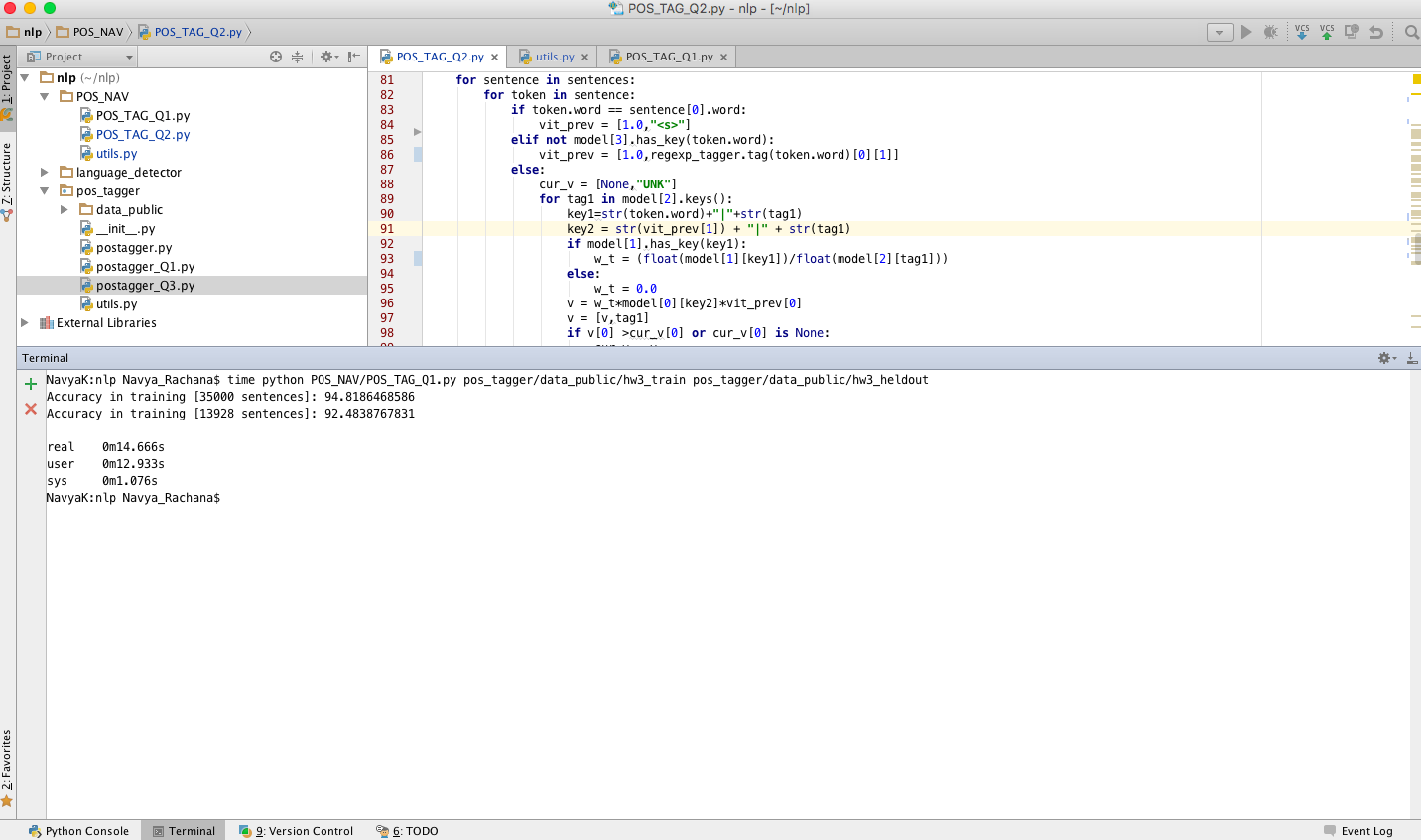
Accuracy in training [13928 sentences]: 90.8741720314

real 0m15.087s

user 0m13.526s

sys 0m1.000s

**Results: (NO LOWERCASING)**



Accuracy in training [35000 sentences]: 94.8186468586

Accuracy in training [13928 sentences]: 92.4838767831

real 0m14.666s

user 0m12.933s

sys 0m1.076s

**Question 3: HMM POS Tagging**

**Implementation:**

**Creating Model:**

* The function initially calculates all the counts of words, tags, word|tag and tag|tag
* It then calculates the probabilities of word|tag and tag|tag
* The model then returns the probabilities and the tags count probabilities

**Predicting test results:**

* The Predict model calculates the Viterbi matrix in two ways: If the word is the start of the sentence it would calculate the from the Viterbi (<s>|<s>) which is taken as 1
* If the word is not the first word of the sentence it then calculates the Viterbi of the previous word|tags and finally get the result

**Problems during Implementation**

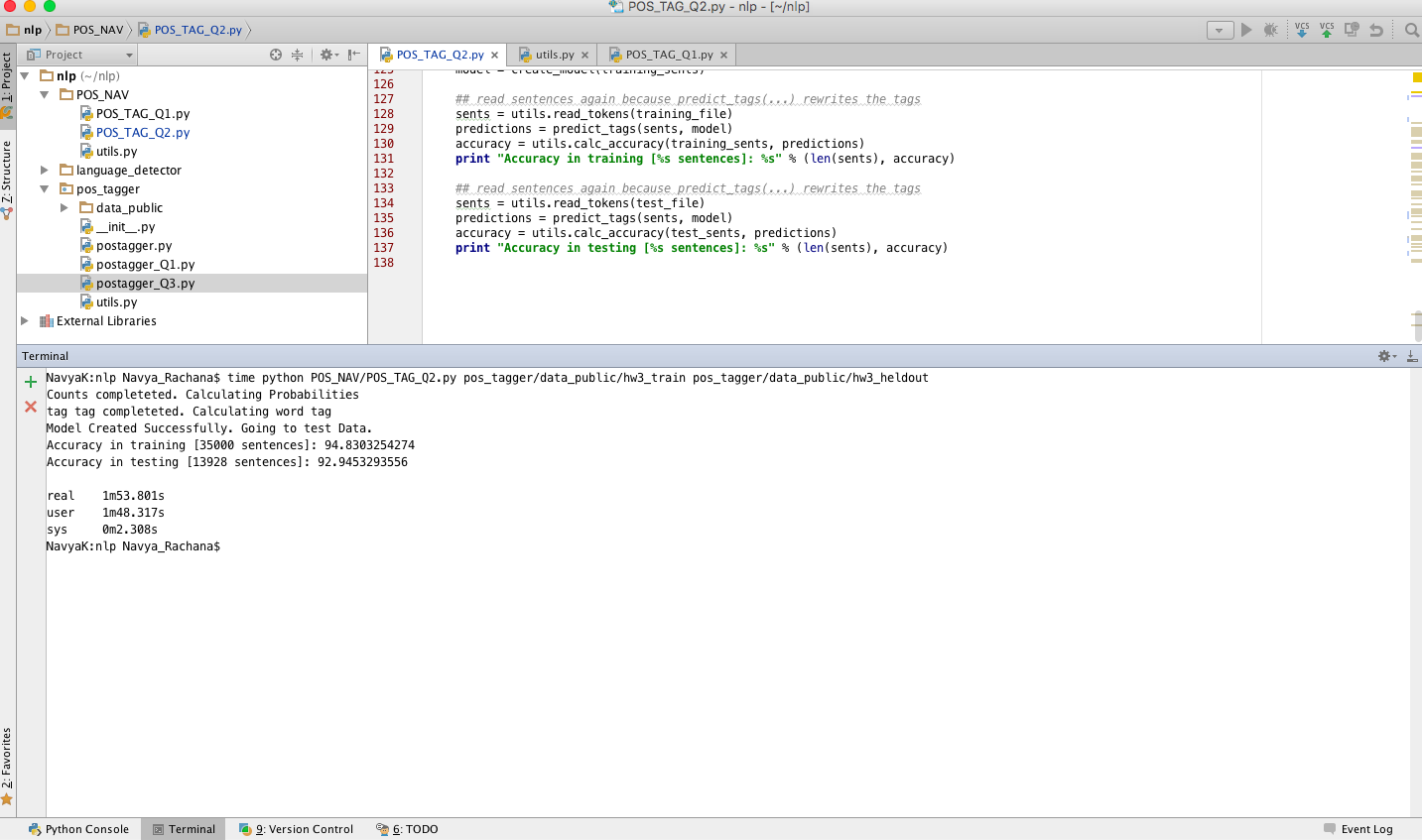
* Calculating the dictionaries for all the counts and probabilities
* Smoothing when probabilities aren’t found
* Storing a dictionary of dictionaries in model
* Accessing the dictionaries passed in model from creating model function
* Logic for the Viterbi Calculation

**Extra credit Unseen words:**

For unseen words Regular Expression POS tagger from nltk is implemented with the following patterns.

patterns = [  
 (**r'.\*ing$'**, **'VBG'**),  
 (**r'.\*ed$'**, **'VBD'**),  
 (**r'.\*es$'**, **'VBZ'**),  
 (**r'.\*ould$'**, **'MD'**),  
 (**r'.\*\'s$'**, **'NN$'**),  
 (**r'.\*s$'**, **'NNS'**),  
 (**r'^-?[0-9]+(.[0-9]+)?$'**, **'CD'**), *# cardinal num.* (**r'.\*'**, **'NN'**) *# nouns (default)*]

**Results: (LOWERCASING)**

****

Accuracy in training [35000 sentences]: 94.8303254274

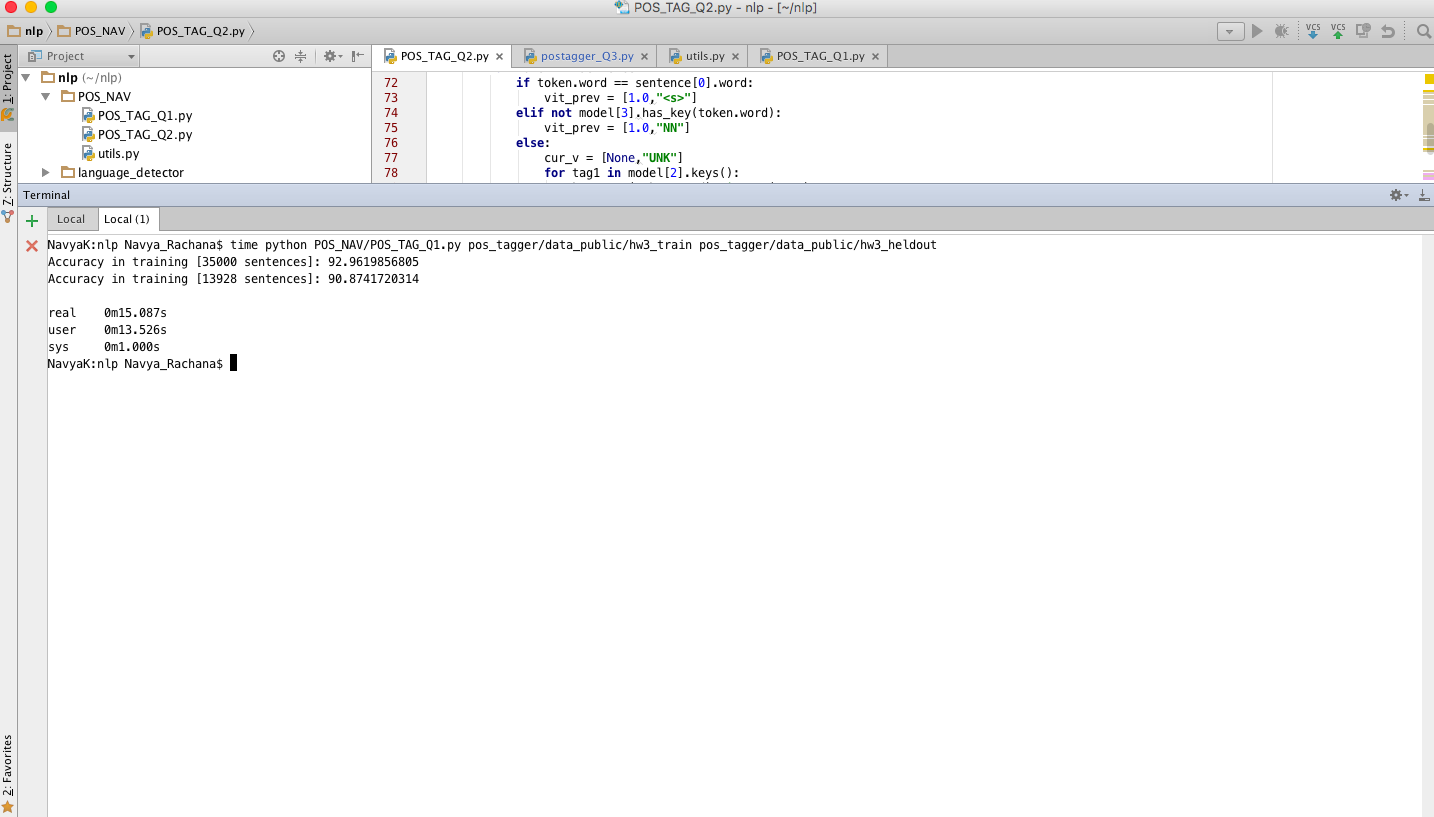
Accuracy in testing [13928 sentences]: 92.9453293556

real 1m53.801s

user 1m48.317s

sys 0m2.308s

**Results: (NO LOWERCASING)**

****

Accuracy in training [35000 sentences]: 96.3441454525

Accuracy in testing [13928 sentences]: 94.2741499328

real 1m33.712s

user 1m32.218s

sys 0m0.990s

**Observation:**

1. Lowercasing the words did not help in improving the performance in fact the performance decreased in both cases.