El Belton

LIN6209 CODING PROJECT PROPOSAL

Most likely next word coding function Yes. Decent project proposal

This Python coding project will aim to produce an output that returns the next most likely word following the input string, which will be a single word. An example purpose of the project is to find common reporting words used in media in regards to political speakers; for instance, if the input string was 'Johnson', the output may return 'claims'.As such, text data can be freely accessed and used from news websites, such as the British Broadcasting Corporation, in order to test the code.

all media or specific dictionaries for ~~pa~~ each ~~a~~media outlet.

(5) The basic functionality of the code will include an all-encompassing function that can be defined as something similar to 'next_word(word1)', with 'word1' being a single-word string. The function would read the text file contents into a single string, and then be split into lines so that each word is on its own line. The function would then look for 'word1' within the file and, upon identifying it, continue onto the following line and adding its contents to a dictionary with a value of +=1 depending on whether the dictionary already contains that string or not. When all instances of 'word1' have been identified and the above steps completed, the key in the dictionary with the highest value would be instructed to be returned, thus 'predicting' the most likely next word following 'word1'.

Testing
Make your own test data! even easier

As specified above, the code can be tested by using text files from the BBC and keywords such as names in order to see whether the correct string would be returned by manually reading through the article and comparing the result to the code's output. Additionally, the assert() function can be used in Python to directly find if the output of the function or value types matches what is expected. As such, the software that will be used to both construct and test this coding project will be Python IDLE.

(1) You will need a good amount of text ~~to~~ for your nex-word() function to train on.

(2) You could store the result in a database (just a suitable text ~~fo~~ file)

(3) If you can rate different text sources as 'easy to predict' or 'difficult to predict' that would be interesting.

(4) Be sure to get something working quickly as that will provide you with valuable feedback.

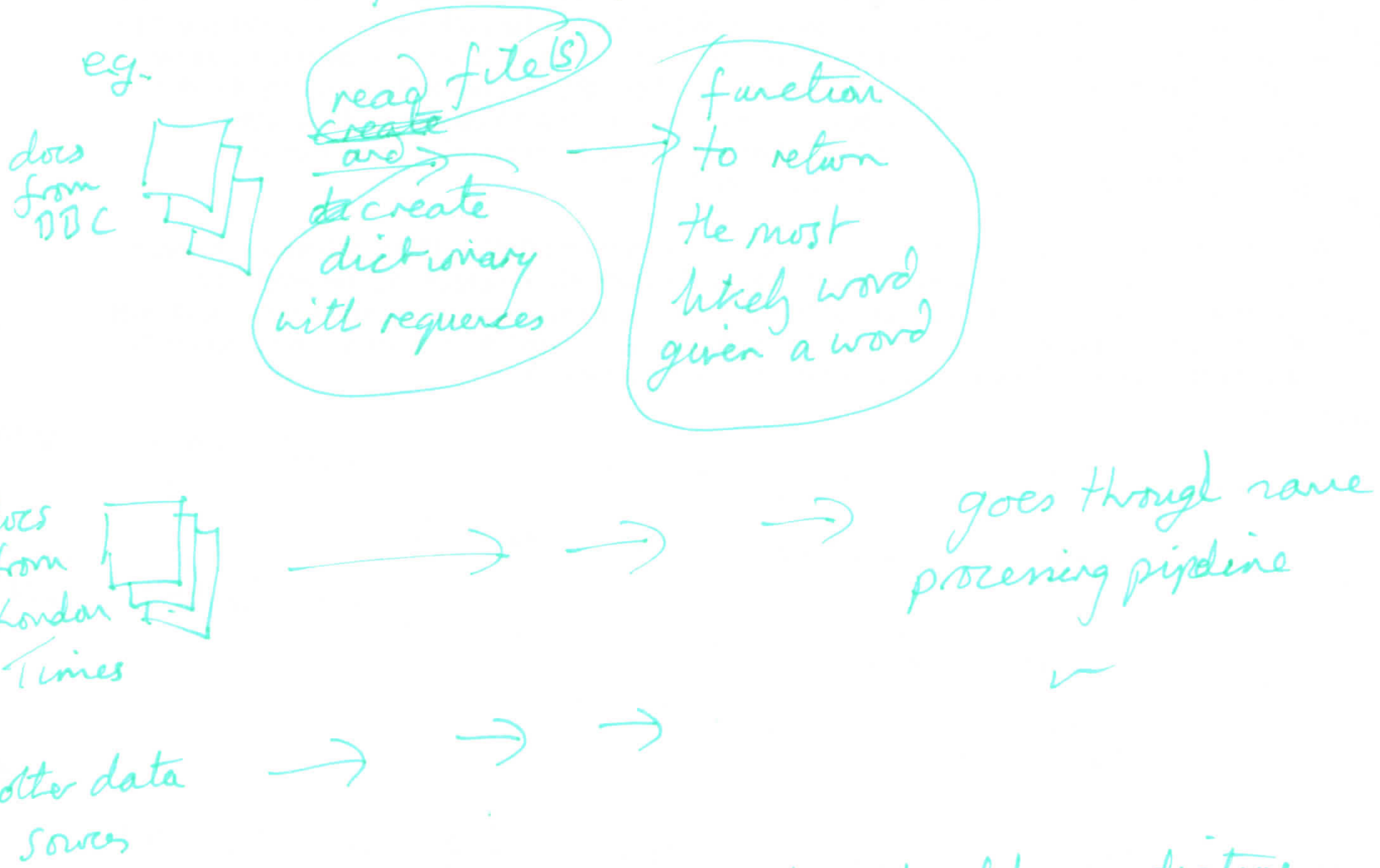
(5) This seems confused, at least to me.
You need a number of functions - to read file data into dictionary, another to store dictionary as a file perhaps.

(6) Will there be one dictionary ~~of~~ or one for each

data source?

Overall, I suggest you think through how your application will work, the sequence of tasks, and hence the functions you need.

e.g.



docs from BBC
read file(s)
~~create~~
and
~~da~~ create dictionary with sequences
function to return the most likely word given a word
docs from London Times
goes through same processing pipeline
✓
other data sources

Thoughts: some words will be highly predictive. Would be interesting to produce a list of them. Is the list the same across different data sources? etc.

Get the basic functions you need specified and built asap and then you will be in a good place

[illegible]