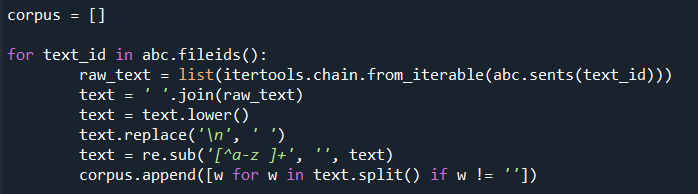
**Problem\_1**

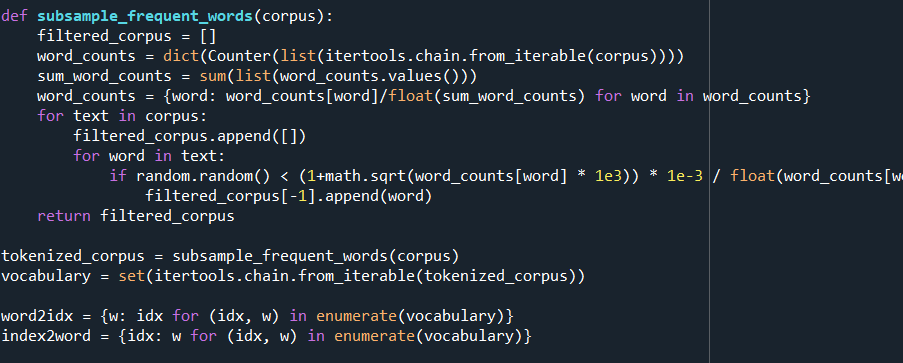
Word2Vec is a two-layer neural networks which is trained to get word vectors from words to find and establish relation between them. It takes as its input a large corpus of words and produces a vector space, with each unique word corresponding to a unique vector in the space.  
Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space.  
Word2Vec is a particularly efficient model for getting word embeddings from raw text.

To implement word2vec we need

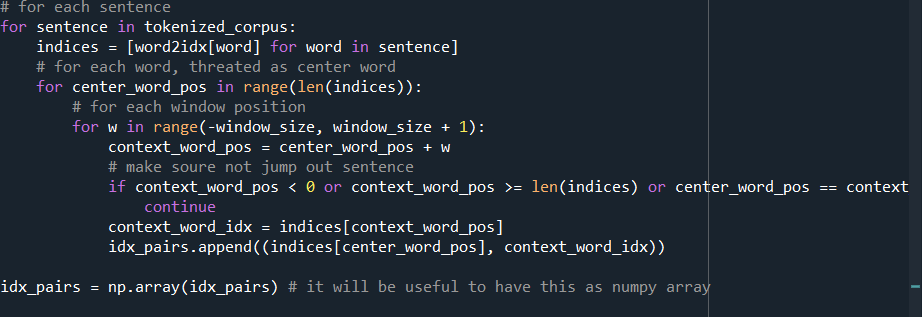
1. A corpus



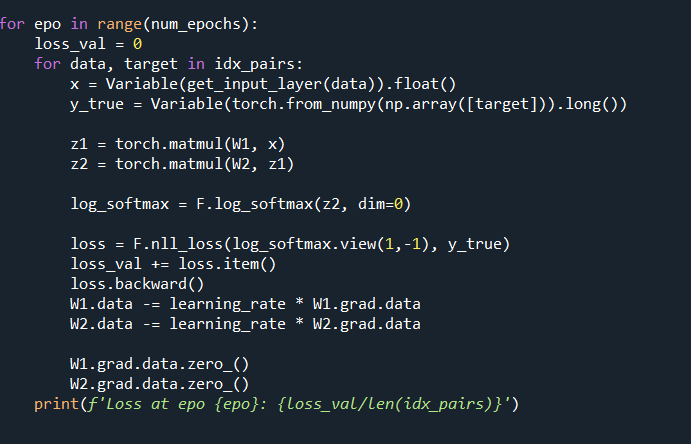
1. Getting a list of unique words in the corpus and tokenizing the words



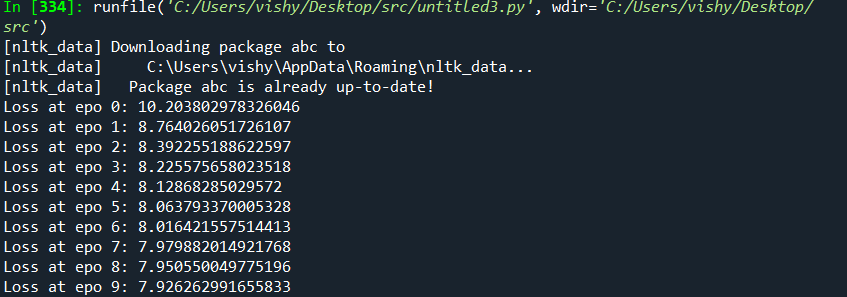
1. Then to get a relation between words we select a neighbour size and find different pairs of words



1. For each existing center, context pair in corpus we’re computing their “similarity score”.
2. We can instead have a probability for each word. To get that, we can replace the softmax out output by a sigmoid, taking values between 0 and 1.
3. Once the network trained, we can use the word embedding and compute the similarity between words. The following function computes the top n closest words for a given word. The similarity used is the cosine.



We run it for 15 epochs and our loss decreases

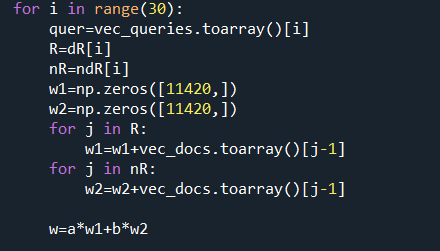


Problem\_2

1. Retrieval with Pseudo feedback

It was developed using the VSM as its basis. Therefore, we represent documents as points in a high-dimensional term space

We take the ground truth as input and see which is document is relevant for the given query and which is not relevant and accordingly optimise our queries



1. Retrieval with Pseudo feedback and query extension

In query extension after the query is modified once similar words are added to it to improve performance using wordnet