Assignment 1. Text similarity and Agglomerative Document Clustering.

Learning outcomes:
1. Read texts from file and splitting them to the words.
2. Transform texts into vector spaces, calculate distances in these spaces
3. Bag of words and TF/IDF vectorizer.

Task 1.

Please download text file. There we have 22 sentence related to the «cat» topic.
- Cat (animal)
- UNIX-utility cat for to display the contents of files
- versions of the OS X operating system named after the feline family

Your task is to find two sentences that are closest in meaning to the first sentence in document («In comparison to dogs, cats have not undergone …….») We will use the cosine distance as a measure of proximity.

Steps:

1. Open the file.
2. Each line is the one sentence. Please make them all in lower case form using string function *lower(). EXAMPLE: in comparison to dogs, cats have not undergone major changes during the domestication process.*
3. Tokenization. Means that splitting the sentences to the words. *For that purpose you can* use regular expressions, that can split the words by space or any other symbols that aren't letters. *re.split('[^a-z]', t).* Do not forgot to remove empty words. *EXAMPLE: ['in', 'comparison', 'to', 'dogs', '', 'cats', 'have', 'not', 'undergone', 'major', 'changes', 'during', 'the', 'domestication', 'process']*
4. Make a list of all the words that appear in the sentences. Note: all the words are unique. And give the index to the each sentence index from 0 to #of_the_unique_words. You can use *dict. Example: {0: 'mac', 1: 'permanently',2: 'osx',3: 'download',4: 'between',5: 'based', 6: 'which', ……., 252: 'safer', 253: 'will'}.* Hint: we have 254 unique words.
5. And create Matrix with N x D dimensions. N is the number of the sentences and D is the number of the unique words (22 x 254). Fill it in: the element with index (i, j) in this matrix must be equal to the number of occurrences of the j-th word in the i-th sentence. (**bag of words**) Example of bag of words:

Document	the	cat	sat	in	hat	with
the cat sat	1	1	1	0	0	0
the cat sat in the hat	2	1	1	1	1	0
the cat with the hat	2	1	0	0	1	1

6. Find the cosine distance from first sentence to the all other sentences. Which two sentences is close to the first sentence. You can use *scipy.spatial.distance.cosine*

Of course, during the Task 1 we implanted very simple method. For example int this method «cat» and «cats» two different words, but the meaning is the same.

Task 2.

For the second Task please do the same step from Task 1 (steps 1- 4).

In this task you will create Term Frequency — Inverse Document Frequency matrix. Find the cosine distance from first sentence to the all other sentences. Which two sentences is close to the first sentence. You can use *scipy.spatial.distance.cosine. Is the any difference from the result of the previous Task? Note: You should not to use any existing libraries for tdf/idf. All the steps similar to the previous example.*

Task 3.

Please run the Hierarchy Clustering algorithm for the Task 1 and Task 2. And plot the dendrogram. Please explain your results. NOTE: by default *scipy.cluster.hierarchy it* uses *euclidean distance. You should change it to the cosine distance. Example from lecture:*

```
import scipy.cluster.hierarchy as model
dend_max = model.dendrogram(model.linkage(data[['X', 'Y']], method='complete', metric="cosine"),  labels=data.index)
```