Exercise 1) Distance Functions:

Which of the following distance functions are metrics? Provide a proof for your claim. Note that distance = 1 – similarity

a. Cosine similarity
b. Jaccard similarity
c. max(x, y) = the larger of x and y.
d. sum(x,y) = x+y.

Solution: See chapter 3.5 of the MMDS textbook (http://infolab.stanford.edu/~ullman/mmds/book.pdf) for this

Exercise 2) Reservoir sampling

Prove that reservoir sampling guarantees that when there are n elements and reservoir size r each element is kept with the probability r/n.

Proof:

Given the reservoir size of r, the first r elements will be chosen with the probability 1 I,e r/r and in this case r = n. However, when r+1th element comes this item will be included in the rervoir with probability r/(r+1), similarly, r/(r+2). Therefore, for any nth element arriving in the stream, it will be kept with the probability r/n.

Exercise 3)

Consider a reservoir sampling process with k units of memory for a stream of data elements. Which of the following statements are true?

1. Each k-subset of the data stream is equally likely to be chosen as the sample
2. Probability that an ith element in the stream replaces an existing item in the reservoir is 1/i
3. Reservoir sampling is done with replacement
4. The k sample elements are the true random samples at any point in the stream
5. The ith element has a higher probability of being included in the sample than jth element provided i < j, (ith element appears before j th)

Options 1 and 4 true.

1. It is easy to verify this with a simple example with 4 items arriving with reservoir size of 1. We can construct a probability tree when each item arrives we either keep it or reject it. And we can compute the probability of keeping an item in the end at the leaves of the tree. You can see that proability of keeping each item is equal. Even though the i keeps increasing and the probability of keeping an item which arrives

later in the stream is low, it is also incredibly difficult for an existing item to remain the reservoir, which balances the probabilities.

4 The k sample elements are the true random samples at any point in the stream is also true because it follows from the above proof that the current k sample is chosen with the same probability as any other k subsets in the stream seen so far.

Exercise 4)

Consider a reservoir sampling process with 10 units of memory for a stream of data elements. What is the probability that an 100th element is not included in the sample?

Probability that ith item is included in the sample is r/i. which is 10/100 = 0.1.
Probability that it is not included is 1 – 0.1 = 0.9

Exercise 5) Document similarity

Take the following three text examples:

1. "Unlike classification or prediction, which analyzes data objects with class labels, clustering analyzes data objects without consulting a known class label. The class labels are not in the data because they are not known."

2. "Classification can be used for prediction of class labels of data objects. However, in many applications, prediction of missing or not known data values rather than class labels is performed to fit data objects into a schema."

3. "Sun Salutation, a ritual performed in the early morning, combines seven different postures. The sun, the life generator, is invoked by this Yogic exercise, an easy way to keep fit."

Solution:

Construct vectors based on the frequency of each word, ignoring the following "stop words given below. Use the cosine similarity function from the lecture to determine their mutual similarities:

Stopwords = { a, an, are, be, because, by, can, for, however, in, into, is, keep, many, not, of, or, rather, than, the, they, this, to, unlike, used, way, which, with, without }

So the first step is to clean the text by removing stop words and we should also throw away full stops "." And commas ",". It is also better to convert them all to lower case as "Classification" and "classification" are the same. Then we are left with following words and term frequencies (tf).

1. {'clustering': 1, 'consulting': 1, 'classification': 1, 'labels': 2, 'prediction': 1, 'analyzes': 2, 'objects': 2, 'known': 2, 'label': 1, 'data': 3, 'class': 3}
2. {'fit': 1, 'classification': 1, 'missing': 1, 'labels': 2, 'prediction': 2, 'applications': 1, 'objects': 2, 'values': 1, 'performed': 1, 'known': 1, 'data': 3, 'class': 2, 'schema': 1}
3. {'exercise': 1, 'combines': 1, 'invoked': 1, 'fit': 1, 'generator': 1, 'sun': 2, 'ritual': 1, 'life': 1, 'seven': 1, 'morning': 1, 'early': 1, 'postures': 1, 'performed': 1, 'easy': 1, 'salutation': 1, 'yogic': 1, 'different': 1}

To compute a vector we need to find a list of unique words, which are here.

['consulting', 'classification', 'labels', 'prediction', 'performed', 'salutation', 'yogic', 'clustering', 'combines', 'invoked', 'fit', 'sun', 'label', 'easy', 'exercise', 'schema', 'life', 'missing', 'different', 'analyzes', 'early', 'applications', 'objects', 'known', 'data', 'class', 'generator', 'ritual', 'seven', 'morning', 'postures', 'values']

We can construct the vectors depending on if each word is in the text or not. If the text contains a word put the tf value in the vector otherwise 0 should be added. This gives us following three vectors of 32 dimensions each:

1. [1, 1, 2, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, 0, 2, 2, 3, 3, 0, 0, 0, 0, 0, 0]
2. [0, 1, 2, 2, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 2, 1, 3, 2, 0, 0, 0, 0, 0, 1]
3. [0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 2, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0]

The cosine similarity values for each text segments is below:

	1	2	3
1	1	0,78049254	0
2	0,78049254	1	0,07784989
3	0	0.07784989	1

Note that this is similarity value therefore 1.0 is the highest and 0 is the lowest. If you need distance value you need to compute 1 – cosinesim() value