Problem Set #5

ECE 4424 / CS 4824 - Machine learning
Virginia Tech

November 21, 2020

- Feel free to collaborate with other classmates in doing the homework. Please indicate your collaborators with their student ID. You should, however, write down your solution yourself. Please try to keep the answers brief and clear.
- Whenever you need clarification, please post the related questions on Piazza under the corresponding homework folder.
- Total: 100 points
- **Due date: 12/5/2020, 11:59PM ET**
- Late submission: each student will have a total of **four** free late (calendar) days to use for homeworks. **Note: this is the total number of late days accumulated through all the homeworks so far. It's NOT reset after each homework!** Once these late days are exhausted, any assignments turned in late will be penalized 20% per late day. However, no assignment will be accepted more than three days after its due date. Each 24 hours or part thereof that a homework is late uses up one full late day.

1 K-Means algorithm (30 pts)

Given the dataset which contains points $x_1 = (1, 2)$, $x_2 = (2, 2)$, $x_3 = (2, 1)$, $x_4 = (-1, 5)$, $x_5 = (-2, -1)$, $x_6 = (-1, -1)$. Suppose we want to have 2 clusters. Answer the following questions:

a) (15 pts) Initialize the clusters cluster by $\mu_1 = x_1$ and $\mu_2 = x_4$, run the K-means clustering algorithm and report the final clusters (in terms of the points in each cluster and the cluster centers). Use L1 distance as the distance between points which is given by

$$d(x, y) = \sum_{j=1}^{d} |x^{(j)} - y^{(j)}|$$

where $x^{(j)}$ is the $j-$th entry of $x \in \mathbb{R}^d$.

b) (5 pts) Draw the points on a 2-D grid and check if the clusters make sense.

c) (10 pts) Can you find an initialization that results in a different clustering?

2 Programming assignment (70 pts)

For the following programming assignment, please download the datasets and iPython notebooks from Canvas and submit the following:

- Completed and ready-to-run iPython notebooks. Note: we will inspect the code and run your notebook if needed. If we cannot run any section of your notebook, you will not receive any points for the task related to that section.
- Responses (texts, codes, and/or figures) to the following problems/tasks

In this programming assignment, we will experiment with distributed representations of words. We'll also see how such an embedding can be constructed by applying principal component analysis to a suitably transformed matrix of word co-occurrence probabilities.

Task P1 (5 pts): Complete the following code to get a list of words and their counts. Report how many times does the word "evidence" and "investigation" appears in the corpus.

Task P2 (10 pts): Decide on the vocabulary. There are two potentially distinct vocabularies: the words for which we will obtain embeddings ('vocab_words') and the words we will consider when looking at context information ('context_words'). We will take the former to be all words that occur at least 20 times, and the latter to be all words that occur at least 100 times. We will stick to these choices for this assignment, but feel free to play around with them and find something better. Also, report the sizes of these two word lists.

Task P3 (10 pts): Get co-occurrence counts. These are defined as follows, for a small constant 'window_size=2':

- Let 'w0' be any word in 'vocab_words' and 'w' any word in 'context_words'.
- Each time 'w0' occurs in the corpus, look at the window of 'window_size' words before and after it. If 'w' appears in this window, we say it appears in the context of (this particular occurrence of) 'w0'.
- Define 'counts[w0][w]' as the total number of times 'w' occurs in the context of 'w0'.

Complete the function 'get_counts', which computes the 'counts' array, and returns it as a dictionary (of dictionaries). Find how many times the word "fact" appears in the context of "evidence" with window_size=2.

Task P4 (10 pts): Define 'probs[w0][]' to be the distribution over the context of 'w0', that is:

probs[w0][w] = counts[w0][w] / (sum of all counts[w0][])

Finish the function 'get_co_occurrence_dictionary' that computes 'probs'. Find the probability that the word "fact" appears in the context of "evidence".

Task P5 (10 pts): Based on the various pieces of information above, we compute the pointwise mutual information matrix (PMI):

$$\text{PMI}[i, j] = \max\left(0, \log \frac{\text{probs[i-th vocab word][j-th context word]}}{\text{context\_frequency[j-th context word]}}\right)$$

Complete the code to compute PMI for every word i and context word j. Report the output of the code.

Task P6 (10 pts): Implement the following function that finds the nearest neighbor of a given word in the embedded space. Note down the answers to the following queries.

Task P7 (15 pts): Implement the function that aims to solve the analogy problem:

A is to B as C is to ?

For example, A=King, B=Queen, C=man, and the answer for ? should be ideally woman (you will see that this may not be the case using the distributed representation).

Finds the K-nearest neighbor of a given word in the embedded space. Note: instead of outputting only the nearest neighbor, you should find the K=10 nearest neighbors and see whether there is one in the list that makes sense. You should also exclude the words C in the output list.

Also report another set A, B, C and the corresponding answer output by your problem. See if it makes sense to you.