INLS 642 - Assignment 1

Your name: Dev Gandhi

Question 1: Word Association (30 points)

Implement a program that finds the top 100-word associations in a given collection of Amazon reviews measured by pointwise mutual information we introduced in class. You may want to (1) remove some punctuations (e.g., comma, quotation marks, and so on); (2) consider only those ordered word pairs that appear within a fixed size of text windows (e.g., window size = 5 consecutive words) for at least a certain number of times (e.g., 50 times). You might want to start by taking the basic statistics, such as the frequency of each word (number of reviews that a word appeared in) and the frequency of each word pair (number of reviews a pair of consecutive words appeared in). Examine the word associations you've found. Do they all make sense? Can you see something interesting or undecipherable?

ANSWER - The top word associations found using pointwise mutual information (PMI) in Amazon reviews reveal a mix of common phrases, brand-related terms, and sentiment-driven expressions. Many associations, such as "tech support," "customer service," "page turner," and "buyer beware," make sense in the context of product reviews, reflecting common customer concerns and experiences. Several word pairs indicate strong brand relationships, including "Harry Potter," "Windows XP," and "Blu-ray," which are linked to specific products. Sentiment-related associations like "highly recommend" (positive), "piece junk" (negative), "rip off" (negative), and "fell apart" (negative) suggest how people express praise or dissatisfaction. Some unexpected findings, such as "blah blah" and "e.g.," appear to stem from review structures rather than product evaluation. Additionally, time-related mentions like "15 minutes" and "30 minutes" could refer to installation, delivery, or usage duration. Overall, the results show that PMI effectively identifies meaningful word relationships in customer reviews, with most associations aligning well with real-world product feedback, while a few require further interpretation.

Question 2: Feature Selection (30 points)

Implement a program that finds 100 single words (so-called “unigrams”) that are most associated with sentiment labels (label = 1: positive; label = 0: negative) in the given collection of Amazon reviews using Chi-square (χ^2) we introduced in class. Examine the words you’ve found. For each of these words, can you tell which sentiment (positive or negative) this word is strongly associated with? Do you see “mysterious” words that do not clearly associate with positive or negative sentiment?

ANSWER - The results for feature selection using Chi-square (χ^2) analysis indicate the most sentiment-associated words in Amazon reviews. Strongly positive words include "great" (2259.2), "love" (769.5), "best" (757.5), "excellent" (646.8), "wonderful" (442.9), "amazing" (332.9), "awesome" (284.8), "fantastic" (224.6), and "beautiful" (218.4), which are frequently found in positive reviews. Conversely, negative sentiment words such as "waste" (1300.8), "poor" (668.0), "worst" (663.6), "disappointed" (636.8), "bad" (592.3), "terrible" (472.0), "boring" (450.7), "horrible" (395.1), "awful" (329.0), "junk" (290.3), "useless" (274.0), "garbage" (240.6), and "refund" (201.2) clearly indicate dissatisfaction. Some words like "was" (451.9), "but" (205.4), "even" (194.1), "product" (189.6), and "had" (154.2) appear in both positive and negative reviews, making their sentiment association less clear. Additionally, transactional or cautionary words like "return" (344.7), "beware" (177.3), "broke" (166.0), "cheap" (165.5), "defective" (135.7), and "avoid" (138.0) are strongly associated with negative reviews, likely indicating customer dissatisfaction with product quality or service. Overall, the Chi-square method effectively highlights the words most indicative of sentiment, with clear positive and negative distinctions, though some ambiguous words exist.

Question 3: Spell Correction (40 points)

Spelling correction is a common functionality provided by most search engines. The basic idea can be simplified as matching an out-of-vocabulary string to a word in vocabulary that is the closest in spelling (for example, "carrolina" -> " carolina"). In other words, we need a function that measures the similarity between two strings. Of course, for those who knew it, a natural measure of string similarity/distance is the Levenshtein Edit Distance (http://en.wikipedia.org/wiki/Levenshtein_distance). You can find lots of implementations at http://en.wikibooks.org/wiki/Algorithm_Implementation/Strings/Levenshtein_distance. The problem in practice, however, is that the computation of edit distance is very costly, especially for long strings. Computationally it takes O(m*n) to find the edit distance between two strings with m and n letters (which can be pretty ugly when m and n are large). Spelling something like "*Parastratiosphecomyia sphecomyioides*" correctly is a challenge, and figuring out "*Parastratioschecomia*" responds to "*Parastratiosphecomyia*" isn't easier.

Alternatively, if we make use of a smart representation of a word/string, we can potentially reduce the time complexity with an approximation. One such approximation is to break a long string into **overlapping tri-grams**. That means you can represent a string with a **set** of tri-grams. That says, "*parastratioschecomia*" becomes {par, ara, ast, str, tra, rat, ati, tio, ios, osc, sch, che, hec, eco, com, omi, mia}, and "*Parastratiosphecomyia*" becomes {par, ara, ast, str, tra, rat, ati, tio, ios, osp, sph, phe, hec, eco, com, omy, myi, yia}. If we quickly compare the two sets, we can see they are highly similar, high enough to draw the conclusion that the two original strings are similar. Even sweeter, it takes a time complexity of O(m+n) to compute the similarity of two sets, instead of O(m*n).

You are provided with a dictionary (all words starting with "a" in wiktionary). Please implement a function so that for any input string, you can return the top 10 words in the dictionary that are the most similar to it. Use the Jaccard similarity that we introduced in class. Include the results for the following 5 strings:

abreviation

abstrictiveness

accanthopterigious

artifitial inteligwnse

agglumetation

Compare the results of your implementation with the results generated by the edit distance (you may use an existing implementation or package). Are you getting a good approximation?

One way to further tune the itemset representation of strings is to vary the length of the "*n*-grams" in your set. As you might already have guessed, an *n*-gram means *n* consecutive letters in a string. Try to use overlapping bigrams (2-grams), 4-grams, and 5-grams instead of tri-grams, and compare the results with the results using tri-grams. Which length of *n*-gram seems to work best for this task?

ANSWER - The spell correction analysis using Jaccard similarity with varying n-grams and Edit Distance demonstrates how well different methods approximate correct spellings. For each input word, n-gram-based Jaccard similarity (n = 2, 3, 4, 5) consistently ranked the correct word at the top, with some minor variations in the additional suggested words. The Edit Distance method also performed well, generally ranking the correct word first but sometimes introducing unrelated words with similar spelling structures.

For "abreviation", the correct spelling "abbreviation" was ranked first across all methods, showing strong agreement between Jaccard and Edit Distance. The same trend appeared for "artifitial inteligwnse", where "artificial intelligence" was correctly identified. However, "abstrictiveness" and "agglumetation" showed some inconsistencies, with Jaccard suggesting words that were closer in structure but not necessarily semantically correct. Notably, Edit Distance performed slightly better for complex words like "agglumetation," ranking "agglomeration" at the top, which is the correct word.

A key observation is that lower n-grams (e.g., bigrams) tend to retrieve more exact matches, whereas higher n-grams (4-grams and 5-grams) sometimes introduce more distant variations. Jaccard similarity with bigrams and trigrams seems to work best for approximate matches, while Edit Distance is more reliable for retrieving the closest valid word. Overall, a combination of these techniques provides the most robust spell correction system, balancing computational efficiency and accuracy.