CS620 HW4
@averm004@odu.edu

Homework-4 CS620

1.Assume you index the following terms "Mars", "planet", “Perseverance”, “Earth” and "rover”, given the document collection of 1525 documents and the document frequencies of index terms, Mars (42), planet (149), Perseverance (24), Earth (216), and rover (85). Also, assume that you will be applying only stemmers (without considering semantic similarity) during preprocessing

> Document d = “The Mars 2020 Perseverance Rover will search for signs of ancient microbial life, which will advance NASA's quest to explore the past habitability of Mars. The rover has a drill to collect core samples of Martian rock and soil, then store them in sealed tubes for pickup by a future mission that would ferry them back to Earth for detailed analysis. Perseverance will also test technologies to help pave the way for future human exploration of Mars. Strapped to the rover's belly for the journey to Mars is a technology demonstration — the Mars Helicopter, Ingenuity, may achieve a "Wright Brothers moment “by testing the first powered flight on the Red Planet. There are several ways that the mission helps pave the way for future human expeditions to Mars and demonstrates technologies that may be used in those endeavors. These include testing a method for producing oxygen from the Martian atmosphere, identifying other resources (such as subsurface water), improving landing techniques, and characterizing weather, dust, and other potential environmental conditions that could affect future astronauts living and working on Mars.”

a. Using this information, calculate the tfidf values for the five terms (Mars, planet, perseverance, earth, rover). Ignore the stop words. Use log2 for idf calculation and maximum normalization for term frequencies.
b. For the query A= “Mars rover from Earth”, calculate the tfidf values for the query terms.
c. For the query B= “Mars rover Helicopter, Ingenuity”, calculate the tfidf values for the query terms.
d. calculates the Cosine similarity, cosine(d,A), and cosine(d,B) using the tfidf values calculated from parts a,b, and c.

Solution(a):

CS620 HW4
@averm004@odu.edu

TF-IDF values:

TF (Mars) = 7
IDF (Mars) = log2(1525 / 42) = 5.182
TF-IDF (Mars) =7 * 5.182 = 36.274

TF (planet) = 1
IDF (planet) = log2(1525 / 149) = 3.355
TF-IDF (planet) = 1 *3.355 = 3.355

TF (Perseverance) = 2
IDF (Perseverance) = log2(1525 / 24) = 5.989
TF-IDF (Perseverance) = 2 * 5.989 = 11.978

TF (Earth) = 1
IDF (Earth) = log2(1525 / 216) = 2.819
TF-IDF (Earth) = 1 * 2.819= 2.819

TF (rover) = 3
IDF (rover) = log2(1525 / 85) = 4.165
TF-IDF (rover) = 3 *4.165 = 12.495

Solution(b):

For the query "Mars rover from Earth", we calculate the term frequencies for each term in the query and then multiply by the idf to get the tfidf values for each term in the query.

TF (Mars) = 1

IDF (Mars) = log2(1525 / 42) = 5.182

TF-IDF (Mars) = 1 * 5.182= = 5.182

TF (rover) = 1

IDF (rover) = log2(1525 / 85) = 4.165

TF-IDF (rover) = 1 * 4.165 = 4.165

TF (Earth) = 1

IDF (Earth) = log2(1525 / 216) =2.819
TF-IDF (Earth) = 1 * 2.819 = 2.819

Solution (c):

TF (Mars) = 1

IDF (Mars) = log2(1525 / 42) = 5.182

TF-IDF (Mars) = 1 * 5.182= = 5.182

TF (rover) = 1

IDF (rover) = log2(1525 / 85) = 4.165

TF-IDF (rover) = 1 * 4.165 = 4.165

TF (Helicopter) = 1

IDF (Helicopter) = log2(1525 / 0) = infinity (assume 0 for simplicity)

TF-IDF (Helicopter) = 0

TF (Ingenuity) = 1

IDF (Ingenuity) = log2(1525 / 0) = infinity (assume 0 for simplicity)

TF-IDF (Ingenuity) = 0

CS620 HW4
@averm004@odu.edu

Solution(d):

For query A:

cosine(d, A) = (TF-IDF(Mars) * TF-IDF(Mars) + TF-IDF(rover) * TF-IDF(rover) + TF-IDF(Earth) * TF-IDF(Earth)) / (sqrt(TF-IDF(Mars)^2 + TF-IDF(rover)^2 + TF-IDF(Earth)^2) * sqrt(TF-IDF(A)^2 + TF-IDF(rover)^2 + TF-IDF(Earth)^2))

- Cosine(d, A) =
36.274*5.182+4.165*12.495+2.819*2.819/sqrt((36.274^2+3.355^2+11.978^2+2.819^2+12.495^2)*(5.182^2+4.165^2+2.819^2))

= 247.960304/291.958630278

= **0.8492**

For query B:

cosine(d, B) = (TF-IDF(Mars) * TF-IDF(Mars) + TF-IDF(rover) * TF-IDF(rover)) / (sqrt(TF-IDF(Mars)^2 + TF-IDF(rover)^2) * sqrt(TF-IDF(Mars)^2 + TF-IDF(rover)^2 + TF-IDF(Helicopter)^2 + TF-IDF(Ingenuity)^2))

- Cosine(d, B) =
36.274*5.182+4.165*12.495/sqrt((36.274^2+3.355^2+11.978^2+2.819^2+12.495^2)*(5.182^2+4.165^2))

=240.013543/sqrt(1634.603371*44.200349)

= 240.013543/268.793674544

=**0.8929**

CS620 HW4
@averm004@odu.edu

2.Consider the transaction database in the table below:

Transaction ID	Items Bought
0001	{a, d, e}
0024	{a, b, c, e}
0012	{a, b, d, e}
0031	{a, c, d, e}
0015	{b, c, e}
0022	{b, d, e}
0029	{c, d}
0040	{a, b, c}
0033	{a, d, e}
0038	{a, b, e}

a) Compute the support for itemsets {a}, {b, c}, and {a,b,c} by treating each transaction ID as a market basket. You have 10 transactions

b) Use the results in part (a) to compute the confidence for the association rules {b, c} → {a} and {a} → {b, c}.

c) Using Apriori algorithm, generate frequent itemsets for min sup count =4. You need to show the complete calculations including intermediate candidate itemsets (Ci and Li where i is the level 1,2...etc.)

d) If you use the Brute-force approach to find all possible association rules, how many candidate itemsets exist for the given market basket? Explain your answer

e) How does the Apriori algorithm improve upon the brute-force algorithm?

f) We generally will be more interested in association rules with a high confidence threshold. However, often we will not be interested in association rules that have a confidence of 100%. Why? Explain your answer

CS620 HW4
@averm004@odu.edu

Solution(a):

Count the occurrences of each item or itemset:

{a} appears in 8 transactions.

{b, c} appears in 6 transactions.

{a, b, c} appears in 4 transactions.

Hence support for a,b and c are given below

Support for {a} = 8 / 10 = 0.8 or 80%.

Support for {b, c} = 6 / 10 = 0.6 or 60%.

Support for {a, b, c} = 4 / 10 = 0.4 or 40%.

Solution(b):

Confidence of {b, c} → {a} = support({a, b, c}) / support({b, c})

0.4/0.6 = 0.67 or 67%

Confidence of {a} → {b, c} = support({a, b, c}) / support({a})

0.4/0.8 =0.5 of 50%

Solution(c):

Iteration 1:

C1: {a}, {b}, {c}, {d}, {e}

L1 (with min sup count = 4): {a}, {b}, {c}, {d}, {e}

Iteration 2:

C2: {a, b}, {a, c}, {a, d}, {a, e}, {b, c}, {b, d}, {b, e}, {c, d}, {c, e}, {d, e}

L2: {a, b}, {a, c}, {a, d}, {a, e}, {b, c}, {b, d}, {b, e}, {c, d}, {c, e}, {d, e}

Iteration 3:

C3: {a, b, c}, {a, b, d}, {a, b, e}, {a, c, d}, {a, c, e}, {a, d, e}, {b, c, d}, {b, c, e}, {b, d, e}, {c, d, e}

L3 (with min sup count = 4): {a, b, c}, {a, c, e}

Solution(d):

The brute-force approach generates possible itemsets where n is the number of unique items. For the given market basket with 5 unique items, the number of possible itemsets would be 2^5 -1 =31.The Apriori algorithm reduces the number of generated itemsets by pruning the sets that don't meet the minimum support threshold, significantly reducing the total number of calculations.

Solution(e):

1. **Reduced Combinatorial Search:**

Brute-Force: Generates all possible itemsets, which grows exponentially with the number of items. This includes many infrequent or irrelevant itemsets.

Apriori: Uses a bottom-up approach to iteratively find and eliminate infrequent itemsets based on support thresholds, significantly reducing the search space. It focuses only on those itemsets that have the potential to be frequent.

2. Pruning Based on Apriori Principle:

Brute-Force: Exhaustively generates all itemsets regardless of their frequency, leading to a huge number of calculations.

Apriori: Employs the Apriori principle, which states that if an itemset is infrequent, its supersets (larger itemsets) will also be infrequent. Hence, it prunes infrequent itemsets, focusing only on those that have potential to be frequent.

3. Optimized Performance:

Brute-Force: Involves checking a vast number of infrequent and redundant itemsets, resulting in extensive computation and memory usage.

Apriori: Reduces computational load and memory usage by concentrating only on potentially frequent itemsets, enhancing computational efficiency.

4. Selective Itemset Generation:

Brute-Force: Generates all possible itemsets, leading to a significant number of calculations, regardless of their significance or frequency.

Apriori: Generates candidate itemset based on the frequency of individual items and uses this information to focus on potentially frequent item sets.

Solution(f): High confidence rules are often more interesting as they suggest stronger relationships between items. However, rules with a confidence of 100% might not be useful due to their infrequency. Such rules, while strong, might only be applicable to a very small number of transactions, limiting their practical application and generalizability to a larger dataset. Therefore, they might not provide enough insights or patterns for meaningful analysis.