Asmt 2: Document Similarity and Hashing

Turn in through **Gradescope** by February 1 at 2:45pm, then come to class:
100 points

Overview

In this assignment, you will explore the use of n-grams, Jaccard distance, min hashing, and LSH in the context of document similarity.

Note: Homework assignments are intended to help you learn the course material, and successfully solve mid-term and final exams that will be done on paper in person.

As usual, it is recommended that you use LaTeX for this assignment. If you do not, you may lose points if your assignment is difficult to read or hard to follow. The latex source for this homework is available on Canvas. I recommend that you drop the two latex files (.tex and .sty) in a new Overleaf project, and compile/edit the .tex file there.

1 Creating n-grams (50 points)

The space counts as a character in character n-grams. Do not do any text pre-processing.

A: (25 points) Using the code below and following the instruction to use scikit-learn, do the following:

(i) Construct 2-grams based on *characters*, for all documents. Do not share these n-grams.

(ii) Construct 3-grams based on *characters*, for all documents. Do not share these n-grams.

(iii) Construct 2-grams based on *words*, for all documents. Do not share these n-grams.

(iv) Answer: How many distinct n-grams are there for each document with each type of n-gram? You should report 4 (documents) $\times$ 3 (n-gram types) = 12 different numbers.

(v) Answer: Which parameters (and their values) and methods of `CountVectorizer` did you use?

```python
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer

def n_grams_sklearn():
    # Let's read some data
    url = "https://raw.githubusercontent.com/koaning/icepickle/main/datasets/imdb_subset.csv"
    df = pd.read_csv(url) # This is how you read a csv file to a pandas frame
    corpus = list(df['text'])
    corpus_small = corpus[:4] # This is a list of 4 movie reviews

    '''
    Read documentation for https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html
    Complete 1.A by using CountVectorizer, its methods, and adjusting certain parameters.
    '''

    # Your code should go here:
```

Listing 1: Code you should build your solution on.

Answer (iv):
Answer (v):

B: (25 points)

(i) Write a function that implements the formula for generalization of set similarities and share your implementation (i.e., code).

(ii) Using that function with appropriate parameters, compute and report the Jaccard similarity between all pairs of documents for each type of n-gram calculated above. Tip: Use a simple example like the one from the class to test your implementation. You should report 3 (n-gram types) × 6 (document pairs) = 18 different numbers.

Answer (i):
Answer (ii):

2 Min Hashing (50 points)

You will use two text documents for this assignment that are available on Canvas. We will consider a hash family $\mathcal{H}$ so that any hash function $h \in \mathcal{H}$ maps from $h : \{k\text{-grams}\} \to [m]$ for m large enough (To be extra cautious, I suggest over $m \geq 10{,}000$; but should work with smaller m too).

A: (35 points) Construct 3-grams based on *characters* for D1 and D2. The space counts as a character. Do not do any text pre-processing. Then build a min-hash signature using the fast min-hashing algorithm for document D1 and D2 using $t = \{20, 60, 150, 300, 600\}$ hash functions. For hash functions, you could use e.g. multiplicative hashing covered in the lecture "Statistical Principles" and to get t of them you just need to provide t different parameters a. For each value of t report the approximate Jaccard similarity between the pair of documents D1 and D2, estimating the Jaccard similarity:

$$\hat{\mathsf{JS}}_t(a,b) = \frac{1}{t}\sum_{i=1}^{t}\begin{cases}1 & \text{if } a_i = b_i\\ 0 & \text{if } a_i \neq b_i.\end{cases}$$

You should report 5 numbers.

Answer:

B: (15 point) What seems to be a good value for t? You may run more experiments. Justify your answer in terms of both accuracy and time.

Answer:

3 Bonus (3 points)

Describe a scheme like Min-Hashing over a domain of size n for the *Andberg* Similarity, defined $\mathsf{Andb}(A,B) = \frac{|A\cap B|}{|A\cup B| + |A \triangle B|}$. That is so given two sets A and B and family of hash functions, then $\mathbf{Pr}_{h\in\mathcal{H}}[h(A) = h(B)] = \mathsf{Andb}(A,B)$. Note the only randomness is in the choice of hash function h from the set $\mathcal{H}$, and $h \in \mathcal{H}$ represents the process of choosing a hash function (randomly) from $\mathcal{H}$. The point of this question is to design this process, and show that it has the required property.

Or show that such a process cannot be done.

Answer: