# IT350 Assignment-2

**Name: Durga Supriya HL**
**Roll no.: 201IT121**

**Dataset:** here
**Code:** here

**A: How many distinct shingles are there for each document with each type of shingle? (1 mark)**

- Initialize an empty dictionary named duplicates to keep track of duplicate shingles, and an empty list named shingles to store the unique shingles.
- It loops over the characters in the data string, up to the len(data) - kth character. For each iteration of the loop:
- It creates a substring of k characters starting at the current character using Python's string slicing syntax: data[i : i + k].
- If the current substring is not already in the duplicates dictionary, it appends it to the shingles list and adds it to the duplicates dictionary with a value of 1.
- Finally, it returns the shingles list.
- Then the length of the lists is printed

**Output:**

```
Number of distinct 8-shingles based on character for document 1:  5027
Number of distinct 8-shingles based on character for document 2:  3014
Number of distinct 8-shingles based on character for document 3:  3678
Number of distinct 8-shingles based on character for document 4:  4466
----------------------------------------------------------------------
```

==========

**B: Compute the Jaccard distance between all pairs of documents for each type of shingling. (1 mark)**

1. Creates a union list by using the set() function to create a set of all unique elements in shingle1 and shingle2, and then uses the | operator to perform set union and convert the result back to a list.
2. Creates an intersection list by using the set() function to create a set of all elements that are in both shingle1 and shingle2, and then uses the & operator to perform a set intersection and convert the result back to a list.
3. Calculate the Jaccard distance by subtracting the ratio of the length of the intersection to the length of the union from 1.
4. Finally, return the Jaccard distance as the result of the function.

The Jaccard distance is a measure of dissimilarity between two sets and is defined as the ratio of the size of the intersection of the sets to the size of the union of the sets. This function calculates the Jaccard distance between two shingle sets by first computing the union and intersection of the shingle sets as lists

**Output:**

```
Jaccard distance between document 1 and document 2 for 5-shingling based on characters:  0.8892112170189252
Jaccard distance between document 1 and document 3 for 5-shingling based on characters:  0.8822547508988187
Jaccard distance between document 1 and document 4 for 5-shingling based on characters:  0.8674540682414698
Jaccard distance between document 2 and document 3 for 5-shingling based on characters:  0.8820581356498497
Jaccard distance between document 2 and document 4 for 5-shingling based on characters:  0.8788968824940048
Jaccard distance between document 3 and document 4 for 5-shingling based on characters:  0.8810112668315471
-----------------------------------------------------------------------------------------------------
Jaccard distance between document 1 and document 2 for 8-shingling based on characters:  0.8892112170189252
Jaccard distance between document 1 and document 3 for 8-shingling based on characters:  0.8822547508988187
Jaccard distance between document 1 and document 4 for 8-shingling based on characters:  0.8674540682414698
Jaccard distance between document 2 and document 3 for 8-shingling based on characters:  0.8820581356498497
Jaccard distance between document 2 and document 4 for 8-shingling based on characters:  0.8788968824940048
Jaccard distance between document 3 and document 4 for 8-shingling based on characters:  0.8810112668315471
-----------------------------------------------------------------------------------------------------
Jaccard distance between document 1 and document 2 for 4-shingling based on words:  0.9991019308486754
Jaccard distance between document 1 and document 3 for 4-shingling based on words:  0.998766954377312
Jaccard distance between document 1 and document 4 for 4-shingling based on words:  1.0
Jaccard distance between document 2 and document 3 for 4-shingling based on words:  0.9977439368302312
Jaccard distance between document 2 and document 4 for 4-shingling based on words:  0.9985429820301117
Jaccard distance between document 3 and document 4 for 4-shingling based on words:  0.9995590828924162
```

==========

**C. Change to any Similarity Function (use any recent similarity distance) and check the distance. (2 marks)**

I used Levenshtein distance.
The Levenshtein distance is a measure of the difference between two strings, defined as the minimum number of single-character edits (i.e., insertions, deletions, or substitutions) required to transform one string into the other.
The Levenshtein distance between the two strings is then computed using the ratio() function of the lev module, which calculates the Levenshtein distance as the ratio of the number of insertions, deletions, and substitutions needed to transform one string into another, to the total number of characters in the two strings.
Finally, the Levenshtein distance as a float value between 0 and 1 is returned as the output of the function.

**Output:**

```
Levenshtein Distance between document 1 and document 2 :  0.38708656432979904
Levenshtein Distance between document 1 and document 3 :  0.40949598246895547
Levenshtein Distance between document 1 and document 4 :  0.43111256974072854
Levenshtein Distance between document 2 and document 3 :  0.43817920852983394
Levenshtein Distance between document 2 and document 4 :  0.4177360828391893
Levenshtein Distance between document 3 and document 4 :  0.43111256974072854
```

==========

**D. Try the above all for anyone Indian language Text or Code-Mixed Text (Hinglish, Kanglish, etc). (2 marks)**

The 4 files of Kannada are in the dataset folder mentioned at the beginning.
The same steps are mentioned in questions A,B, and C are repeated on these text files.

**Output:**

Number of distinct shingles

```
Number of distinct 5-shingles based on character for document 1:  6231
Number of distinct 5-shingles based on character for document 2:  4570
Number of distinct 5-shingles based on character for document 3:  6243
Number of distinct 5-shingles based on character for document 4:  4953
--------------------------------------------------------------------------
Number of distinct 8-shingles based on character for document 1:  8008
Number of distinct 8-shingles based on character for document 2:  5786
Number of distinct 8-shingles based on character for document 3:  8596
Number of distinct 8-shingles based on character for document 4:  6631
--------------------------------------------------------------------------
Number of distinct 4-shingles based on words for document 1:  1125
Number of distinct 4-shingles based on words for document 2:  751
Number of distinct 4-shingles based on words for document 3:  1227
Number of distinct 4-shingles based on words for document 4:  981
```

## Jaccard distance

```
Jaccard distance between document 1 and document 2 for 5-shingling based on characters:  0.9232379623168179
Jaccard distance between document 1 and document 3 for 5-shingling based on characters:  0.9214940342382847
Jaccard distance between document 1 and document 4 for 5-shingling based on characters:  0.9160690056212444
Jaccard distance between document 2 and document 3 for 5-shingling based on characters:  0.9163158949689316
Jaccard distance between document 2 and document 4 for 5-shingling based on characters:  0.9280729401170644
Jaccard distance between document 3 and document 4 for 5-shingling based on characters:  0.9319851187637127
-----------------------------------------------------------------------------------------------------------
Jaccard distance between document 1 and document 2 for 8-shingling based on characters:  0.9885613726352838
Jaccard distance between document 1 and document 3 for 8-shingling based on characters:  0.9902700072974945
Jaccard distance between document 1 and document 4 for 8-shingling based on characters:  0.9879709643968199
Jaccard distance between document 2 and document 3 for 8-shingling based on characters:  0.9883230163196398
Jaccard distance between document 2 and document 4 for 8-shingling based on characters:  0.9890083048363458
Jaccard distance between document 3 and document 4 for 8-shingling based on characters:  0.9918564618644068
-----------------------------------------------------------------------------------------------------------
Jaccard distance between document 1 and document 2 for 4-shingling based on words:  1.0
Jaccard distance between document 1 and document 3 for 4-shingling based on words:  1.0
Jaccard distance between document 1 and document 4 for 4-shingling based on words:  1.0
Jaccard distance between document 2 and document 3 for 4-shingling based on words:  1.0
Jaccard distance between document 2 and document 4 for 4-shingling based on words:  1.0
Jaccard distance between document 3 and document 4 for 4-shingling based on words:  1.0
```

==========

## E. Build a min hash signature for the above experiments. (2 marks)

To find the signature matrix for a given dataset, we use the min-hashing technique. The signature matrix is a compact representation of the dataset that can be used to efficiently estimate the Jaccard similarity between pairs of documents.

The steps to find the signature matrix for a given dataset are as follows:

- Generate a set of k-shingles for each document in the dataset. This we already did.
- Assign a unique integer identifier to each k-shingle in the entire dataset. This is done to speed up the computation of the minhash function.
- Generate a set of hash functions. The number of hash functions should be equal to the number of rows in the signature matrix.
- For each document in the dataset, compute the minhash signature by applying the set of hash functions to the set of k-shingles in the document. The minhash signature is a vector of integers, where each element represents the minimum hash value for each hash function across all k-shingles in the document.

Concatenate the minhash signatures of all the documents to form the signature matrix. The signature matrix has one row for each hash function and one column for each document.

**Output:**

```
Min hash signature of 5-shingle construct based on character of document 1:  [2, 2, 0, 0, 2, 6, 2, 0, 1, 2, 1, 0, 2, 11, 9, 1, 0, 3, 1,
Min hash signature of 5-shingle construct based on character of document 2:  [0, 6, 3, 0, 0, 3, 0, 1, 0, 1, 0, 3, 4, 4, 2, 2, 1, 2, 1,
Min hash signature of 5-shingle construct based on character of document 3:  [3, 3, 1, 0, 0, 6, 9, 6, 5, 7, 0, 0, 2, 4, 0, 1, 2, 0, 6,
Min hash signature of 5-shingle construct based on character of document 4:  [1, 1, 2, 0, 0, 0, 2, 1, 0, 0, 0, 3, 1, 1, 5, 1, 0, 0, 0,
----------------------------------------------------------------------------------------------------
Min hash signature of 8-shingle construct based on character of document 1:  [1, 2, 2, 9, 3, 0, 3, 1, 1, 4, 0, 7, 5, 0, 0, 1, 6, 0, 2,
Min hash signature of 8-shingle construct based on character of document 2:  [13, 1, 0, 2, 6, 0, 0, 6, 3, 6, 0, 2, 1, 9, 7, 4, 6, 1, 0
Min hash signature of 8-shingle construct based on character of document 3:  [0, 2, 1, 0, 0, 9, 5, 3, 1, 0, 2, 0, 5, 1, 0, 2, 3, 1, 2,
Min hash signature of 8-shingle construct based on character of document 4:  [2, 3, 0, 2, 1, 0, 11, 2, 1, 0, 0, 0, 0, 5, 0, 4, 8, 6, 3
----------------------------------------------------------------------------------------------------
Min hash signature of 4-shingle construct based on words of document 1:  [4, 6, 11, 2, 1, 2, 2, 5, 10, 27, 6, 11, 2, 3, 5, 1, 1, 15, 2
Min hash signature of 4-shingle construct based on words of document 2:  [29, 61, 14, 0, 11, 3, 13, 20, 14, 6, 6, 3, 1, 16, 10, 8, 1,
Min hash signature of 4-shingle construct based on words of document 3:  [7, 2, 14, 4, 9, 1, 4, 6, 2, 1, 1, 17, 0, 33, 3, 23, 2, 11, 1
Min hash signature of 4-shingle construct based on words of document 4:  [8, 1, 1, 10, 1, 0, 13, 4, 10, 19, 2, 11, 6, 1, 3, 42, 0, 4,
```

**F. Find the candidate pairs from the Signature Matrix and Plot the probabilities for different similarity and band values. Provide your conclusions for the entire experiment (2 Marks)**

The idea behind LSH is to use hash functions to reduce the number of comparisons needed to find similar documents. The documents are first divided into a set of small bands, and within each band, similar documents

are identified using a hash function. The hash function maps similar documents to the same bucket with high probability, which allows us to narrow down the search space for candidate pairs.

To identify candidate pairs using LSH, the process involves the following steps:

- Split the minhash signature matrix into a number of bands, each consisting of r rows.
- Hash the signature of each band into a hash table, where the hash table has b buckets.
- For each band, identify the candidate pairs by looking for documents that have the same hash value for the band in the hash table.
- For each candidate pair, calculate the Jaccard similarity using the full minhash signature.
- Return the candidate pairs that have a Jaccard similarity above a certain threshold.

By using LSH, we can significantly reduce the number of document pairs that need to be compared and speed up the process of finding similar documents in large datasets.

**Output:**

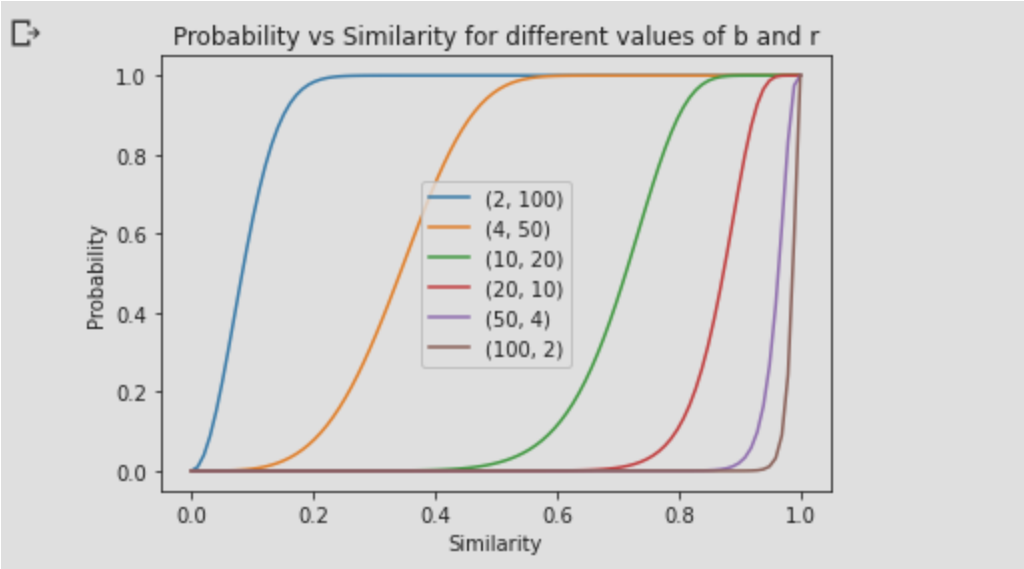
```
Candidate pairs for Signature Matrix 5 for b = 2 :
set()
Candidate pairs for Signature Matrix 5 for b = 4 :
set()
Candidate pairs for Signature Matrix 5 for b = 10 :
set()
Candidate pairs for Signature Matrix 5 for b = 20 :
set()
Candidate pairs for Signature Matrix 5 for b = 50 :
set()
Candidate pairs for Signature Matrix 5 for b = 100 :
{(0, 1), (1, 2), (0, 3), (2, 3), (0, 2), (1, 3)}
Candidate pairs for Signature Matrix 8 for b = 2 :
set()
Candidate pairs for Signature Matrix 8 for b = 4 :
set()
Candidate pairs for Signature Matrix 8 for b = 10 :
set()
Candidate pairs for Signature Matrix 8 for b = 20 :
set()
Candidate pairs for Signature Matrix 8 for b = 50 :
{(0, 1)}
Candidate pairs for Signature Matrix 8 for b = 100 :
{(0, 1), (1, 2), (0, 3), (2, 3), (0, 2), (1, 3)}
Candidate pairs for Signature Matrix 4 for b = 2 :
set()
Candidate pairs for Signature Matrix 4 for b = 4 :
set()
Candidate pairs for Signature Matrix 4 for b = 10 :
set()
Candidate pairs for Signature Matrix 4 for b = 20 :
set()
Candidate pairs for Signature Matrix 4 for b = 50 :
set()
Candidate pairs for Signature Matrix 4 for b = 100 :
{(0, 2)}
```

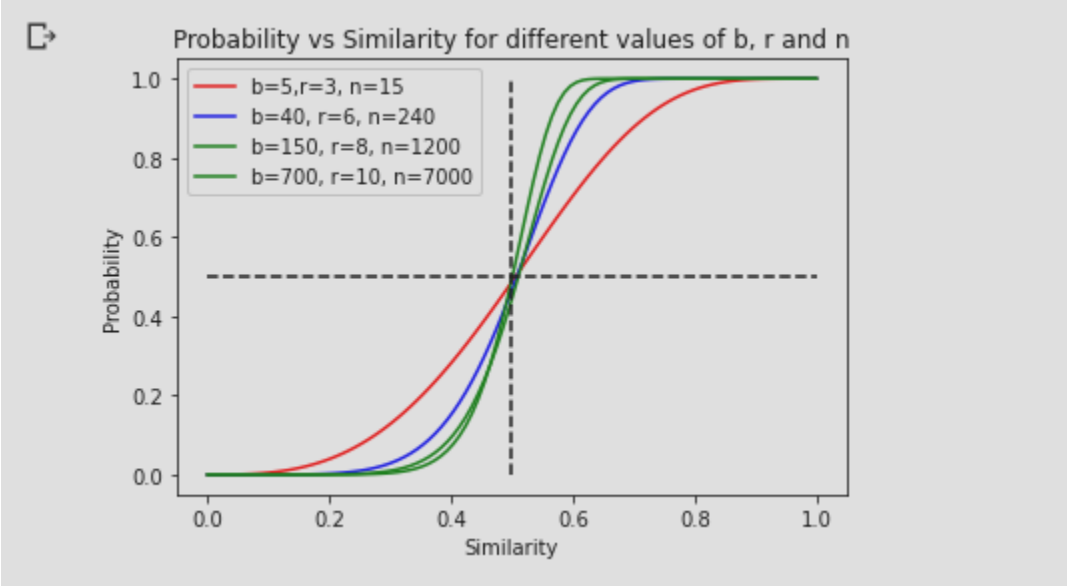
**For graph**

Say columns C1 and C2 have similarity t  Pick any band (r rows)

- Prob. that all rows in the band equal = t^r
- Prob. that some row in the band unequal = 1 - t^r
- Prob. that no band identical = (1 - t^r ) b
- Prob. that at least 1 band identical = 1 - (1 - t^r )^b

So we plot this probability vs s(threshold values)




## Conclusion

I created the k-shingles as described in the task, and then I used them to calculate the Jaccard and Levenshtein distance. I repeated this experiment with Kannada, which is an Indian language. After that, I developed a function to create a min-hash signature of the k-shingles. Min-hashing involves compressing the sets of unique k-shingles into smaller representations called 'signatures'. The

size of the signature depends on the number of hashes used, and a larger number of hashes leads to a more accurate representation of the k-shingles. Min-hashing is used to approximate the Jaccard distance more quickly, but the approximation is not completely accurate.