Mini Project 2

This project is designed to test the similarity or dissimilarity between different strings of textual data using multiple algorithms that are designed to compare data quantitatively. I have used the example data given in the prompt as well as other textual data i found on Kaggle as the test cases.

The objective of this lab is to implement various algorithms on a given set of strings.

Data

The Sample data provided in the prompt.

- This is a textual English document. The quick brown fox jumps over the lazy dog. Ths is a document.
- This is a textual English document. The quick white wolf eats the lazy sheep.
- This is a textual English document. The slow brown fox jumps into the quizzical dog.
- This is a textual English document. The slow white wolf lays next to the lazy dog.
- This is a textual English document. The quick brown fox jumps over the laz cat.
- This is a textual English document. The quick brown fox jumps over the lazy dog. This is a document.

Larger string data for some variation in input size and complexity.

- Mayor Bill de Blasio's counsel and chief legal adviser, Maya Wiley, is resigning next month from her City Hall position to become the chairwoman of the Civilian Complaint Review Board, New York City's independent oversight agency for the Police Department.
- In the early morning hours of Labor Day last year, a group of gunmen from the 8-Trey street gang made their way through a crowd of revelers gathered near a Brooklyn public housing project to celebrate J'ouvert, a pre-dawn party that precedes the annual West Indian American Day Parade. The housing project was "enemy territory," the authorities said, the stronghold of a rival gang, the Folk Nation.
- Londoners will decide whether to elect Sadiq Khan, the Labour front-runner, as their first Muslim mayor on Thursday, but in elections being held across Britain the wider results will pose a bigger test for the party as a whole — especially for Jeremy Corbyn, its left-wing leader.
- Federal prosecutors in Manhattan asked a judge on Wednesday to deny bail to Reza Zarrab, a prominent gold trader based in Turkey who has been jailed in New York pending trial on charges that he conspired to violate United States sanctions on Iran.
- The book, Bible Emoji: Scripture 4 Millenials, is based on a Twitter feed that has been posting emojified Bible verses for almost a year. (The author intentionally misspelled "millennial," he said, to thumb his nose at critics who might dismiss the project as the work of the young and foolish.)
- Peter Parker, a.k.a. Spider-Man, grew up in Queens, and several superheroes have lived in Manhattan, including Captain America, a son of the Lower East Side. But this Saturday and Sunday, Brooklyn will pack the most kapow of any borough, producing dozens of its own young challengers, ready to vanquish evil and soar to great heights (at least in their imaginations).

```
;; namespace and requirements
```

In [141…
```
(ns main.core
  (:require [clojure.string :as str]
            [clojure.java.io :as io]))

;; Defining our textual documents

    ;; control Test string
(def Tester ["This is a test."
             "This is a test."])


    ;; Sample docs
(def sample_docs ["This is a textual English document. The quick brown fox jumps over t
                  "This is a textual English document. The quick white wolf eats the la
                  "This is a textual English document. The slow brown fox jumps into th
                  "This is a textual English document. The slow white wolf lays next to
                  "This is a textual English document. The quick brown fox jumps over t
                  "This is a textual English document. The quick brown fox jumps over t

    ;; Larger Varied docs
(def larger_docs ["Mayor Bill de Blasio’s counsel and chief legal adviser, Maya Wiley,
                  "In the early morning hours of Labor Day last year, a group of gunmen
                  "Londoners will decide whether to elect Sadiq Khan, the Labour front-
                  "Federal prosecutors in Manhattan asked a judge on Wednesday to deny
                  "The book, Bible Emoji: Scripture 4 Millenials, is based on a Twitter
                  "Peter Parker, a.k.a. Spider-Man, grew up in Queens, and several supe

;; (def sample_binary ["01001101 01100001 01111001 01101111 01110010 00100000 01000010
;;                      "01001001 01101110 00100000 01110100 01101000 01100101 00100000
;;                      "01001100 01101111 01101110 01100100 01101111 01101110 01100101
;;                      "01000110 01100101 01100100 01100101 01110010 01100001 01101100
;;                      "01010100 01101000 01100101 00100000 01100010 01101111 01101111
;;                      "01010000 01100101 01110100 01100101 01110010 00100000 01010000
```

Out[141… #'main.core/larger_docs

Additional/Helper functions

Function for splitting the textual string input

In [142…
```
; Take input, make uppercase, split by omitting all except numerical and alphabet chara
(defn split-words [input]
  (str/split (str/upper-case input) #"\W+"))
```

Out[142… #'main.core/split-words

Algorithims explored

- Jaccard index
- Hamming distance

Algorithm #1 - Jaccard coefficient

- The Jaccard coefficient alogrithm will calculate the similarity betweeen two or more finite datasets. It will base its result on the size of the intersection divided by the size of the union.

In basic terms it is --> (similar content/total content)

In [143…
```
;; ALGO 1 - Jaccard similarity
(defn Jaccard-Similarity [string1 string2]
  (let [string1 (set (split-words string1))
        string2 (set (split-words string2))]
    (clojure.set/intersection string1 string2)))
```

Out[143…
```
#'main.core/Jaccard-Similarity
```

Testing using Jaccard Similarity

In [144…
```
(println  (str "TEST CASE - To see if it works if identical string is compared\n"))
(println (str "Jaccard Similarity = " (Jaccard-Similarity (nth Tester 0) (nth Tester 1)
```

```
TEST CASE - To see if it works if identical string is compared

Jaccard Similarity = #{"IS" "THIS" "TEST" "A"}
```

Out[144…
```
nil
```

In [145…
```
;; (println (str "Running Jaccard Similarity check on Sample documents\n"))
;; (println (str "1. Test on sample documents 1 & 2 = " (Jaccard-Similarity (nth sample
;; (println (str "\t Number of matching words on Test #1 --> " (count (Jaccard-Similari
;; (println (str "2. Test on sample documents 2 & 3 = " (Jaccard-Similarity (nth sample
;; (println (str "\t Number of matching words on Test #2 --> " (count (Jaccard-Similari
;; (println (str "3. Test on sample documents 3 & 4 = " (Jaccard-Similarity (nth sample
;; (println (str "\t Number of matching words on Test #3 --> " (count (Jaccard-Similari
;; (println (str "4. Test on sample documents 4 & 5 = " (Jaccard-Similarity (nth sample
;; (println (str "\t Number of matching words on Test #4 --> " (count (Jaccard-Similari
;; (println (str "5. Test on sample documents 5 & 6 = " (Jaccard-Similarity (nth sample
;; (println (str "\t Number of matching words on Test #5 --> " (count (Jaccard-Similari
```

Out[145…

In [146…
```
;; (println (str "Running Jaccard Similarity check on Sample documents\n"))
;; (println (str "1. Test on larger documents 1 & 2 = " (Jaccard-Similarity (nth larger
;; (println (str "\t Number of matching words on Test #1 --> " (count (Jaccard-Similari
;; (println (str "2. Test on larger documents 2 & 3 = " (Jaccard-Similarity (nth larger
;; (println (str "\t Number of matching words on Test #2 --> " (count (Jaccard-Similari
;; (println (str "3. Test on larger documents 3 & 4 = " (Jaccard-Similarity (nth larger
;; (println (str "\t Number of matching words on Test #3 --> " (count (Jaccard-Similari
;; (println (str "4. Test on larger documents 4 & 5 = " (Jaccard-Similarity (nth larger
;; (println (str "\t Number of matching words on Test #4 --> " (count (Jaccard-Similari
;; (println (str "5. Test on larger documents 5 & 6 = " (Jaccard-Similarity (nth larger
;; (println (str "\t Number of matching words on Test #5 --> " (count (Jaccard-Similari
```

Out[146…

Algorithm #2 - Hamming Distance

Hamming distance is a metric for comparing two binary data strings. While comparing two binary strings of equal length, Hamming distance is the number of bit positions in which the two bits are different. The Hamming distance between two strings, a and b is denoted as d(a,b).

The standard algorithm, typicaly assumes that strings comared are of equal lenght. The function below is for addressing unequal strings. The function adds characters in to the string that is less to

then match the length of the max string.

In [147…
```
(defn balance[a b]
    (if(<(count a)(count b))
        (do (let [x (-(-(count b)(count a)) 1)]
            (str a (dotimes [i x]) ))
            (count (filter true? (map = a b))))
        (do (let [x (-(-(count a)(count b)) 1)]
            (str b (dotimes [i x]) ))
            (count (filter true? (map = a b)))))
    )
```

Out[147…
```
#'main.core/balance
```

Creating the standard function that accounts for equal strings while adding the balance function.

In [148…
```
;; ALGO 2 - HAMMING DISTANCE
(defn hammingdistance [string1 string2]
  (if (= (count string1) (count string2))
    (count (filter true? (map = string1 string2)))
    (balance string1 string2)))
```

Out[148…
```
#'main.core/hammingdistance
```

Testing using Hamming Distance

In [149…
```
(println  (str "!!!TEST CASE - To see if it works if identical string is compared\n"))
(println (str "Hamming Distance = "  (hammingdistance (nth Tester 0) (nth Tester 1))))
```

```
!!!TEST CASE - To see if it works if identical string is compared

Hamming Distance = 15
```

Out[149…
```
nil
```

In [150…
```
;; (println (str "Running Hamming Distance check on Sample documents\n"))
;; (println (str "1. Test on sample documents 1 & 2 = "(hammingdistance (nth sample_doc
;; (println (str "2. Test on sample documents 2 & 3 = "(hammingdistance (nth sample_doc
;; (println (str "3. Test on sample documents 3 & 4 = "(hammingdistance (nth sample_doc
;; (println (str "4. Test on sample documents 4 & 5 = "(hammingdistance (nth sample_doc
;; (println (str "5. Test on sample documents 5 & 6 = "(hammingdistance (nth sample_doc
```

Out[150…

In [151…
```
;; (println (str "Running Hamming Distance check on Larger documents\n"))
;; (println (str "1. Test on larger documents 1 & 2 = "(hammingdistance (nth larger_doc
;; (println (str "2. Test on larger documents 2 & 3 = "(hammingdistance (nth larger_doc
;; (println (str "3. Test on larger documents 3 & 4 = "(hammingdistance (nth larger_doc
;; (println (str "4. Test on larger documents 4 & 5 = "(hammingdistance (nth larger_doc
;; (println (str "5. Test on larger documents 5 & 6 = "(hammingdistance (nth larger_doc
```

Out[151…

Testing on actual data

Algo #1 - Jaccard Index

In [152…

```
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #0. Matching Words on sample dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #1. Matching Words on sample dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #2. Matching Words on sample dataset #"x": " (Jaccard
      (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Simila
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #3. Matching Words on sample dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #4. Matching Words on sample dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #5. Matching Words on sample dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
```

```
Base document #0. Matching Words on sample dataset #0: #{"THS" "IS" "DOCUMENT" "FOX" "TE
XTUAL" "THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "LAZY" "QUICK" "OVER" "DOG"}
Number of matching words on Test #0 --> 15
Base document #0. Matching Words on sample dataset #1: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "LAZY" "QUICK"}
Number of matching words on Test #1 --> 9
Base document #0. Matching Words on sample dataset #2: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "DOG"}
Number of matching words on Test #2 --> 11
Base document #0. Matching Words on sample dataset #3: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "LAZY" "DOG"}
Number of matching words on Test #3 --> 9
Base document #0. Matching Words on sample dataset #4: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "QUICK" "OVER"}
Number of matching words on Test #4 --> 12
Base document #0. Matching Words on sample dataset #5: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "LAZY" "QUICK" "OVER" "DOG"}
Number of matching words on Test #5 --> 14


Base document #1. Matching Words on sample dataset #0: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "LAZY" "QUICK"}
Number of matching words on Test #0 --> 9
Base document #1. Matching Words on sample dataset #1: #{"IS" "EATS" "DOCUMENT" "TEXTUA
L" "THE" "SHEEP" "THIS" "ENGLISH" "WOLF" "WHITE" "A" "LAZY" "QUICK"}
Number of matching words on Test #1 --> 13
Base document #1. Matching Words on sample dataset #2: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
```

```
"THIS" "ENGLISH" "A"}
Number of matching words on Test #2 --> 7
Base document #1. Matching Words on sample dataset #3: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "WOLF" "WHITE" "A" "LAZY"}
Number of matching words on Test #3 --> 10
Base document #1. Matching Words on sample dataset #4: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "QUICK"}
Number of matching words on Test #4 --> 8
Base document #1. Matching Words on sample dataset #5: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "LAZY" "QUICK"}
Number of matching words on Test #5 --> 9


Base document #2. Matching Words on sample dataset #0: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "DOG"}
Number of matching words on Test #0 --> 11
Base document #2. Matching Words on sample dataset #1: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A"}
Number of matching words on Test #1 --> 7
Base document #2. Matching Words on sample dataset #2: #{"QUIZZICAL" "IS" "DOCUMENT" "FO
X" "TEXTUAL" "THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "INTO" "SLOW" "DOG"}
Number of matching words on Test #2 --> 14
Base document #2. Matching Words on sample dataset #3: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "SLOW" "DOG"}
Number of matching words on Test #3 --> 9
Base document #2. Matching Words on sample dataset #4: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A"}
Number of matching words on Test #4 --> 10
Base document #2. Matching Words on sample dataset #5: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "DOG"}
Number of matching words on Test #5 --> 11


Base document #3. Matching Words on sample dataset #0: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "LAZY" "DOG"}
Number of matching words on Test #0 --> 9
Base document #3. Matching Words on sample dataset #1: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "WOLF" "WHITE" "A" "LAZY"}
Number of matching words on Test #1 --> 10
Base document #3. Matching Words on sample dataset #2: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "SLOW" "DOG"}
Number of matching words on Test #2 --> 9
Base document #3. Matching Words on sample dataset #3: #{"LAYS" "NEXT" "IS" "DOCUMENT"
"TEXTUAL" "THE" "THIS" "ENGLISH" "TO" "WOLF" "WHITE" "A" "LAZY" "SLOW" "DOG"}
Number of matching words on Test #3 --> 15
Base document #3. Matching Words on sample dataset #4: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A"}
Number of matching words on Test #4 --> 7
Base document #3. Matching Words on sample dataset #5: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "LAZY" "DOG"}
Number of matching words on Test #5 --> 9


Base document #4. Matching Words on sample dataset #0: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "QUICK" "OVER"}
Number of matching words on Test #0 --> 12
Base document #4. Matching Words on sample dataset #1: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "QUICK"}
Number of matching words on Test #1 --> 8
Base document #4. Matching Words on sample dataset #2: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A"}
Number of matching words on Test #2 --> 10
Base document #4. Matching Words on sample dataset #3: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A"}
Number of matching words on Test #3 --> 7
```

```
Base document #4. Matching Words on sample dataset #4: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "LAZ" "ENGLISH" "A" "QUICK" "OVER" "CAT"}
Number of matching words on Test #4 --> 14
Base document #4. Matching Words on sample dataset #5: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "QUICK" "OVER"}
Number of matching words on Test #5 --> 12


Base document #5. Matching Words on sample dataset #0: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "LAZY" "QUICK" "OVER" "DOG"}
Number of matching words on Test #0 --> 14
Base document #5. Matching Words on sample dataset #1: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "LAZY" "QUICK"}
Number of matching words on Test #1 --> 9
Base document #5. Matching Words on sample dataset #2: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "DOG"}
Number of matching words on Test #2 --> 11
Base document #5. Matching Words on sample dataset #3: #{"IS" "DOCUMENT" "TEXTUAL" "THE"
"THIS" "ENGLISH" "A" "LAZY" "DOG"}
Number of matching words on Test #3 --> 9
Base document #5. Matching Words on sample dataset #4: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "QUICK" "OVER"}
Number of matching words on Test #4 --> 12
Base document #5. Matching Words on sample dataset #5: #{"IS" "DOCUMENT" "FOX" "TEXTUAL"
"THE" "BROWN" "JUMPS" "THIS" "ENGLISH" "A" "LAZY" "QUICK" "OVER" "DOG"}
Number of matching words on Test #5 --> 14
```

Out[152… nil

Jaccard Index	sample doc 0	sample doc 1	sample doc 2	sample doc 3	sample doc 4	sample doc 5
sample document 0	15	9	11	9	12	14
sample document 1	9	13	7	10	8	9
sample document 2	11	7	14	9	10	11
sample document 3	9	10	9	15	7	9
sample document 4	12	8	10	7	14	12
sample document 5	14	9	11	9	12	14

In [153…

```
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #0. Matching Words on larger dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #1. Matching Words on larger dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
```

```
    (println  (str "Base document #2. Matching Words on larger dataset #"x": " (Jaccard
      (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Simila
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #3. Matching Words on larger dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #4. Matching Words on larger dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
(println "\n")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #5. Matching Words on larger dataset #"x": " (Jaccard
     (println (str "Number of matching words on Test #"x" --> " (count (Jaccard-Similar
    (recur (+ x 1))))
```

```
Base document #0. Matching Words on larger dataset #0: #{"FROM" "AGENCY" "MONTH" "FOR"
"MAYA" "MAYOR" "INDEPENDENT" "BLASIO" "NEW" "BILL" "CITY" "NEXT" "IS" "POSITION" "S" "CO
MPLAINT" "REVIEW" "CIVILIAN" "THE" "DEPARTMENT" "AND" "CHAIRWOMAN" "HALL" "RESIGNING" "P
OLICE" "TO" "BECOME" "OVERSIGHT" "DE" "BOARD" "CHIEF" "LEGAL" "COUNSEL" "ADVISER" "WILE
Y" "HER" "YORK" "OF"}
Number of matching words on Test #0 --> 38
Base document #0. Matching Words on larger dataset #1: #{"FROM" "THE" "TO" "OF"}
Number of matching words on Test #1 --> 4
Base document #0. Matching Words on larger dataset #2: #{"FOR" "MAYOR" "THE" "TO"}
Number of matching words on Test #2 --> 4
Base document #0. Matching Words on larger dataset #3: #{"NEW" "TO" "YORK"}
Number of matching words on Test #3 --> 3
Base document #0. Matching Words on larger dataset #4: #{"FOR" "IS" "THE" "AND" "TO" "O
F"}
Number of matching words on Test #4 --> 6
Base document #0. Matching Words on larger dataset #5: #{"THE" "AND" "TO" "OF"}
Number of matching words on Test #5 --> 4


Base document #1. Matching Words on larger dataset #0: #{"FROM" "THE" "TO" "OF"}
Number of matching words on Test #0 --> 4
Base document #1. Matching Words on larger dataset #1: #{"ANNUAL" "FROM" "YEAR" "LABOR"
"GANG" "TERRITORY" "RIVAL" "LAST" "8" "THAT" "PRE" "DAWN" "EARLY" "STREET" "THEIR" "J"
"INDIAN" "THE" "ENEMY" "WAY" "REVELERS" "MADE" "MORNING" "BROOKLYN" "FOLK" "WAS" "HOURS"
"OUVERT" "GUNMEN" "TO" "PROJECT" "HOUSING" "TREY" "AMERICAN" "PRECEDES" "A" "PARADE" "CE
LEBRATE" "STRONGHOLD" "DAY" "PUBLIC" "CROWD" "WEST" "NATION" "IN" "THROUGH" "NEAR" "PART
Y" "AUTHORITIES" "GATHERED" "GROUP" "SAID" "OF"}
Number of matching words on Test #1 --> 53
Base document #1. Matching Words on larger dataset #2: #{"THEIR" "THE" "TO" "A" "IN" "PA
RTY"}
Number of matching words on Test #2 --> 6
Base document #1. Matching Words on larger dataset #3: #{"THAT" "TO" "A" "IN"}
Number of matching words on Test #3 --> 4
Base document #1. Matching Words on larger dataset #4: #{"YEAR" "THAT" "THE" "TO" "PROJE
CT" "A" "SAID" "OF"}
Number of matching words on Test #4 --> 8
Base document #1. Matching Words on larger dataset #5: #{"THEIR" "THE" "BROOKLYN" "TO"
"A" "IN" "OF"}
Number of matching words on Test #5 --> 7


Base document #2. Matching Words on larger dataset #0: #{"FOR" "MAYOR" "THE" "TO"}
```

```
Number of matching words on Test #0 --> 4
Base document #2. Matching Words on larger dataset #1: #{"THEIR" "THE" "TO" "A" "IN" "PA
RTY"}
Number of matching words on Test #1 --> 6
Base document #2. Matching Words on larger dataset #2: #{"MUSLIM" "DECIDE" "HELD" "FOR"
"MAYOR" "LONDONERS" "ON" "LABOUR" "LEFT" "ACROSS" "CORBYN" "THEIR" "SADIQ" "RESULTS" "EL
ECTIONS" "POSE" "WHOLE" "BIGGER" "THE" "BEING" "THURSDAY" "BRITAIN" "TEST" "ESPECIALLY"
"TO" "KHAN" "RUNNER" "AS" "WING" "FRONT" "WIDER" "A" "WILL" "WHETHER" "JEREMY" "ITS" "EL
ECT" "IN" "LEADER" "PARTY" "BUT" "FIRST"}
Number of matching words on Test #2 --> 42
Base document #2. Matching Words on larger dataset #3: #{"ON" "TO" "A" "IN"}
Number of matching words on Test #3 --> 4
Base document #2. Matching Words on larger dataset #4: #{"FOR" "ON" "THE" "TO" "AS" "A"}
Number of matching words on Test #4 --> 6
Base document #2. Matching Words on larger dataset #5: #{"THEIR" "THE" "TO" "A" "WILL"
"ITS" "IN" "BUT"}
Number of matching words on Test #5 --> 8


Base document #3. Matching Words on larger dataset #0: #{"NEW" "TO" "YORK"}
Number of matching words on Test #0 --> 3
Base document #3. Matching Words on larger dataset #1: #{"THAT" "TO" "A" "IN"}
Number of matching words on Test #1 --> 4
Base document #3. Matching Words on larger dataset #2: #{"ON" "TO" "A" "IN"}
Number of matching words on Test #2 --> 4
Base document #3. Matching Words on larger dataset #3: #{"STATES" "WEDNESDAY" "JAILED"
"SANCTIONS" "DENY" "THAT" "CONSPIRED" "NEW" "ON" "IRAN" "WHO" "REZA" "HAS" "UNITED" "TRI
AL" "PENDING" "PROSECUTORS" "TURKEY" "JUDGE" "TRADER" "TO" "BEEN" "BAIL" "BASED" "HE" "V
IOLATE" "ASKED" "CHARGES" "FEDERAL" "A" "MANHATTAN" "GOLD" "ZARRAB" "YORK" "IN" "PROMINE
NT"}
Number of matching words on Test #3 --> 36
Base document #3. Matching Words on larger dataset #4: #{"THAT" "ON" "WHO" "HAS" "TO" "B
EEN" "BASED" "HE" "A"}
Number of matching words on Test #4 --> 9
Base document #3. Matching Words on larger dataset #5: #{"TO" "A" "MANHATTAN" "IN"}
Number of matching words on Test #5 --> 4


Base document #4. Matching Words on larger dataset #0: #{"FOR" "IS" "THE" "AND" "TO" "O
F"}
Number of matching words on Test #0 --> 6
Base document #4. Matching Words on larger dataset #1: #{"YEAR" "THAT" "THE" "TO" "PROJE
CT" "A" "SAID" "OF"}
Number of matching words on Test #1 --> 8
Base document #4. Matching Words on larger dataset #2: #{"FOR" "ON" "THE" "TO" "AS" "A"}
Number of matching words on Test #2 --> 6
Base document #4. Matching Words on larger dataset #3: #{"THAT" "ON" "WHO" "HAS" "TO" "B
EEN" "BASED" "HE" "A"}
Number of matching words on Test #3 --> 9
Base document #4. Matching Words on larger dataset #4: #{"YEAR" "DISMISS" "FOR" "4" "THA
T" "ON" "VERSES" "INTENTIONALLY" "YOUNG" "ALMOST" "WHO" "MISSPELLED" "IS" "BOOK" "MIGHT"
"TWITTER" "AT" "BIBLE" "WORK" "HAS" "THE" "AND" "FEED" "TO" "BEEN" "PROJECT" "BASED" "H
E" "HIS" "SCRIPTURE" "MILLENNIAL" "AS" "MILLENIALS" "A" "NOSE" "CRITICS" "POSTING" "FOOL
ISH" "THUMB" "AUTHOR" "EMOJI" "EMOJIFIED" "SAID" "OF"}
Number of matching words on Test #4 --> 44
Base document #4. Matching Words on larger dataset #5: #{"YOUNG" "AT" "THE" "AND" "TO"
"A" "OF"}
Number of matching words on Test #5 --> 7


Base document #5. Matching Words on larger dataset #0: #{"THE" "AND" "TO" "OF"}
Number of matching words on Test #0 --> 4
Base document #5. Matching Words on larger dataset #1: #{"THEIR" "THE" "BROOKLYN" "TO"
"A" "IN" "OF"}
Number of matching words on Test #1 --> 7
```

```
Base document #5. Matching Words on larger dataset #2: #{"THEIR" "THE" "TO" "A" "WILL"
"ITS" "IN" "BUT"}
Number of matching words on Test #2 --> 8
Base document #5. Matching Words on larger dataset #3: #{"TO" "A" "MANHATTAN" "IN"}
Number of matching words on Test #3 --> 4
Base document #5. Matching Words on larger dataset #4: #{"YOUNG" "AT" "THE" "AND" "TO"
"A" "OF"}
Number of matching words on Test #4 --> 7
Base document #5. Matching Words on larger dataset #5: #{"K" "GREW" "DOZENS" "PACK" "YOU
NG" "BOROUGH" "MAN" "SON" "SPIDER" "SUPERHEROES" "THEIR" "SIDE" "AT" "CHALLENGERS" "AMER
ICA" "GREAT" "UP" "READY" "THE" "EAST" "SUNDAY" "EVIL" "BROOKLYN" "AND" "THIS" "ANY" "T
O" "SEVERAL" "KAPOW" "SOAR" "HAVE" "LEAST" "PARKER" "SATURDAY" "HEIGHTS" "CAPTAIN" "MOS
T" "A" "WILL" "LOWER" "MANHATTAN" "LIVED" "PETER" "OWN" "ITS" "IN" "IMAGINATIONS" "BUT"
"VANQUISH" "INCLUDING" "QUEENS" "OF" "PRODUCING"}
Number of matching words on Test #5 --> 53
```

Out[153… nil

Jaccard Index	larger doc 0	larger doc 1	larger doc 2	larger doc 3	larger doc 4	larger doc 5
larger document 0	38	4	4	3	6	4
larger document 1	4	53	6	4	8	7
larger document 2	4	6	42	4	6	8
larger document 3	3	4	4	36	9	4
larger document 4	6	8	6	9	44	7
larger document 5	4	7	8	4	7	53

Algo #2 - Hamming Distance

In [154…

```
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #0. Hamming distance on sample dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #1. Hamming distance on sample dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #2. Hamming distance on sample dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #3. Hamming distance on sample dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #4. Hamming distance on sample dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #5. Hamming distance on sample dataset #"x": " (hammi
     (recur (+ x 1))))
```

```
Base document #0. Hamming distance on sample dataset #0: 99
Base document #0. Hamming distance on sample dataset #1: 52
Base document #0. Hamming distance on sample dataset #2: 41
Base document #0. Hamming distance on sample dataset #3: 41
Base document #0. Hamming distance on sample dataset #4: 74
Base document #0. Hamming distance on sample dataset #5: 83

Base document #1. Hamming distance on sample dataset #0: 52
Base document #1. Hamming distance on sample dataset #1: 77
Base document #1. Hamming distance on sample dataset #2: 41
Base document #1. Hamming distance on sample dataset #3: 41
Base document #1. Hamming distance on sample dataset #4: 52
Base document #1. Hamming distance on sample dataset #5: 52

Base document #2. Hamming distance on sample dataset #0: 41
Base document #2. Hamming distance on sample dataset #1: 41
Base document #2. Hamming distance on sample dataset #2: 84
Base document #2. Hamming distance on sample dataset #3: 51
Base document #2. Hamming distance on sample dataset #4: 41
Base document #2. Hamming distance on sample dataset #5: 41

Base document #3. Hamming distance on sample dataset #0: 41
Base document #3. Hamming distance on sample dataset #1: 41
Base document #3. Hamming distance on sample dataset #2: 51
Base document #3. Hamming distance on sample dataset #3: 82
Base document #3. Hamming distance on sample dataset #4: 41
Base document #3. Hamming distance on sample dataset #5: 41

Base document #4. Hamming distance on sample dataset #0: 74
Base document #4. Hamming distance on sample dataset #1: 52
Base document #4. Hamming distance on sample dataset #2: 41
Base document #4. Hamming distance on sample dataset #3: 41
Base document #4. Hamming distance on sample dataset #4: 79
Base document #4. Hamming distance on sample dataset #5: 74

Base document #5. Hamming distance on sample dataset #0: 83
Base document #5. Hamming distance on sample dataset #1: 52
Base document #5. Hamming distance on sample dataset #2: 41
Base document #5. Hamming distance on sample dataset #3: 41
Base document #5. Hamming distance on sample dataset #4: 74
Base document #5. Hamming distance on sample dataset #5: 100
```

Out[154… `nil`

Hamming Distance	sample doc 0	sample doc 1	sample doc 2	sample doc 3	sample doc 4	sample doc 5
sample document 0	99	52	41	41	74	83
sample document 1	52	77	41	41	52	52
sample document 2	41	41	84	51	41	41
sample document 3	41	41	51	82	41	41
sample document 4	74	52	41	41	79	74
sample document 5	83	52	41	41	74	100

In [155…

```
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #0. Hamming distance on larger dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #1. Hamming distance on larger dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #2. Hamming distance on larger dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #3. Hamming distance on larger dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #4. Hamming distance on larger dataset #"x": " (hammi
     (recur (+ x 1))))
    (println  "")
(loop [x 0]
  (when(< x 6)
    (println  (str "Base document #5. Hamming distance on larger dataset #"x": " (hammi
     (recur (+ x 1))))
```

```
Base document #0. Hamming distance on larger dataset #0: 255
Base document #0. Hamming distance on larger dataset #1: 14
Base document #0. Hamming distance on larger dataset #2: 14
Base document #0. Hamming distance on larger dataset #3: 20
Base document #0. Hamming distance on larger dataset #4: 17
Base document #0. Hamming distance on larger dataset #5: 22

Base document #1. Hamming distance on larger dataset #0: 14
Base document #1. Hamming distance on larger dataset #1: 399
Base document #1. Hamming distance on larger dataset #2: 15
Base document #1. Hamming distance on larger dataset #3: 16
Base document #1. Hamming distance on larger dataset #4: 18
Base document #1. Hamming distance on larger dataset #5: 17

Base document #2. Hamming distance on larger dataset #0: 14
Base document #2. Hamming distance on larger dataset #1: 15
Base document #2. Hamming distance on larger dataset #2: 279
Base document #2. Hamming distance on larger dataset #3: 19
Base document #2. Hamming distance on larger dataset #4: 22
Base document #2. Hamming distance on larger dataset #5: 17

Base document #3. Hamming distance on larger dataset #0: 20
Base document #3. Hamming distance on larger dataset #1: 16
Base document #3. Hamming distance on larger dataset #2: 19
Base document #3. Hamming distance on larger dataset #3: 247
Base document #3. Hamming distance on larger dataset #4: 20
Base document #3. Hamming distance on larger dataset #5: 21

Base document #4. Hamming distance on larger dataset #0: 17
Base document #4. Hamming distance on larger dataset #1: 18
Base document #4. Hamming distance on larger dataset #2: 22
Base document #4. Hamming distance on larger dataset #3: 20
Base document #4. Hamming distance on larger dataset #4: 295
```

```
Base document #4. Hamming distance on larger dataset #5: 19

Base document #5. Hamming distance on larger dataset #0: 22
Base document #5. Hamming distance on larger dataset #1: 17
Base document #5. Hamming distance on larger dataset #2: 17
Base document #5. Hamming distance on larger dataset #3: 21
Base document #5. Hamming distance on larger dataset #4: 19
Base document #5. Hamming distance on larger dataset #5: 367
```

Out[155… `nil`

Hamming Distance	larger doc 0	larger doc 1	larger doc 2	larger doc 3	larger doc 4	larger doc 5
larger document 0	255	14	14	20	17	22
larger document 1	14	399	15	16	18	17
larger document 2	14	15	279	19	22	17
larger document 3	20	16	19	247	20	21
larger document 4	17	18	22	20	295	19
larger document 5	22	17	17	21	19	367