**סעיף :a**

**מחלקת :parser**

תפקיד המחלקה לבצע עיבוד לטקסט המקורי בעזרת שימוש בחוקיים במהלך העיבוד נבצע הסרה של מחרוזות לא רלוונטיות ,שינוי לפורמט אחיד של מחרוזות וכו'...

המחלקה תקבל אובייקט מסוג document ושדה אופציונלי של שימוש בstemmer במידה והמשתמש לא הזין בחירה המחלקה לא תשתמש בstemmer.

ראשית נשתמש בפונקציה tokenized\_parse()

הפונקציה תסיר את מילות המפתח ותווים לא רצויים ואימוג'ים ,תתקן את ייצוג המספרים לפורמט אחיד,

תרחיב את term המיצגים #,

תבצע הצגה אחיד לterm המייצגים % ותמצא ישויות.

**מחלקת :indexer**

תפקיד המחלקה עדכון ואחזקה של מבני הנתונים שמחזיקים בפלט של פעולת הפרסור.

המחלקה תקבל שני מילונים זמניים :post dictionary ו- inverted dictionary המילונים יחזיקו במידע חלקי הודות המחרוזות שנמצאו באיטרציה האחרונה.

מחלקת ה-indexer תחזיק 17 שדות של מילונים עבור inverted dictionary והתאמה עבור post dictionary.

השיטות

Update\_local\_posting\_dicts() update\_local\_inv\_dicts()

יגשו אל המילונים הזמניים ויחלקו אותם ל15 מילונים על פי האות התחילית שלהם .

השיטות:

Update\_global\_posting\_dicts() update\_global\_inv\_dicts()

יגשו אל התיקייה הרלוונטית בבסיס הנתונים וישמרו את המילון וימחקו אותו מזיכרון.

בעזרת השיטות הmerge נבצע איחוד לכל המילונים בשיטת העץ (נאחד שני בנים לאב)עד אשר יתקבלו 17 מילונים עבור inverted dictionary כאשר המפתח הוא המחרוזת המפורסרת וערך הוא מספר הטוויטים בהם הוא מופיע.ו-17 מילונים עבור post dictionary כאשר המפתח הוא המחרוזת המפורסרת והערך הוא רשימה של מידע הודות המחרוזת.

**מחלקת :stemmer**

שימוש בחבילה חיצונית nltk.stem .השימוש במחלקה כרוך בבקשה מצד המשתמש .במידה ויבחר להשתמש השיטה תקבל את הטוויט לאחר הפרסור ותבצע . stemming

**מחלקת reader:**

המחלקה ניגשת לדיסק ומבצעת יבוא למידע תוך התחשבות בסיומת המבוקשת.

**מחלקת searcher:**

המחלקה מקבלת שאילתא ומעבירה אותה למחלקת פרסר בכדי שתעבור את אותו התהליך שעברו הציוצים. לאחר מחלקת פרסר מחזירה את השאילתא המתוקנת המחלקה ניגשת למילונים שהשמורים בדיסק ומחפשת את המסמכים הרלוונטים (כאלה שמופיעה לפחות מילה אחת בהם).המחלקה תייצר מילון המחזיק עבור כל מילה את המסמכים הרלוונטים אליה.

**מחלקת ranker:**

המחלקה תקבל מילון של מסמכים רלוונטים ממחלקת searcher ותבצע דירוג למסמכים על פי הנוסחה שמפורטת בסעיף h.לבסוף המחלקה תמצא את כמות הציוצים שהמשתמש ביקש הרלוונטים ביותר.

**סעיף :b**

כיווון שאנו עובדים במחשבים הביתיים אשר מכילים 8gb-16gb זיכרון היה עלינו לבחור דרך מיטבית לפצל את המידע שאנו מחזיקים בזיכרון .לאחר ניתוח הנתונים ניתן היה לראות שעיקר הזיכרון נדרש להחזקת מילוני ה-posting ו-inverted חיפשנו דרך לפצל את המילונים ובחרנו לשלב בין שני שיטות שונות :

1.פיצול המילונים על פי התפלגות המילים באנגלית כך בקירוב נקבל פיזור אחיד של מידע בין המילונים.

בחרנו לפצל את המילונים כך שכל מילון יחזיק בקירוב 6% מהחרוזות.

המידע נלקח (<https://en.wikipedia.org/wiki/Letter_frequency> )

2. אתחול הזיכרון לאחר לכל היותר פרסור של 1,000,000 טויטים.

**סעיף c:**

בחרנו לשמור קבצי הposting באובייקט מסוג מילון כיוון שזמן הגישה הוא o(1) (<https://wiki.python.org/moin/TimeComplexity> )

בשביל גישה נוחה הגדרנו את המפתח זהה לאופן שבו מוגדר במילון .inverted

המידע ישמר בפורמט הבא

**term:[Tweet ID ,trem freq, term location , tf ,num unique terms in tweet, max\_tf ,date], [ ]..**

נפרט על השדות המידע :

* Tweet ID מזהה ייחודי עבור על ציוץ.
* Term frequency מספר הפעמים שהופיע המפתח (term) בציוץ .
* Tern location מיקומי המחרוזת בציוץ.
* Tf- term frequency מחושב על פי הנוסחה שנלמדה בהרצאה.
* Number of unique term in Tweet מספר המילים הייחודיות בציוץ.
* Max Tf- term frequency תדירות המילה בעלת הtf הגבוהה ביותר.
* Date תאריך בו נכתב הציוץ.

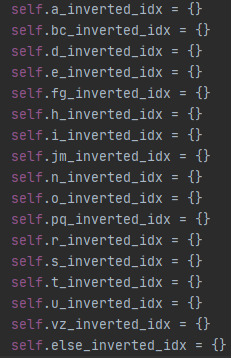
שדות אילו נשמרים עבור מופע של term בציוץ.

**סעיף d:**

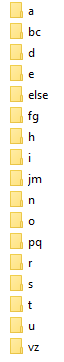
הסיבה לבחירת גודל קבוצת הציוצים הגיע לאחר בחינה של שימוש בזיכרון במחשב נייד.ההנחה הייתה כי הדרישה היא שהמנוע ירוץ בכל מצב מחשב סנדרטי בעל 8gb של זיכרון ההנחה נוספת היא שתוכנות נוספות רצות ברקע ועל כן התוכנית תדרוש במצב הקיצון שלה לכל היותר 4gb. לאחר מספר הרצות וניתוח הזיכרון הגענו למסקנה גודל קבוצת הציוצים המיטבי עבור התוכנית שלנו הוא 1,000,000 הציוצים לכל היותר וכן פיצול ל17 מילונים.

תהליך יצירה בזיכרון עבור קבצים הופכיים ו-posting:

אתחול מילונים:



לאחר מכן המילונים יתעדכנו במידע המפורסר שמגיע וישמרו בזיכרון בתיקיות:



**סעיף :e**

שמרנו שני שדות מידע נוספים בכדי לשפר את תוצאות האחזור.

* Tern location מיקומי המחרוזת בציוץ.
* Date תאריך בו נכתב הציוץ.

**תאריך הציוץ** כדי להחזיר ציוצים עדכניים לדוגמא אם נחפש ציוצים הנוגעים להנחיות הקורונה נרצה לקבל את הנחיות העדכניות ביותר.

**מיקום בציוץ** משתמש לתיעדוף מילים צמודות לדוגמא בשאילתא sweet apple נעדיף לקבל תוצאות אשר צמד המילים מופיעות יחד או שהמרחק בניהם קטן.

**סעיף :f**

החלטנו שכדי ליצור פורמט להצגת מטבעות גלובליים ולכן בנינו את **הפונקציה currency\_parse():**

USD->United States Dollar ו- United States Dollar-> United States Dollar

כיוון שהשפה שמשתמשים בה בטוויטר אינה אחידה. מצד אחד פרסומים רשמיים המשתמשים בשפה גבוהה ומצד שני שפת רחוב ,סלנג. החלטנו לבנות פונקציה אשר תגשר על הפערים ותרחיב את הסלנג אשר משתמשי טוויטר חובבים לביטוי המקורי.

דוגמא לשימוש בפונקציה **:extend\_contractions(**)

OMG->Oh My Gad, IRL->In Real Life cause->because ,

**סעיף g:**

חוקים נוספים ב-Parser:

* מילים מתחת לאורך 2 תווים לא ישמרו (לאחר ניתוח הנתונים מצאנו כי מילים אלה אינם אינפורמטיביות)
* בחרנו לא לשמור URL מסוג t.co ו- twitter.com,www,http

כמות המופעים הייתה חריגה על והמידע מספקים אינו רלוונטי

* הסרה של אימוגים ,הכתיבה בטוויטר לרוב היא בסגנון סלנג ועל כן המשתמשים בהם פעמיים רבות לעומת זאת אנו לא תומכים בשאילתא הכוללת אימוגי.

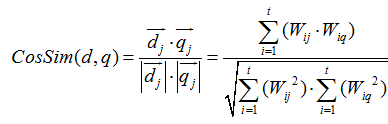
חוקים בindexer:

* פיצול מילונים (מפורט בחלק של חיסכון בזיכרון וזמן ריצה)
* איחוד מילונים בסוף הריצה (עבור כל אחד מ-17 המילונים)

**סעיף :h**

שימוש באלגוריתם Cosine Similarity Measure- **.CosSim**

השיטה מודדת את קוסינוס הזווית בין וקטור השאילתה לווקטור המסמך תוך שימוש בנוסחא:



Wij – משקל המילה i במסמך j

Wiq – משקל המילה i בשאילתה q

**סעיף i:**

**Utils-** שימוש במחלקת reader טעינת הקובץ אל הזיכרון והעברת קבצים**.**

**Path-**שימוש במחלקת reader במסגרת מציאת נתיב עבור הקבצים בדיסק

**Nltk.cotpus->stop\_words-** שימוש במאגר מידע של מילות קישור

**Math-**שימוש לחישוביים כגון LOG SQRT

**Json-**שימוש במחלקת searcher,indexer בכדי לשמור את המילונים בדיסק ועבור טעינה.

**Os-**שימוש במחלקת indexer לשם מחיקת קובץ מהדיסק

**GloVe-**שימוש במחלקת searcher בכדי להרחיב את מילות השאילתא במילים דומות. עבור כל מילה נבחר לקחת לכל היותר 2 מילים נוספות

**סעיף j:**

לאחר בחינת האחזור המידע בחרנו להתעלם(לאחר עיון בפורום) retweet וזאת כיוון שהוא גורם לנו לפרסור אותו הטקסט פעמיים רבות ומתוך הנחה כי הציוץ המקורי הוא הרלוונטי ביותר והוא נמצא בקורפוס. מלבד צריכת זיכרון וזמן ריצה ישנה בעיה גם בדירוג המסמכים כאשר ציוץ הוא רלוונטי פעמים רבות חלק מתוצאות הם retweet של ציוץ שמופיע בתוצאות.

**סעיף k:**

**יתרונות:**

* הרחבת הציוץ בכך שעבור מילים הכתובות בסלנג ומילים המופיעות בצורתם המקורית יופיעו בצורה זהה.
* צריכת זיכרון, התפלגות המילונים היא אחידה ולכן צריכת הזיכרון יציבה מיזוג מילון מסוג אחד ומסוג שני יצרכו בקירוב אותה כמות זיכרון.
* ביצועים ביצוע התוכנית לא דורש תנאים מקדימים(זיכרון גדול במיוחד או שימוש בהרבה חבילות חיצוניות)כך שמנוע ירוץ בכל מחשב סטנדרטי.

**חסרונות:**

* עדכון המילונים התוכנית מחזיקה הרבה קבצים בדיסק ולכן לאחר שלב האינדוקס האחרון מתבצעות קריאות רבות לדיסק בכדי למזג את המילונים.
* פרסור גם לאחר ניקוי תווים לא רצויים וכיוון שאנו משתמשים בפרסר שאנחנו כתבנו עדין ישנם תווים לא רצויים שתופסים מקום יקר במילונים.
* זמן ריצת,התוכנית מתבצעת בזמן ריצה אשר לא תואם זמן סביר שמשתמש היה מוכן לחכות.

**חלק 2:**

**סעיף a:**

Terms שונים ללא stemming: 2,738,966

**סעיף b:**

Terms שונים עם stemming: 2,646,289

**סעיף c:**

מספר המספרים שמופיעים במאגר הוא : 167686

**סעיף d:**

'COVID': 342,711,

'PEOPLE': 263,132,

'MASK': 254,606,

'COVID-19': 240,137,

'PANDEMIC': 232,045,

'#covid19': 190,135

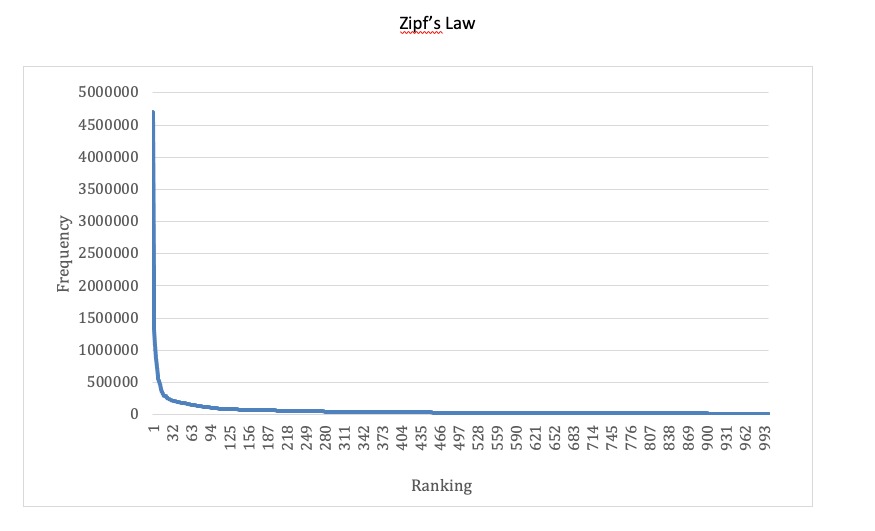
, 'GET': 164,795

, 'LIKE': 164,369

, 'CORONAVIRUS': 156,548

, 'WEAR': 190,417

**סעיף e:**



ניתן לראות כי עקב כמות המידע העצומה וכיוון שלא ניתן להציג את הגרף במלואו ההתפלגות לא זהה לזו שנלמדה בהרצאה

**סעיף :f**  .

גודל Posting ללא 3303014 Kb: stemming

גודל Posting עם 3292528 Kb: stemming

**סעיף g:**

חלוקת הזמן מתחלקת לשני חלקים זמן האינדוקס הכולל= זמן אינדוקס איטרציה+מיזוג מילונים בדיסק

זמן אינדוקס המילונים : 07:40

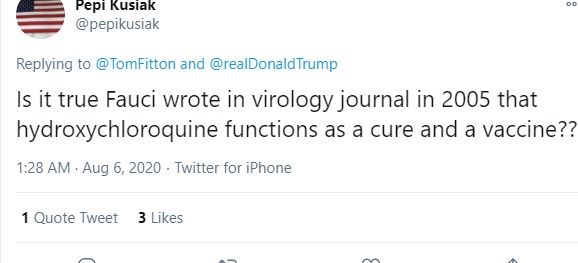
זמן מיזוג המילונים: 29:11

**סה"כ36.51 דקות**

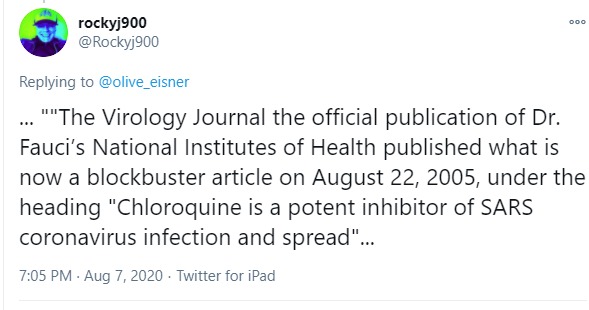
**סעיף h:**

**שאילתא מספר 1:**

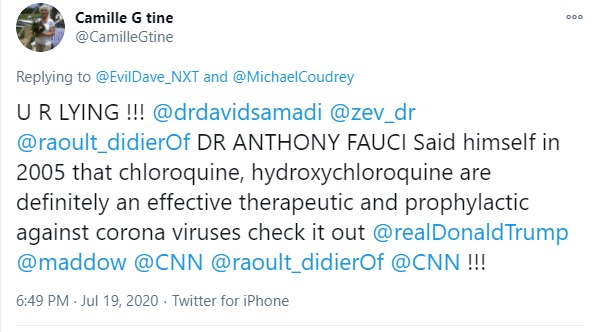
**Dr. Anthony Fauci wrote in a 2005 paper published in Virology Journal that hydroxychloroquine was effective in treating SARS**



שאילתא ובתוצאה המחוזרת ישנן מילים משותפות על כתיבת מאמרים ופרסומם



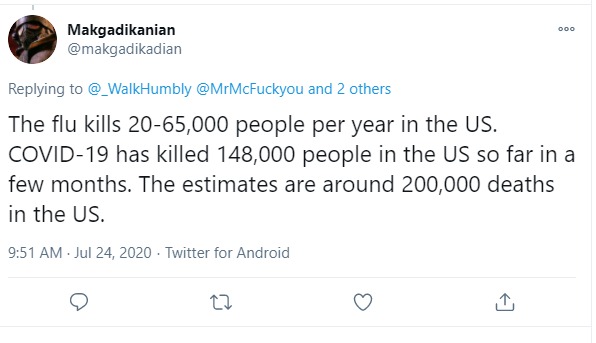
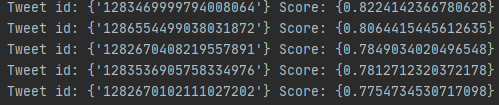
שאילתא ובתוצאה המחוזרת ישנן מילים משותפות מספרים משותפים מילים נרדפות



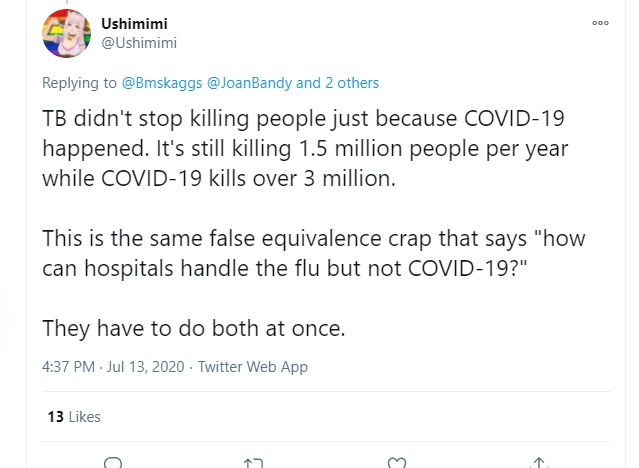
שאילתא ובתוצאה המחוזרת ישנן מילים משותפות מספרים משותפים מילים נרדפות

**שאילתא מספר 2:**

**The seasonal flu kills more people every year in the U.S. than COVID-19 has to date.**



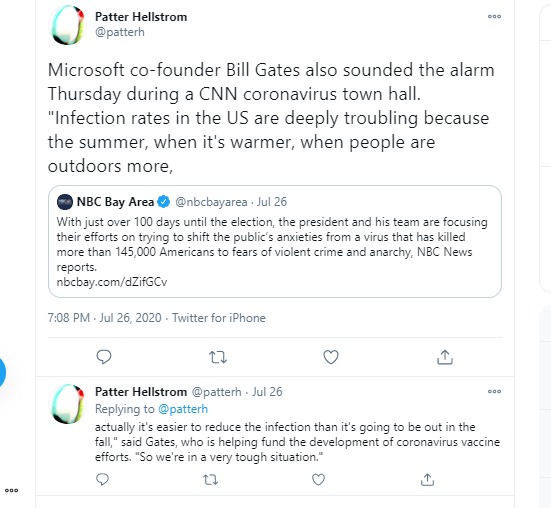
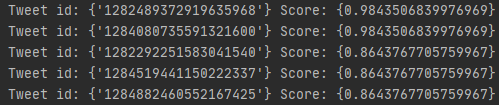
שאילתא ובתוצאה המחוזרת ישנן מילים משותפות מספרים משותפים מילים נרדפות כמו מילים מתחילות בצורה זהה **COVID-19** זהה



שאילתא ובתוצאה המחוזרת ישנן מילים משותפות מספרים משותפים מילים נרדפות כמו מילים מתחילות בצורה זהה **COVID-19** זהה

**שאילתא מספר 4:**

The coronavirus pandemic is a cover for a plan to implant trackable microchips and that the Microsoft co-founder Bill Gates is behind it

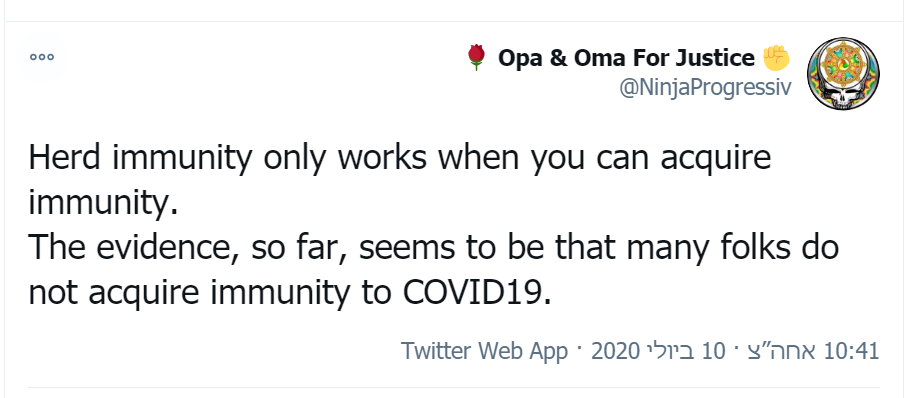
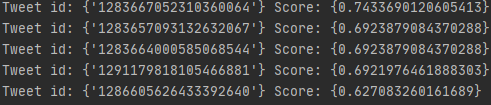


שאילתא ובתוצאה המחוזרת ישנן מילים משותפות מספרים משותפים מילים נרדפות כמו מילים מתחילות בצורה זהה ניתן לראות כי השאילתא מדברת על אותה החברה ואותו האדם

שאילתא ובתוצאה המחוזרת ישנן מילים משותפות מספרים משותפים מילים נרדפות כמו מילים

**שאילתא מספר 7:**

Herd immunity has been reached.

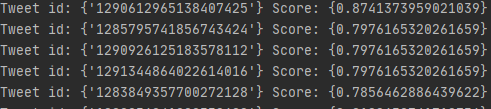


הציוץ מזכיר את אותה תופעה

שאר הציוצים הוסרו או שהחשבונות נסגרו

**שאילתא מספר 8:**

Children are “almost immune from this disease



שאילתא ובתוצאה המחוזרת ישנן מילים משותפות מספרים משותפים מילים נרדפות