המחלקה למדעי המידע

עבודה מסכמת בביג דאטה

שם קורס: ביג דאטה

מספר קורס: 803560501

שנה אקדמית: תשי״פ

סמסטר (בו ניתן הקורס): א׳

מוגש למרצה: ד״ר אריאל רוזנפלד

מגישות:

דיאנה גוטמן 305914061

מאשה שמידוב 320908361

הקדמה:

העבודה עוסקת בניתוח מאמרים בתחום למידת מכונה מתוך מאגר נרחב המכיל אלפי מאמרים, כאשר ברוב הכלים שעשינו בהם שימוש לא היה ניתן להריץ את הבדיקות על כל התוכן, ולכן העבודה כוללת ברוב הפעמים ניתוחים של חלקים מסוימים מתוך המאגר הגדול שכלל מאמרים מהשנים 1987 עד 2015.
בעבודה נעשה שימוש בשלל כלים שלמדנו בקורס Big Data, בחלק מהמקרים בExcel או כלים אחרים שמצאנו באינטרנט בהתאם לצרכים והשערות שרצינו לבדוק. גוף העבודה כולל 6 סעיפים. הסעיפים הראשונים בודקים נתונים כלליים ובסעיפים אחרים נעשה Drill down למאמרים ספציפיים שבעזרתם רצינו לבדוק השערות או לקבל ניתוח על הנתונים שנבחרו.

כל הקודים נמצאים בקבצים בפרויקט שפתחתנו ב GitHub.
הקישור לפרויקט: https://github.com/mashmashS/m_d_project_2019

1. <u>ניתוח חישובים בסיסיים</u>

על סמך הקובץ NIPS_1987-2015.CSV בדקנו כמה נתונים כלליים:

- כמות המאמרים שישנם סך הכל בקובץ - ע"י סכימה שעשינו על **העמודות** ניתן לראות את סך מספר המאמרים בקובץ בין השנים 1987 ל-2015, סך הכל 5811 מאמרים
- כמות המילים שישנן סך הכל במאמרים בכל השנים - את אותה הסכימה שמקודם בדקנו על מספר **השורות**. סך הכל ישנם 11463 מילים שונות בכל המאמרים
- מציאת המילה שמופיעה הכי הרבה פעמים והמילה שמופיעה הכי מעט פעמים - כדי למצוא את המילים הללו סכמנו את כל השורות, כלומר עבור כל מילה הצגנו כמה פעמים היא הופיעה סך הכול בכל המאמרים בין השנים 1987 ל-2015 ולאחר מכן מצאנו את הסכום שמופיע הכי מעט פעמים בעזרת פונקציות min ואת הסכום שהופיעה הכי הרבה פעמים בעזרת פונקציית max

את הקוד ניתן לראות בקובץ חישובים בסיסיים בGitHub (קישור כאן)

פלט:

```
C:\WINDOWS\system32\cmd.exe
1987_1       787
1987_2      2167
1987_3      1171
1987_4      1582
1987_5      2275
            ...
2015_399    2214
2015_400    1534
2015_401    2141
2015_402    2090
2015_403    2050
Length: 5811, dtype: int64
11040357
0        111
1        147
2        195
3         57
4         70
        ...
11458     52
11459     75
11460    147
11461    220
11462    117
Length: 11463, dtype: int64
80080
51
Press any key to continue . . .
```

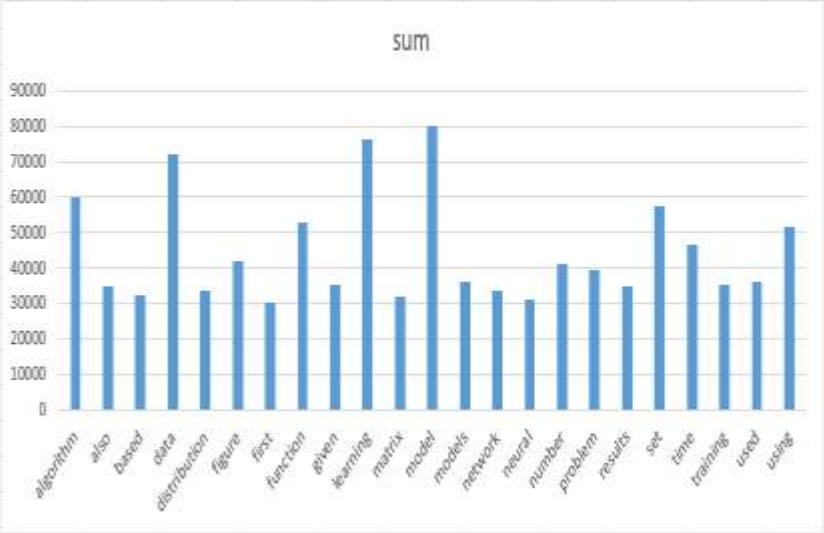
גם באקסל סכמנו את כל העמודות והשורות למציאת סך כל המילים שמופיעות בכל השנים ולמציאת המילה שהופיע הכי הרבה פעמים והמילה שהופיעה הכי מעט פעמים. נעזרנו בפונקציית VLOOKUP כדי לקבל את המילה המתאימה לסכום שקיבלנו- ראה צילום מסך כאן

תוצאות:

Additive factorial hidden Markov model (AFHMM) - מילה זאת הופיעה 51 פעמים והיא קיצור של תוספת למודל מרכוב. היא מופיע יחסית מעט פעמים היות וזאת מילה יחסית לא נפוצה ושייכת לתחום ספציפי של למידת מכונה. לעומת זאת המילה model הופיעה 80080 פעמים. תוצאה זו נתנת להסבר בכך שמדובר במילה כללית שיכולה להיות שייכת לתחומים רבים, זאת היות ומשתמשים הרבה במודלים במאמרים ומחקרים בתחומים רבים ומגוונים ובפרט בלמידת מכונה.

לאחר מכן בדקנו מה הן המילים פופולריות בעזרת דיאגרמת מקלות. שמנו את העמודה של המילים ליד העמודה של הסכומים וסיננו למילים שמופיעות מעל 30 אלף פעמים. ניתן לראות שהעמודה הארוכה ביותר היא של המילה model ואחריה המילה learning ולאחריה המילים data ו-algorithm. אלו מילים נפוצות מאוד בעולם הלמידת מכונה ולכן יופיעו פעמים רבות מאוד במאמרים בנושאים הללו.

	word	sum
288	algorithm	59769
332	also	34826
878	based	32307
2387	data	72295
2864	distributio	33518
3727	figure	42064
3774	first	30218
3998	function	52656
4160	given	35304
5595	learning	76406
6074	matrix	31721
6371	model	80080
6376	models	36171
6702	network	33429
6709	neural	31166
6868	number	41214
7905	problem	39612
8685	results	34715
9228	set	57526
10535	time	46402
10669	training	35308
11023	used	35968
11030	using	51628
11465		



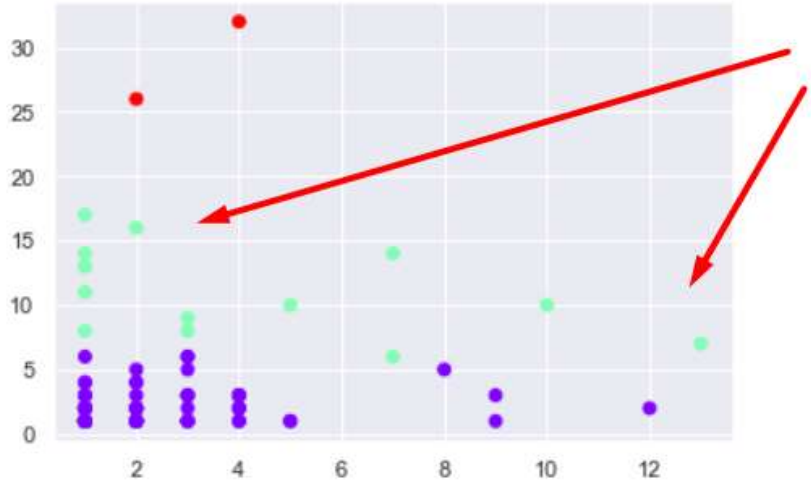
2. השוואה בין מאמרים לפי Kmeans (Clustering)

2.1 השוואה בין המאמר הראשון של שנת 1987 לעומת המאמר האחרון של 2015

תחילה העברנו את 2 העמודות של המאמר הראשון והאחרון לקובץ אקסל חדש בשם firstvslast.csv, ניתן לראות אותו בGitHub (קישור כאן). לקחנו 100 שורות ראשונות והורדנו את המילים שלא מופיעות כלל. תחילה הרצנו את האלגוריתם עם K=3 וכפי שניתן לראות מהגרף שלהלן, הפיזור אינו מתאים מכיוון שהנקודות שבקצוות בקבוצות הירוקה והסגולה הן במרחק גדול מאוד אחת מהשניה ויש הרבה נקודות חריגות, לדוגמא הנקודות הסגולות שבין הערך 8 לערך 12 שבציר הX לגמרי תלושות מהמרכז של הקבוצה הסגולה ולכן הוחלט לנסות עם **K=4**. ראה קוד ופלט בקובץ בGitHub כאן

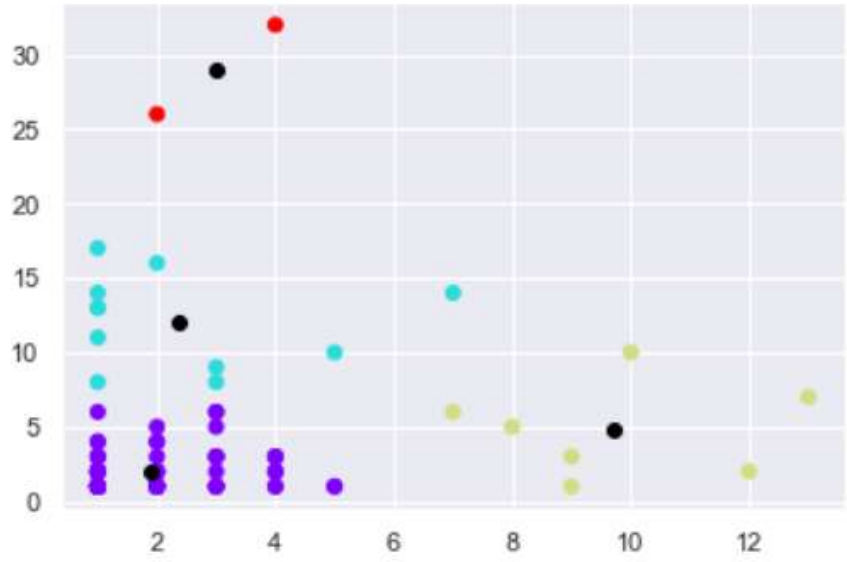
גרף עבור K=3:

```
Out[6]: <matplotlib.collections.PathCollection at 0x167be3cfb08>
```



גרף עבור K=4:

```
Out[2]: <matplotlib.collections.PathCollection at 0x1fc0a2c3e08>
```



מקרא:
מאמר 1- משנת 1987 -ציר X
מאמר 2- משנת 2015 - ציר Y

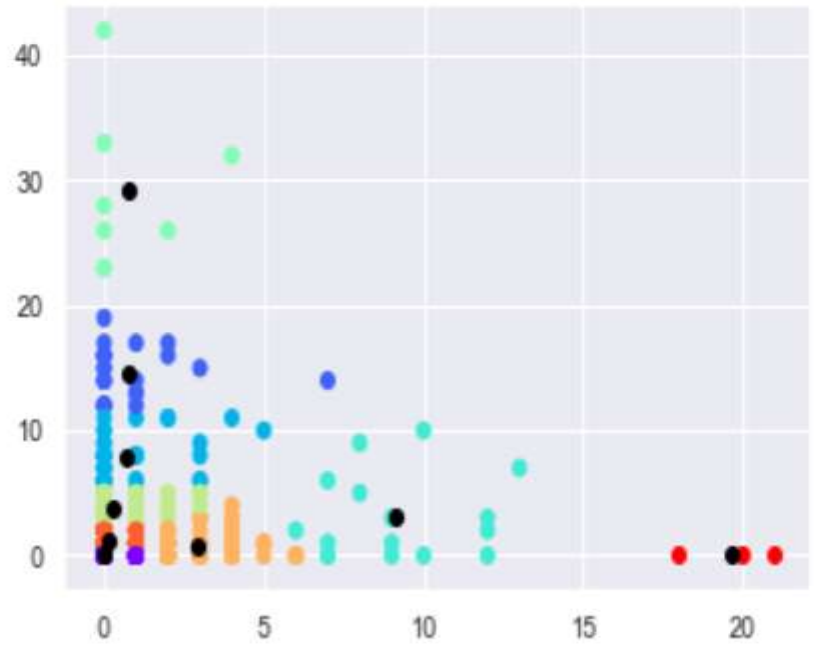
ניתן לראות מהגרף שלהלן שלאחר החלוקה לK=4 המרחק בין הנקודות בתוך כל אחת מהקבוצות הוא קטן יחסית מהמרחק בגרף הקודם וכמו כן הצפיפות של הנקודות בכל אחת מהקבוצות היא גבוהה מאוד יחסית לגרף הקודם. הנקודות הסגולות שבגרף שמרכזן הוא (1.9, 1.9375) וקרוב ל-0, אלו המילים שנמצאות הכי מעט בשני המאמרים והמרחקים ביניהן ממש קטנים היות והערכים שלהם דומים וקטנים מאוד. לאחר בדיקה אקראית

ניתן להבין שמדובר במילים כלליות שאינן קשורות לנושא המאמר כגון: abstract, also, basic וכיוצא באלו. הקבוצות השנייה והשלישית (תכלת וירוק) מראות את המילים הנפוצות לכל אחד משני המאמרים לחוד. כלומר, הירוקים נפוצים יותר במאמר הראשון ואילו התכולים נפוצים יותר במאמר השני. כלומר, ניתן לזהות את הנושאים של אותם מאמרים לפי הקבוצות האלו.
הנקודות הירוקות והתכלת החריגות (10,10) ו- (7,14) הנמצאות במרכז הגרף הן המילים הכי נפוצות המשותפות לשני המאמרים. מבדיקה בקובץ עלה שהמילים האלו הן learning ו-probability, כאשר המילה הראשונה קשורה באופן כללי לנושא של למידת מכונה - זה הוא הנושא הכללי אליו משויכים המאמרים והמילה השניה היא הבסיס עליו מסתמך תחום זה.
הקבוצה הרביעית (האדומה) והחריגה שמרכזה הוא (3,29) מכילה מילים המופיעות הכי הרבה פעמים במאמר השני ומעט מאוד במאמר 1. לדוגמא: מילה distribution מופיעה 32 פעמים במאמר 2 לעומת 4 במאמר 1, וגם המילה השנייה היא number שמופיעה 26 פעמים במאמר השני לעומת 2 בלבד במאמר הראשון. על סמך הנתונים האלה ניתן להסיק שמאמר 2 מתרכז יותר בנושא של סטטיסטיקה ואכן מבדיקה כללית של המאמר נמצא כי הוא עוסק במודלים סטטיסטיים.

לעומת זאת כאשר לקחנו את כל השורות (באותו הקובץ), הקבוצות של המילים הנפוצות לכל מאמר לחוד (הירוקה והאדומה) היו עוד יותר מרוחקות אחת מהשניה והצפיפות של הנקודות בכל אחת מהקבוצות הייתה גדולה יותר היות ונוספו עוד הרבה מילים חדשות. כעת, המספר הגבוה ביותר שהופיע הוא 22 לכן עשינו חלוקה ל9 מרכזים היות ורק החל מחלוקה ל9 הערכים הקרובים לערך 20 בציר ה-X השתייכו למרכז ייחודי משלהם (0,19) ונצבעו בצבע אחיד (אדום). ניתן לראות זאת בגרף שלהלן, כאשר המרכזים מסומנים בשחור. קוד ניתן לראות כאן

פלט:

```
Out[65]: <matplotlib.collections.PathCollection at 0x216c0ab50c8>
```

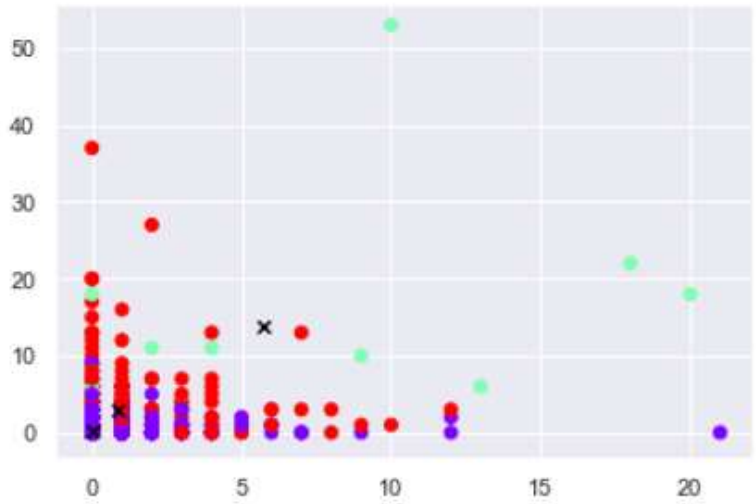


2.2 השוואה בין 10 מאמרים:

2.2.1. על מנת לבדוק אם 10 המאמרים הראשונים עוסקים באותו נושא, עשינו קלסטרינג ל10 מאמרים ראשונים של 1987- הקוד והפלט בקובץ כאן
בדקנו 10 מאמרים ראשונים של 1987 על הקובץ המקורי:

פלט:

```
Out[23]: <matplotlib.collections.PathCollection at 0x21481cae648>
```



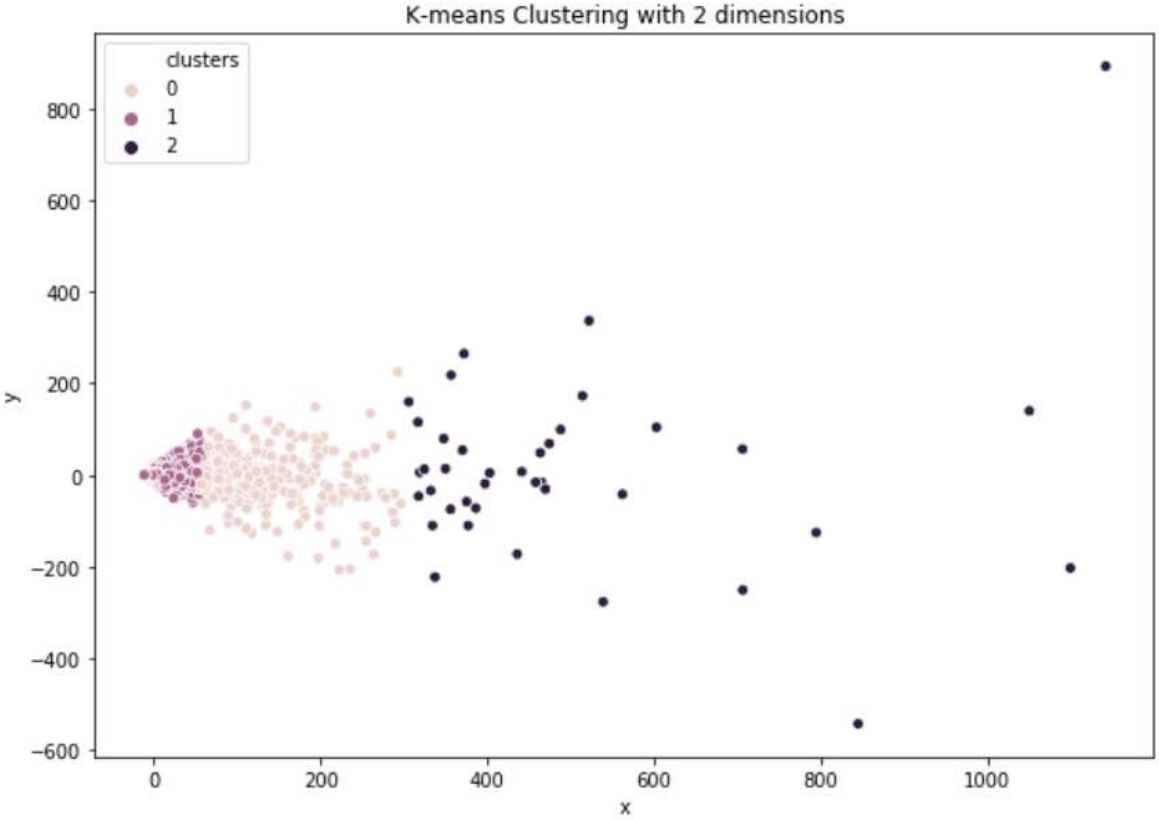
כאן בחרנו לחלק ל3 אשכולות כי המרכזים פחות או יותר מחולקים ומסודרים מהקבוצה עם הצפיפות הכי גדולה (מרחקים קטנים בין הנקודות בקבוצה לקבוצה בעלת הצפיפות הכי קטנה (מרחקים גדולים בין הנקודות).
לפי התוצאה, הגרף הנ"ל אינו מתאים לתצוגה דו ממדית מכיוון שמוצגים בו רק 2 מאמרים בלבד מתוך 10. ניקח לדוגמא את הנקודה הכי קיצונית בציר X והיא המילה learning - שמופיעה 53 פעמים במאמר 2 (1987_2) ו- 10 פעמים במאמר 1 (1987_1) וניתן לראות שאין ייצוג לאותה מילה במאמר 8 (1987_8) המופיעה 69 פעמים ואינה מוצגת בגרף היות וזה אינו גרף רב ממדי. לכן, גרף זה אינו מדויק היות ו-Kmeans שלמדנו מציג גרף דו ממדי ולכן נעשה שימוש באלגוריתם PCA עבור גרף רב מימדי, כאשר באמצעותו נעשתה השוואה כוללת על **כל** המאמרים בקובץ. (כפי שנראה בסעיף הבא)

PCA הוא הקיצור של ניתוח גורמים ראשיים (Principal Components Analysis). מדובר בשיטה סטטיסטית להתמרה ליניארית של נתונים למערכת קואורדינטות חדשה שבה המידע בקואורדינטות השונות הוא אורתוגונלי ובעל שונות הולכת וקטנה. שימוש נפוץ בשיטה הוא לצורך מציאת ייצוג מממד נמוך למידע מממד גבוה (תהליך הנקרא הורדת ממד). שיטה זו הומצאה בשנת 1901 על ידי קרל פירסון והיא שימושית מאוד בסטטיסטיקה ובלמידת מכונה. (ההגדרה לקוחה מויקיפדיה)

2.2.2.קלסטרינג עבור כל המאמרים - ראה קוד מצורף כאן

לאחר שהרצנו את הקוד התקבל הגרף שלהלן, המציג את עמודות המאמרים (מערכים) הרב מימדיים בצורה לינארית.

פלט:



כאן בחרנו לחלק ל3 קלסטרים היות והתקבלה חלוקה סימטרית מבחינת הפיזור של הנקודות. הקלסטר הראשון הוא בעל הצפיפות הגבוהה ביותר. לפי דעתנו הסיבה לכך היא שרוב המילים לא מופיעות או מופיעות ממש קצת במאמרים ולכן הן מתרכזות סביב הנקודה (0,0). כלומר רוב המילים לא שייכות ישירות לנושאים של המאמרים. בקלסטר השני יש ערכים בערך בין 100 ל-250 בציר ה-X והצפיפות היא פחותה יותר מהקלסטר הראשון. לרוב אלו מילים שלא שייכות ישירות לנושא אלא מילים שמשתמשים בהם הרבה במאמרים כמו too, is וכיוצא באלו. הקלסטר השלישי מהערך 250 ומעלה בקירוב בציר הX הוא בעל הצפיפות הנמוכה ביותר. אלו לרוב מילים הקשורות לנושאים בקשר ישיר. ישנם מילים שהערך שלהם בציר ה -Y הוא 0 והסיבה לכך לדעתנו היא שהמילה שייכת לנושא של מאמר ספציפי אבל אינה שייכת לנושאים במאמרים אחרים. מהערך 1000 שבציר הX ומעלה ניתן לראות שתי נקודות שחורות וחריגות המתייחסות למילים הנפוצות ביותר במאמרים מסויימים, אך היות ובציר הY הן לא נמצאות על ציר ה-0 הן יחסית פחות נפוצות במאמרים אחרים. ולעומת זאת אנחנו מסיקות על הנקודה השחורה בפינה הימנית שלמעלה בגרף שהיא הנקודה החריגה ביותר שנפוצה בכל המאמרים היות שהערכים שלה בשני הצירים גבוהים מאוד. דוגמאות למילים כאלו הן כדוגמת learning ו-model הקשורות באופן ישיר לנושא של למידת מכונה ומשתמשים בהן הרבה במאמרים ומחקרים כמו שהראנו בסעיף 1.

3. <u>הסקה על נושא מאמר על סמך המילים המופיעות בו</u>

תחילה נבדוק אילו מילים מופיעות הכי הרבה במאמר ולפי זה נסיק על הנושא שלו. בחרנו במאמר 2-1987 והוצאנו פלט עבור המילים המופיעות בו הכי הרבה פעמים.
לשם כך נעשה שימוש בלולאה המדפיסה את המילים המופיעות מעל 15 פעמים. את מיקומן וכמות הפעמים שהופיעו (ראה קוד בגיטהב ובקישור כאן)

הפלט:

```
The index is: 5593 currant max number: 53 word: learning
The index is: 6771 currant max number: 37 word: nodes
The index is: 10255 currant max number: 27 word: symptoms
The index is: 6699 currant max number: 22 word: netw
The index is: 3722 currant max number: 20 word: fig
The index is: 8462 currant max number: 20 word: reid
The index is: 6718 currant max number: 18 word: neuromodulation
The index is: 11292 currant max number: 18 word: wedge
The index is: 7903 currant max number: 17 word: problem
The index is: 388 currant max number: 16 word: analog
```

ניתן לראות לפי המילים שהתקבלו שהמאמר שייך לתחומי הלמידת מכונה, רשתות ואלקטרוניקה ואכן המאמר Stochastic Learning Networks and their Electronic Implementation עוסק בתחום זה.

4. <u>דמיון בין מאמרים לפי TF-IDF</u>

א. נבדוק 2 מאמרים ראשונים של שנת 1987 (1_1987 + 2_1987) אם הם עוסקים **בנושאים דומים** תוך סינון 1,000 מילים ראשונות. ניתן לראות קוד ופלט כאן.

המאמרים הם רשימה של אובייקטים pd.Series וכל אחד מהם מייצג התפלגות שכיחויות של מילים במסמך. נעשה שימוש ב- sklearn.feature_extraction.text.TfidfVectorizer כדי להמיר את הערכים במסמכים לערכי TF-IDF. (התעלמנו מסדר המילים מהסיבה שרק התדירות חשובה עבור TF-IDF)
ניתן לראות בפלט בעמודה השמאלית ביותר את מספר המאמר שאליו שייכת המילה ובעמודה האמצעית את מספר המילה באוצר המילים שהתקבל.
חישוב TF-IDF התבצע בעזרת TfidfVectorizer של ספריית sklearn. חושב מרחק בין שני המאמרים שהוא 0.48. תוצאה שמצביעה על כך שיש דמיון קטן בין המאמרים.
בנוסף בדקנו מילים שכיחות בשני המאמרים וניתן לראות שיש 2 קבוצות של מילים שערך ה-IDF הוא 0:
קבוצה ראשונה מורכבת ממילים כגון case, without, work שאלו מילים כלליות שלא תורמות להבנת הנושא.
קבוצה שניה: visual, training, defined, conference - מילים שבעזרתן ניתן לקבל מושג על נושא המאמר.
בנוסף בבדיקה שמנו לב שאין טיפול בהטיות למשל מילים כגון: used = use, value = values ודוגמא נוספת היא term-terms שזה אותה מילה בהטיה אחרת וערך שלה ב IDF הוא שונה וזה מטעה מכיוון שמבחינתנו מדובר באותה מילה.

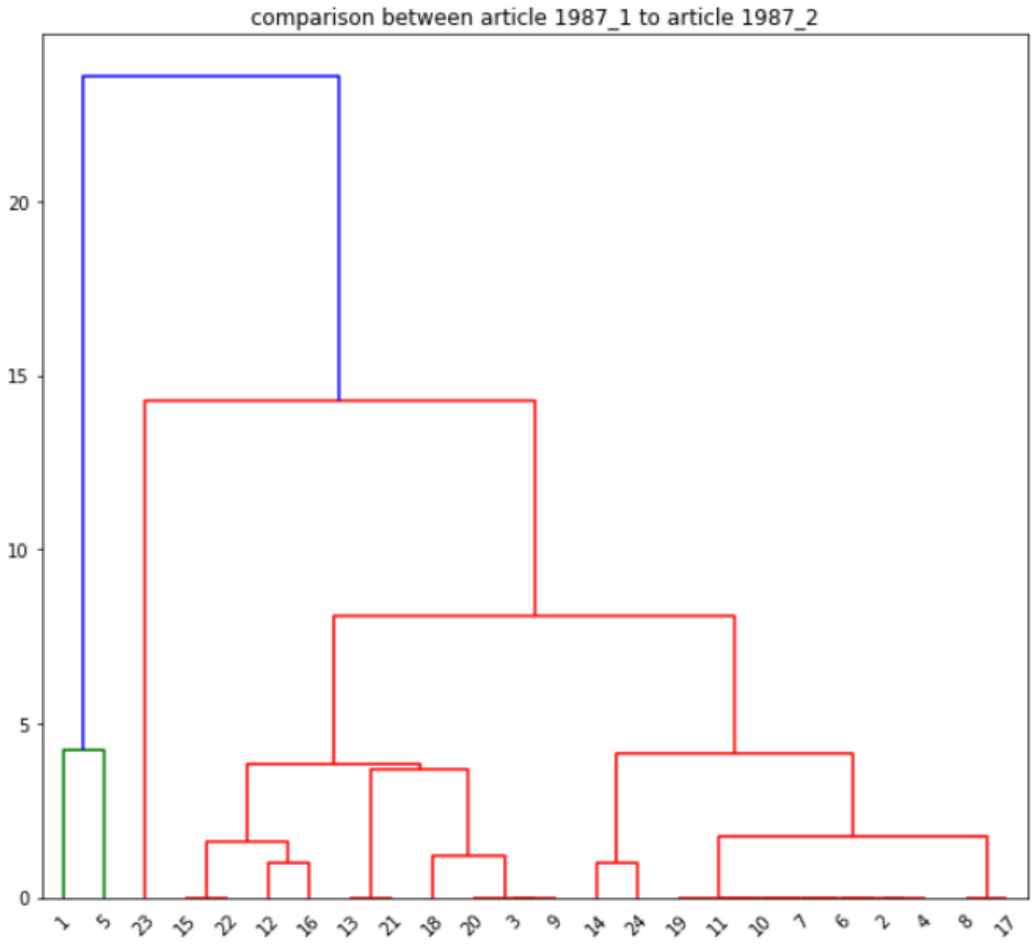
ב. נבדוק 2 מאמרים נוספים, הפעם ניסינו לבדוק דמיון בין **2 מאמרים של אותו מחבר**:
77_1992 ו- 128_1991 ושם המחבר של שניהם הוא Robert C. Williamson. לפי שמות

המאמרים נראה כי הם אף עוסקים בנושאים דומים ולכן גילינו עניין עוד יותר בבדיקת מחבר זה. ניתן לראות קוד ופלט כאן
תוצאות הבדיקה מחזקת את ההשערה שלנו היות והדמיון ביניהם הוא 0.74, תוצאה שמצביעה על כך שהמסמכים דומים. לגבי מילים שכיחות במאמרים אלה ניתן לראות שגם כאן יש מילים שאין להם משמעות או תרומה להבנת הנושא והפוך : מילים כגון approximation, theory , architecture, amount , analog וכיוצא באלו שבעזרתן לא ניתן להסיק מה הנושא של המאמר.

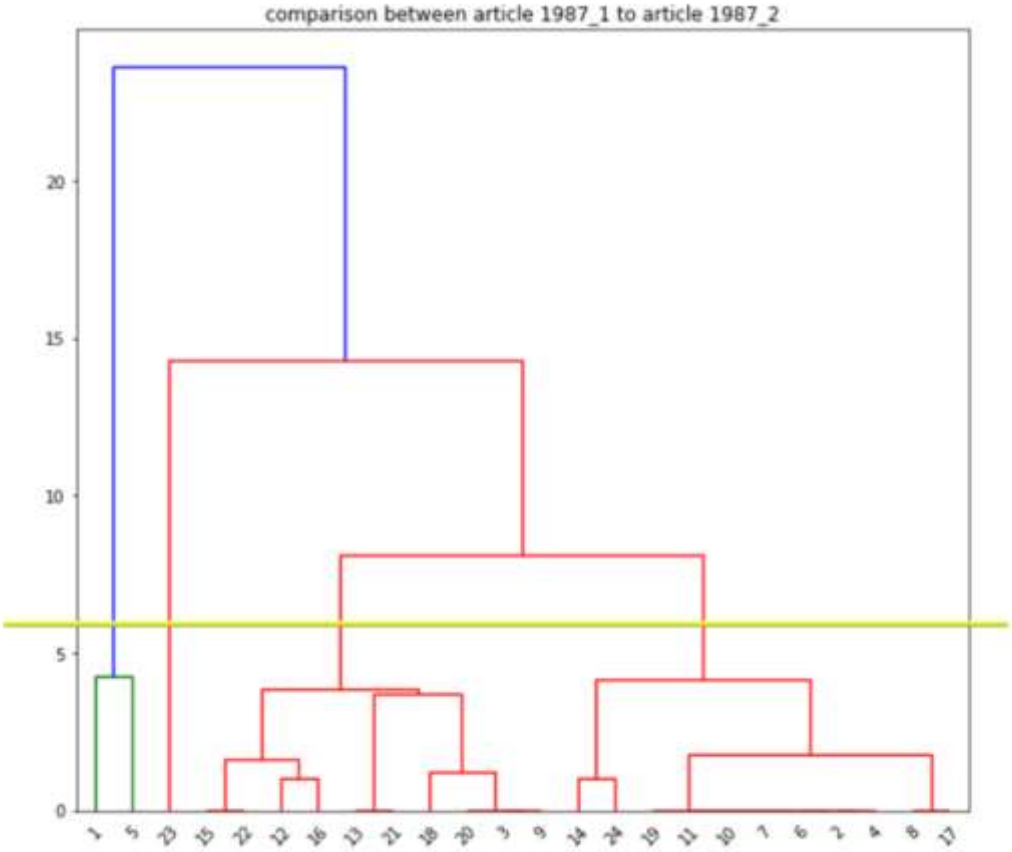
5. <u>דמיון בין מאמרים לפי דיאגרמה היררכית</u>

כאן העברנו את ה2 העמודות של המאמרים של 1987_1 ו1987_2 לקובץ חדש וסיננו לפי מילים שמופיעות בשתיהם. קישור לקובץ כאן.
כעת נראה אם יש קשר בין המאמרים באמצעות דיאגרמה היררכית. ניתן לראות את הקוד והפלט כאן

הפלט של הדיאגרמה :



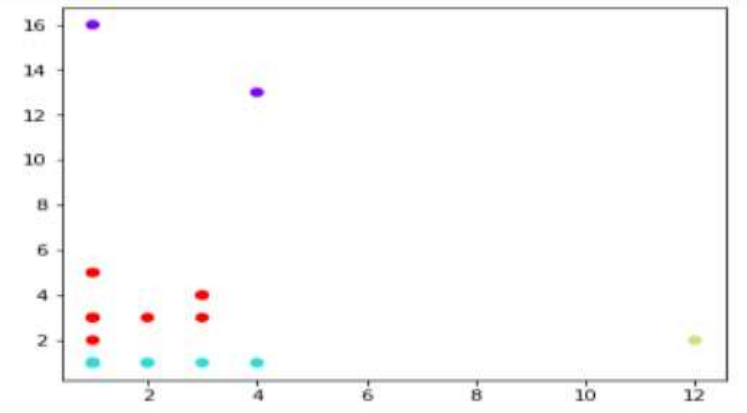
בעזרת הדיאגרמה ההיררכית נמצא את המילים שבין הערכים שלהם המרחק האוקלידי הקטן ביותר ומילים עם ערכים שיש להם את המרחק האוקלידי הגדול ביותר. ניתן לראות למשל המילים become, becomes, architecture, amount, appears, although כמות הפעמים שהן מופיעות ב2 המאמרים הן בכמות שווה לכן המרחק ביניהן הוא הקצר ביותר שהוא 0. ניתן לראות שאלו הן מילים כלליות מאוד ולא שייכות לנושא באופן ישיר ולכן מופיעות יחסית מעט והצבע שלהן אדום.
מילים שנמצאות בקבוצה אחרת מופיעות הרבה יותר פעמים ב2 המאמרים והמרחק האוקלידי שלהם קטן יחסית הן לדוגמא המילים also ו-analog שהמרחק האוקלידי שלהם הוא 4.24 (בצבע ירוק). המילה Analog קשורה לנושא בעיקר של המאמר השני והמילה also אמנם לא קשורה ישירות לנושא אך זאת מילה כללית שחוזרת הרבה במשפטים ומכאן שהיא תופיע הרבה בכל מאמר שהוא.
לעומת זאת ישנן מילים עם ערכים שהמרחק האוקלידי ביניהן גדול יחסית כמו למשל connectivity ו- amount שהמרחק ביניהן הוא 11.04, ולכן הקבוצות שהמילים נמצאות בהן מרוחקות מאוד.
הצבע הכחול מראה שהמילה connectivity שקשורה בעיקר לנושא של המאמר הראשון דומה למילים also ו-analog מבחינת המרחק האוקלידי הקטן ביניהם ואז בהדרגה ל-biological שהמרחק האוקלידי כבר גדול יותר וכיוצא באלה.
כאשר נעביר קו אופקי העובר במרחק הארוך ביותר ללא קו אופקי בערך 6 שעל ציר ה Y אז החלוקה המתקבלת היא ל4 אשכולות.



פלט של החלוקה של הערכים ל4 אשכולות:
array [1, 0, 1, 3, 1, 0, 1, 1, 1, 3, 1, 1, 3, 3, 1, 3, 3, 1, 3, 1, 3, 3, 3, 2, 1], dtype=int64

מהפלט ניתן לראות שרוב הערכים משתייכים לקבוצות 1 ו3 ומדובר במילים כלליות שלא קשורות ישירות לנושא.

לאחר מכן יצרנו Clustering לערכים:



ניתן לראות שרוב הערכים מתרכזים סביב הערכים 4 ו-5 בצירי הX ו-Y ודומים זה לזה, אנו מסיקות שאלו מילים כלליות שמופיעות ב2 המאמרים ושייכות לקבוצות 1 ו-3. בקבוצה 0 ישנם רק 2 ערכים, המילים בהן מופיעות הרבה במאמר השני אך מעט במאמר הראשון. בקבוצה 2 ישנו רק ערך אחד, המילה מופיעה הרבה במאמר הראשון, אך מעט במאמר השני. מכאן ניתן להסיק שהמאמרים שונים זה מזה וראינו בסעיף 4 שהם אכן שונים.

6. <u>מציאת חוקי אסוציאציה בין מאמרים באמצעות אפריורי</u>

6.1. בשלב הראשון לקחנו 30 מאמרים משנת 2015, את הקובץ סידרנו באקסל על ידי שימוש בנוסחה ו Transpose מכיוון שהאלגוריתם עובד על מבנה שונה של מהקובץ מקור. בהתחלה עשינו Transpose לגיליון חדש ומשם הפעלנו נוסחה אשר החליפה בין כמות הפעמים שהמילה הופיעה למילה עצמה והיא חזרה על עצמה בקובץ מספר פעמים שהיה רשום שם קודם לכן.
הנוסחה:

```
=IF(test!B2>0,test!B$1,"")
```

כלומר העמודות הן המילים והשורות הן המאמרים. (קישור לקובץ reversed_2015 כאן).
לאחר הגדרת פרמטרים בנוסחה של אפריורי בדומה במה שהוגדר בכיתה ובקובץ FPM שבמודל ולאחר הרצה של הקוד לא התקבלו חוקים ולכן הוחלט לשנות את הערכים של הפרמטרים: (הרצנו על עמודות 75-90):

```
min_confidence=0.030,  association_rules = apriori(records, min_support=0.0045,
min_lift=3, min_length= 2)
```

לאחר הרצה קיבלנו שלושה חוקים, ניתן לראות שהחוקים שקיבלנו מראים חוקים הקשורים להטיות. האלגוריתם אינו מתחשב בהטיות ומתייחס למילים כשונות אחת משניה (יחיד לעומת רבים) וכמו כן, החוקים אינם אינפורמטיביים.
דוגמא לחוק אחד: (שאר החוקים נמצאים בGitHub בקישור כאן)

```
RelationRecord(items=frozenset({'acknowledge', 'acknowledges'}),
support=0.034482758620689655,
ordered_statistics=[OrderedStatistic(items_base=frozenset({'acknowledge'}),
items_add=frozenset({'acknowledges'}), confidence=0.3333333333333333,
lift=4.833333333333333),
```

OrderedStatistic(items_base=frozenset({'acknowledges'}),items_add=frozenset({'acknowledge'}), confidence=0.5, lift=4.833333333333333)])

6.2. בשלב השני הוחלט לשנות את הערכים של הפרמטרים כדי לקבל חוקים יותר אינפורמטיביים והפרמטר הראשון ששונה הוא min_support - הורדנו את השכיחות של הסל ל-0.002 מ-0.0045 ועדיין התקבלו אותן תוצאות גם כאשר העלנו את הערך. לכן הוחלט לעבור לפרמטר הבא - min_confidence וגם השינוי שלו לא הביא לחוקים אחרים ולכן הוחלט לשנות את העמודות שבדקנו בקובץ במקום 75 עד 90 ל 1000-1050 (מילים אחרות) וקיבלנו 3 חוקים שונים והתוצאות לדעתנו אף יותר מעניינות, לדוגמא:

RelationRecord(items=frozenset({'binary', 'binomial'}), support=0.034482758620689655, ordered_statistics=[OrderedStatistic(items_base=frozenset({'binary'}), items_add=frozenset({'binomial'}), confidence=0.19999999999999998, lift=5.8), OrderedStatistic(items_base=frozenset({'binomial'}), items_add=frozenset({'binary'}), confidence=1.0, lift=5.8)])

ניתוח תוצאה:
רמת הביטחון של החוק היא 20% וזה אומר שבמאמרים שמכילים את המילה binary, כ-20% מתוכם מכילים גם את המילה binomial. מדד היחס (lift) מראה שהסיכוי שהמילה binomial תופיע 5.8 פעמים יותר כאשר במאמר נמצאת המילה binary. ניתן גם לראות כי מדובר באותו תחום. ניתן לראות את יתר התוצאות בGitHub ובקישור כאן

6.3. באקסל חושב סך הכל לכל המילים בקובץ (סך הכל לשורה) על ידי פונקצית sum. שלב הבא עשינו מיון מהמספר הכי גדול לקטן. החלטנו לקחת את ה100 המילים הראשונות מהרשימה. (קישור לקובץ reversed כאן) הרצנו את האלגוריתם של אפריורי על הקובץ הזה. המטרה הייתה לראות חוקים בעלי חשיבות רבה יותר מחוקים שקיבלנו בסעיף מספר 5.2:
כאשר הרצנו עם min_support ו - min_confidance יחסית נמוכים (0.0045 ו- 0.002 בהתאמה) יצאו תוצאות רבות כ- 15005 חוקים ולכן החלטנו לצמצם אותם ולהעלות את הערכים ל0.2 ו0.05 בהתאמה וכעת קיבלנו 4 חוקים שונים (ניתן לראות אותם בקישור כאן) לדוגמא:
RelationRecord(items=frozenset({'also', 'results', 'distribution', 'information', 'error', 'neural'}), support=0.20202020202020202, ordered_statistics=[OrderedStatistic(items_base=frozenset({'results', 'distribution', 'information', 'error', 'neural'}), items_add=frozenset({'also'}), confidence=1.0, lift=2.020408163265306)])

ניתוח תוצאה:
רמת הsupport של החוק הזה היא 20%, זאת אומרת שהמילה also חוזרת ב כ-20% מהמאמרים. מילה זו היא לא משמעותית יחסית אבל ניתן לראות מהתוצאות שבדרך כלל אחריה יבואו הסברים על רשתות נוירוניות בתחום של למידת מכונה שכוללות בתוכן נושאים סטטיסטיים המכילים בתוכן מילים כגון: התפלגויות, תוצאות, שגיאות לפי התוצאה ניתן לראות שרמת הבטחון גבוהה מאוד (100%). מכאן ניתן להסיק שבכל המאמרים בהם תופיע המילה also, ב- 100% תבוא אחריה גם המילה results, distribution, information, error והמילה neural. לפי מדד הlift שיצא 2.02 ניתן להסיק שמילים כמו error, results או neural ואחרים שפורטו למעלה, הסבירות שיהיו באותו מאמר יחד עם המילה also הוא פי 2.02 פעמים.
ניתן לראות את שאר התוצאות בGitHub ובקישור כאן

דיון

- למדנו מהתוצאות שלפעמים נדרשנו לעשות מספר מניפולציות כדי לקבל תוצאות רצויות (כמו לדוגמא בסעיף 2 כשבפעם הראשונה בחרנו K=3 ואז ראינו שהתוצאה פחות מתאימה והחלפנו לK=4 כדי לקבל תוצאה מתאימה יותר או בסעיף 6 באפריורי לקבלת תוצאות רצויות וחוקים משמעותיים.
- למדנו שבחלק מהמקרים הטיות אינן נלקחות בחשבון (כמו בסעיף 4 כשבדקנו את הדמיון בין המאמר\מחבר או בחיפוש אחר חוקים בסעיף האחרון).
- דבר נוסף הוא שניסינו לבנות את הפונקציה של TF-IDF בצורה ידנית על מנת להבין כיצד תתי שלבים של האלגוריתם עובדים על מנת שנוכל לנתח את התוצאות הסופיות וגם כי לא תמיד האלגוריתם שאנו מכירות או הדוגמא שלמדנו רצה על אותו מבנה נתונים (ניתן לראות בGitHub בקישור כאן)
- למדנו שרוב העבודה היא בעצם ניקוי והכנת קובץ להרצת אלגוריתמים שלמדנו בכיתה ושניתן למצוא ברשת. הקושי המרכזי הוא להתאים את הנתונים כך שנוכל לעבוד איתם. בעצם שלב ההכנה הוא ארוך וקשה יותר וכמובן הכי חשוב כי הכול תלוי בו (ההתקדמות והתוצאה).
- בתחילת העבודה עבדנו ב- Visual Studio והיה קשה ולא נוח לקרוא את הנתונים. בנוסף לכך, לא היה ברור איך נוכל לראות פלטים צבעוניים וברורים כפי שראינו ברשת או בדוגמאות מהשיעור. לכן לאחר בדיקה ברשת עברנו ל- Jupyter Notebook וגילינו שמדובר בכלי מאוד נוח ותומך בוויזואליזציה של נתונים.
- במהלך העבודה ראינו שאנחנו עוברות את המספר המקסימלי של העמודים ולכן החלטנו לשים את הקבצי קוד והפלטים שלהם בGitHub.
- היו הרבה אפשרויות לניתוחים היות ומדובר בקובץ ביג דאטה ונאלצנו כל פעם לחתוך חלק אחר ולהחליט מה הניתוח עבור כל חלק ולא תמיד ההחלטה שקיבלנו הביאה לתוצאות שקיוונו אליהן. לפעמים אפילו נעשתה בחירה של דברים ידועים מראש או ברירת מחדל התחלתית או דוגמאות מהכיתה (לדוגמא בסעיף האחרון) על מנת להבין את הפלט או לבדוק את האלגוריתם ושאנחנו עושות את הפעולות הנכונות כדי להימנע מטעויות (לדוגמא סעיף 4 בו בדקנו מאמרים משנים שונות של אותו מחבר).
- מכיוון שיש כמות גדולה של מאמרים, בכל פעם החלטנו איזה מאמרים עדיף לקחת גם מבחינת השיטה שעשינו שימוש בה וגם מבחינת התוצאות שהתקבלו שעבור חלק מהמאמרים היו רלוונטיות ועבור חלק אחר לא. כלומר הרבה **ניסוי וטעיה**.
- לפעמים איפה שציפינו שתהיינה תוצאות גבוהות קיבלנו תוצאות נמוכות ממה שחשבנו וההפך.
- בבחינת השימוש בKmeans כשראינו את הפיזור של הנקודות בהתחלה לא היה מובן לנו עד הסוף מה התוצאה אומרת, אבל בעזרת אקסל כאשר דגמנו מילים בכל קלסטר, זה עזר לנו להבין מה הקשר בין המילים או למה הן נמצאות באותה קבוצה וכך בעצם זה עזר לנו להבין עוד יותר כיצד לקרוא את תוצאות הגרפים.
- בכל העבודה לא היה מספיק להריץ קוד על הנתונים שבחרנו לאחר חשיבה, אלא לאחר מכן גם נדרש תהליך ניתוח שכלל הרצות נוספות כדי להבין איך התוצאות מסבירות את הנתונים בקובץ ובדיקה מול הקובץ האם אכן זה המצב.
- באפריורי בסעיף 4 נתקלנו בכמה קשיים במהלך העבודה. לאחר הרצת האלגוריתם היינו מקבלות חוקים "ריקים", כלומר בכל חוק הינו מקבלות מילה Nan וכל פעם היינו צריכות לעשות החלטה על איזה עמודות, שורות ומילים להפעיל את האלגוריתם
- היישום בכלים שלמדנו במהלך הקורס העשיר אותנו מאוד ואף עודד אותנו למצוא ולהתעניין בכלים נוספים חדשים כמו דוגמת הPCA עבור גרף רב מימדי, בעיקר לאור העובדה שהשיטה הזאת מתאימה יותר היות ו-Kmeans שלמדנו מציג גרף דו ממדי. בעזרת אלגוריתם PCA הראנו הצגה עבור עמודות המאמרים שאלו הם מערכים רב מימדיים בצורה לינארית.

- סך הכל נהננו מאוד לעשות את הפרוייקט ושמחות על כך שכלים אלה יעזרו לנו כעת ובעתיד לנתח באופן הרבה יותר טוב נתונים ובעיקר נתוני עתק וגם לייעל את עבודתנו במקומות העבודה, וכמו כן תרמו רבות לחשיבה האנליטית שלנו מנקודת מבט אחרת