נאור סמואלוב 307939447
אימרי אוזן 209005909

בס"ד

ביולוגיה חישובית – דו"ח תרגיל 3:

איך להריץ את התרגיל?

להיכנס לקישור: https://github.com/ImryUzan/Ex3ComputationalBiology ולהוריד את קובץ הריצה bio3.exe . בנוסף, העלנו לגיט את קובץ האקסל Elec_24.csv כפי שמופיע בתרגיל. אחרי שהורדנו את הקבצים מהגיט (לחילופין, ניתן לעשות clone לגיט), נשים את קובץ הexe ואת קובץ הקלט (למשל Elec_24.csv) באותה התיקיה. נפתח cmd בתיקיה ונריץ את הפקודה:

```
C:\Users\uzan2\PycharmProjects\merkle\dist>bio3.exe C:\Users\uzan2\PycharmProjects\merkle\dist\Elec_24.csv
iter: 0
iter: 1
iter: 2
iter: 3
iter: 4
iter: 5
iter: 6
```

הפרמטר הראשון בפקודה הוא שם קובץ הריצה bio3.exe והפרמטר השני הוא **הנתיב המלא** במחשב עד לקובץ Elec_24.csv (במידה ונרצה להריץ עם קובץ אחר, יש לשנות את שמו להיות Elec_24.csv ולשים אותו בתיקיה עם הexe). לאחר הרצת הפקודה (זמן ריצה מוערך של התרגיל: כ2 דקות), מודפסות האיטרציות (כחיווי למשתמש על התקדמות התרגיל, אפשר לראות בצילום המסך) . הפלט שיתקבל מריצת התרגיל הוא ארבעת הגרפים - נפרט עליהם בהמשך הדו"ח ועבור כל עיר את האינדקס של הנוירון שמייצג אותה (הקרוב ביותר אליה). דוגמא לפלט:

```
city: Fasuta  --->  index: (3 , 7)
city: Ashdod  --->  index: (5 , 1)
city: Kasrasmie  --->  index: (4 , 0)
city: Raanana  --->  index: (0 , 3)
city: Lehavim  --->  index: (0 , 4)
city: Carmiel  --->  index: (1 , 1)
city: Givatayim  --->  index: (0 , 4)
city: Bueeninogidat  --->  index: (8 , 3)
city: Baana  --->  index: (6 , 6)
```

תיאור האלגוריתם:

חישוב מרחקים: לאורך האלגוריתם שבנינו נדרשנו לחשב מרחקים בין וקטורים. את המרחק הזה חישבנו בעזר RMS ($\sqrt{\sum_{i=1}^{|V|}(V_i - N_i)^2}$). כלומר, בנינו פונקציה שבהינתן שני וקטורים, סוכמת את תוצאת החיסור בחזקת שתיים של התאים במקומות הזהים של שני הוקטורים ולבסוף מבצעת שורש ריבועי. חשוב לציין כי על מנת שמדידת המרחקים תהיה מיטבית, נרמלנו כל שורה ב DATA על ידי חילוק כל עמודה בערך של העמודה "Total Vote" פרט לעמודה "Economic Cluster" אותה חילקנו ב 10 (הערך הגבוה ביותר שהעמודה הזו מקבלת). בנוסף, ערכי המספרים שהגרלנו עבור יצירת הרשת הרנדומאלית בתחילת החישוב היו בין 0 ל 1. כך קיבלנו וקטורים באותו טווח ערכים לאורך כל ריצת האלגוריתם.

קירוב התאים: כחלק מאלגוריתם ה – SOM היינו צריכים לבחור לכל שורה ב DATA את הנוירון הקרוב ביותר אליה (תוך שימוש בפונקציית המרחק שתיארתי לעיל) ולאחר מכן לבצע קירוב של הנוירונים המתאימים לשורה הזו. נוסחת הקירוב בה השתמשנו היא:

$$N_k(t+1) = N_k(t) + \alpha(t) \cdot h_{ki}(t) \cdot [V_i - N_k(t)]$$

ערך הנוירון לאחר העדכון

ערך הנוירון לפני העדכון

Learning rate = 0.1

ערך שתלוי ברמת הקרבה של הנוירון לנוירון הקרוב ביותר לשורה הנבדקת.

רמה 1: 0.5

רמה 2: 0.25

רמה 3: 0.125

הוקטור של השורה שנבחרה ב DATA

ערך הנוירון לפני העדכון

לאחר שבחרנו את הנוירון הקרוב ביותר לשורה ב DATA , עדכנו את הנוירון עצמו (כך ש h = 0.5), את כל שכניו (כך ש h=0.25) ואת כל שכניו בדרגה השניה (כך ש h=0.125). על מנת לבחור את ההיפר פרמטרים הללו, בנינו script פייטון קצר שניסה קומבינציות רבות של ההיפר פרמטרים הללו (כולל גם את ה learning rate) ולבסוף החזיר את הקומבינציה הטובה ביותר שמצא. הפתרון הטוב ביותר נבחר על ידי חישוב ממוצע המרחקים של כל שורה ב DATA לנוירון אליו היא ממופת עבור כל אחד מהפתרונות שהציגו ההיפר פרמטרים השונים, ובחירת ההיפר פרמטרים שהשיגו את הממוצע הנמוך ביותר. חשוב לציין שכחלק מהניסיונות שלנו ניסינו גם להשתמש בטכניקה של הקטנת ה learning rate ככל שמתקדמים באיטרציות אבל נוכחנו לגלות שכאשר ה learning rate קטן יחסית כבר מהאיטרציה הראשונה, האלגוריתם שלנו השיג תוצאות טובות יותר.

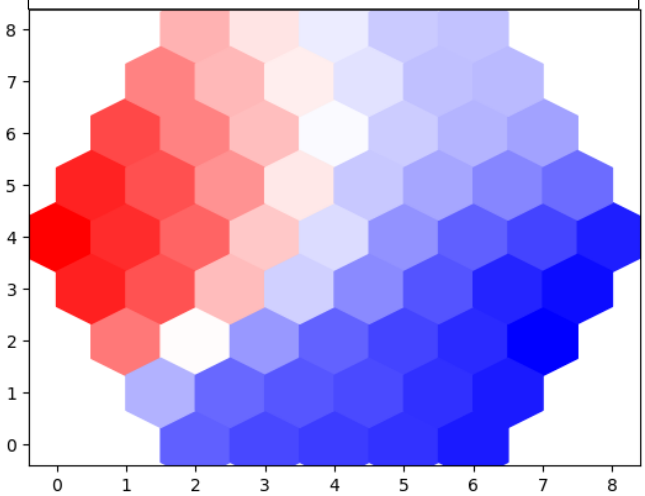
סדר השורות: כחלק ממימוש האלגוריתם שלנו, בנינו פונקציה שמריצה את אלגוריתם ה SOM 15 פעמים ולבסוף בוחרת את התוצאה הטובה ביותר (כפי שמצוין בהמשך הדו"ח). על מנת להגדיל את הרנדומיות של האלגוריתם, לפני כל איטרציה ביצענו עירבוב של השורות ב DATA כך שאין השפעה לסדר השורות כפי שהן מופיעות ב data set עצמו.

בחירת הפתרון: את בחירת הפתרון ביצענו בשני שלבים: תנאי עצירה ובחירת הפתרון הטוב ביותר.

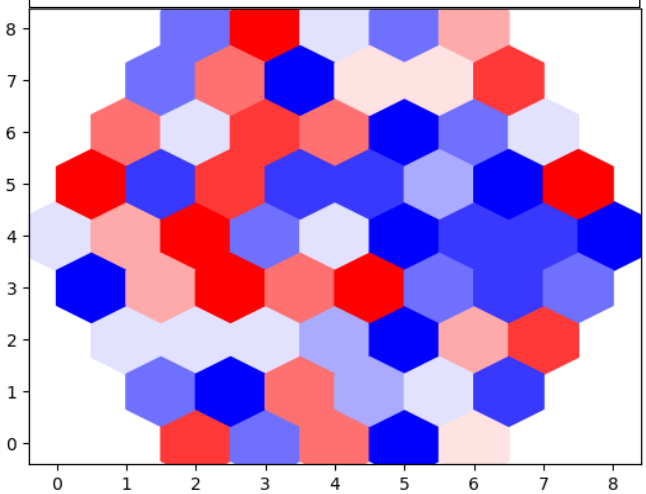
- <u>תנאי עצירה:</u> כפי שתיארנו כבר לעיל, האלגוריתם שלנו מריץ 15 פעמים את אלגוריתם ה SOM ובוחר את הפתרון הטוב ביותר. בכל איטרציה הוא רץ בלולאת for מ 0 ועד 300 כך שבכל איטרציה הוא עובר על כל ה DATA ומעדכן את הנוירונים המתאימים. בתוך לולאת ה for הזו יש תנאי עצירה שבודק אם לפחות 90% מהנוירונים מקיימים שכל שורה ב DATA ממופה לאותו נוירון בשתי האיטרציות האחרונות וגם עבור כל שורה שהמרחק בין הנוירון הקרוב ביותר אליה לנוירון השני הכי קרוב אליה הוא 1 (כלומר שהם שכנים) וגם מספר האיטרציות גדול מ 70 אז מתבצע BREAK. הסיבה שהוספנו את התנאי היא שמספר האיטרציות חייב להיות גדול מ 70 זה כיוון שראינו שבדרך כלל ההתכנסות קוראת לפני האיטרציה מספר 70 ובנוסף ראינו שכיוון שאנחנו מאתחלים את הנוירונים באופן רנדומאלי, אז מבחינה הסתברותית יכול להיות שבאיטרציות הראשונות שני התנאים האחרים יתקיימו עוד לפני ההתכנסות ורצינו למנוע מצב שכזה.
- <u>בחירת הפתרון:</u> את בחירת הפתרון הטוב ביותר הצענו על ידי חישוב ממוצע המרחקים של כל שורה ב DATA לנוירון אליו היא ממופת עבור כל אחד מהפתרונות השונים, ובחירת זה שהשיג את הממוצע הנמוך ביותר. חשוב לציין שכיוון שתנאי העצירה מכיל את בדיקת ממוצע המרחקים בין הנוירונים הקרובים ביותר לשניים הכי קרובים, וכיוון שבחלק גדול מהפתרונות של 15 האיטרציות 100% מהנוירונים מקיימים את התנאי שהמרחק הזה הוא 1, בחרנו לא להתייחס אליו בבחירת הפתרון הטוב ביותר.

בס"ד

תצוגה גרפית שמראה את ערכי המעמד הסוציואקונומי בנוירונים, לאחר ריצת האלגוריתם - ככל שהגוון אדום יותר, המעמד הסוציואקונומי גדול יותר ככל שהגוון כחול יותר המעמד הסוציואקונומי נמוך יותר.

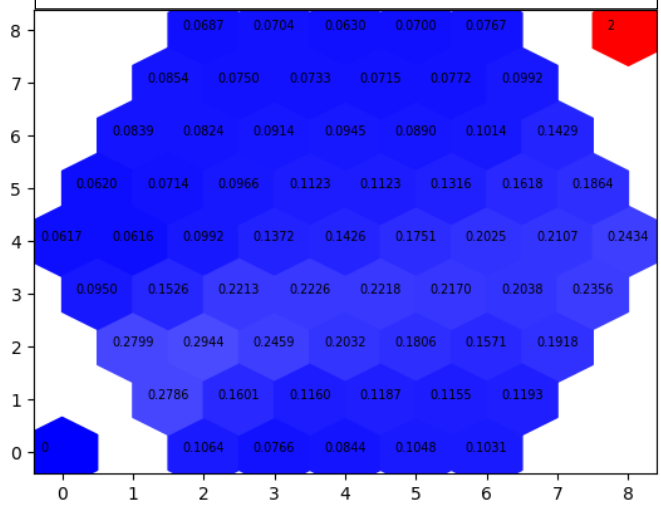


תצוגה גרפית שמראה את ערכי המעמד הסוציואקונומי בנוירונים, לפני ריצת האלגוריתם (בתחילת האלגוריתם הרשת מאותחלת רנדומאלית)- ככל שהגוון אדום יותר, המעמד הסוציואקונומי גדול יותר ככל שהגוון כחול יותר המעמד הסוציואקונומי נמוך יותר.

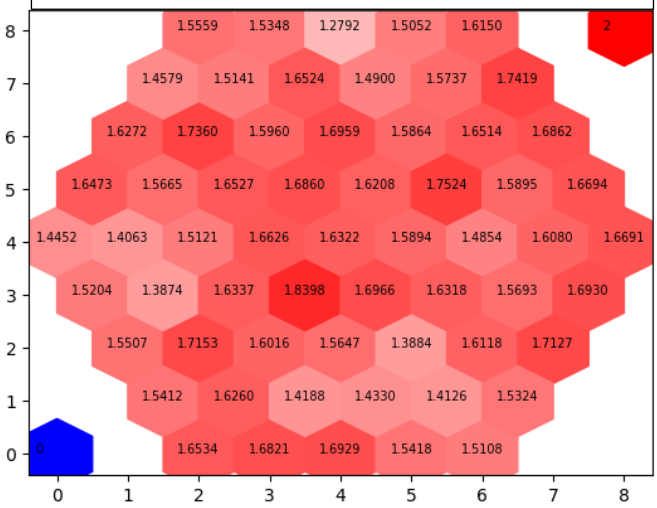


ניתן לראות שלפני ריצת האלגוריתם, הפיזור הוא ראנדומאלי כלומר הערכים של נוירונים (המעמד הסוציואקונומי) באזורים קרובים ברשת אינם דומים זה לזה לכן וקטורים דומים מה data set לא ימופו לנוירונים קרובים (לא מייצג clustering בצורה טובה). לעומת זאת לאחר ריצת האלגוריתם ניתן לראות מגמתיות בצבעים ושגוונים דומים נמצאים סמוך זה לזה מכאן וקטורים דומים מה data set ימופו לנוירונים קרובים (תחת ההנחה שהמעמד הסוציואקונומי מייצג בצורה טובה את הclusters השונים).

תצוגה גרפית שמראה עבור כל נוירון את המרחק הממוצע של הוקטור שלו משכניו, לאחר ריצת האלגוריתם - ככל שהגוון אדום יותר, המרחק גדול יותר ככל שהגוון כחול יותר המרחק קטן יותר.



תצוגה גרפית שמראה עבור כל נוירון את המרחק הממוצע של הוקטור שלו משכניו, לפני ריצת האלגוריתם (בתחילת האלגוריתם הרשת מאותחלת רנדומאלית) - ככל שהגוון אדום יותר, המרחק גדול יותר ככל שהגוון כחול יותר המרחק קטן יותר.



נשים לב: שתי המצולעים בקצוות ההקסואיד (במקומות 0,0 ו 8,8) אינן חלק מהרשת ומטרתם היא לאפשר לטווח הצבעים לייצג את המרחקים כתלות בטווח קבוע ולא את המרחקים כתלות בטווח הערכים העצמיים שלהם.

ניתן לראות שלפני ריצת האלגוריתם, המרחק הממוצע של כל נוירון משכניו הוא גדול יחסית (גוון אדום) מכאן ניתן להסיק שהוקטורים של נוירונים שכנים אינם דומים ומכאן אנו למדים שוקטורים דומים מה data set לא ימופו לנוירונים קרובים ברשת ולכן ניתן להסיק שהרשת הזו אינה ביצעה clustering. לעומת זאת, ניתן לראות שלאחר ריצת האלגוריתם, המרחק הממוצע של כל נוירון משכניו הוא קטן יחסית (גוון כחול) מכאן ניתן להסיק שהוקטורים של נוירונים שכנים דומים זה לזה ומכאן אנו למדים שהרשת ביצעה clustering בצורה טובה.