# מטלה 3 ביולוגיה חישובית – זיהוי תבניות באמצעות רשת נוירונים

**איתי אלקלאי 2060071110, אביאל זכריה 207456641 – תרגיל משולב**

## זיהוי התבניות

על מנת לזהות את התבניות ניסינו לנחש כל מיני לוגיקות שיכולות לאפיין פונקציה בוליאנית שמקבלת כקלט 16 פרמטרים בינאריים, ובהתאם לכתוב סקריפט פייתון שיאמת את זה:

# Read nn1.txt file  
nn1 = open('nn1.txt', 'r').readlines()  
nn1 = [i.split() for i in nn1 if i]

# Making it a binary list, and a Boolean output - ([1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0], 0)  
nn1 = [(list(map(int, i)), bool(int(j))) for i, j in nn1]

# Print unmatched inputs  
print([(i, o) for i, o in l1 if o != logic(i)])

כמובן שהנחנו שאין לנו Outliers, ובנוסף שהחוק הוא הגיוני וניתן להכללה בהנחה שאין לנו את כל מרחב הקלטים האפשרי בעולם (או לפחות מגוון מספיק כדי לזהות את הפונקציה בהכללתה)

### nn1.txt

ניסינו המון אופרטורים בינאריים כמו AND, XOR, OR וכלום לא עבד, ניסינו לחלק את זה לשני חלקים של שמונה ועל זה להפעיל פונקציות וזה גם לא עבד וכו'. לבסוף ניסינו להדפיס hitmap של מספר האחדות אל מול התוצאה הרצויה ושמנו לב לתופעה הבאה:

Print(set([(sum(i), j) for i, j in nn1]))

{(1, 1),  
 (2, 1),  
 (3, 1),  
 (4, 1),  
 (5, 1),  
 (6, 1),  
 (7, 1),  
 (8, 0),  
 (9, 0),  
 (10, 0),  
 (11, 0),  
 (12, 0),  
 (13, 0),  
 (14, 0),  
 (15, 0)}

**כלומר, הפונקציה logic שלנו היא האם מספר האחדות קטן ממש מ-8.**כמובן שיש רשת נוירונים (ואפילו פתרון לינארי) שמוצא את החוקיות ללא שכבה פנימית (וכך גם עשינו).

### nn0.txt

לאחר מכן ביצענו בדיוק את אותו ה-hitmap ל-nn0.txt וקיבלנו תוצאה אחרת מעניינת:

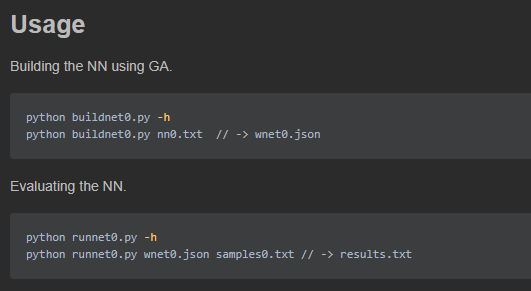
set([(sum(i), j) for i, j in nn0])

{(1, 0),  
 (2, 0),  
 (3, 0),  
 (4, 0),  
 (5, 0),  
 (6, 0),  
 (7, 0),  
 (8, 1),  
 (9, 1),  
 (10, 1),  
 (11, 1),  
 (12, 1),  
 (13, 0),  
 (14, 0),  
 (15, 0),  
 (16, 0)}

**כלומר נראה שמספר האחדות צריך להיות בין 8 ל-12 כולל כדי שהפונקציה תתאמת.**  
כמובן שיש רשת נוירונים שמחשבת את זה ומוצאת את החוקיות אפילו ללא שכבה פנימית (פתרון לינארי).

## הוראות הרצה

כאמור, הקוד ממומש בשפת פייתון וניתן להריץ אותו בפשטות בעזרת תיעוד הנמצא בקובץ ה-README.md המצורף בתרגיל.



## מימוש הקוד

### רשת הנוירונים

הקובץ הראשי שמממש את רשת הנוירונים שלנו הוא neural\_net.py

* מבנה הרשת הנבחר – כאמור, בהתאם לניחוש שלנו לחוק שמתאר את ה-data שקיבלנו נראה שפתרון לינארי יהיה מספק ולכן בחרנו לא לבחור שכבה חבויה ברשת הנוירונים שלנו.  
  כלומר הרשת מורכבת משכבת כניסה בעלת 16 כניסות ושכבת יציאה בעלת יציאה אחת בלבד.
* Layer threshold – בחרנו אותו להיות הקבוע 1 בעזרת כך שהחסרנו את הערך 1 מהווקטור לפני שהכנסנו אותו לתוך פונקציית האקטיבציה.
* פונקציית אקטיבציה – בחרנו ב-sigmoid כסטנדרט לקבלת הסתברות לתשובה ולא תשובה בינארית.
* פונקציית ה-fitness (או loss) – כל טעות בסיווג שילמה מחיר של המרחק מההסתברות האמיתית. המחיר של סך דוגמאות היה ממוצע מחירי הטעויות של הדגימות שניתנו.

### האלגוריתם הגנטי

הקובץ הראשי בו כתובה הלוגיקה של האלגוריתם הגנטי הוא genetic\_algo.py בפונקציה genetic\_train\_nn.

* גודל אוכלוסייה: 300
* מספר סיבובים מקסימלי: 1000
* גבול לעצירה: 98 אחוז הצלחה
* מספר דורות ללא שינוי: 50
* בחירת המובילים: 10 אחוז
* יצירת מוטציות: 40 אחוז
* יצירת crossover: 40 אחוז

### יצירת מוטציות

???

### יצירת crossovers

???

## ביצועים

* על רשת הקלט nn0.txt המודל שלנו wnet0.json השיג בתוצאה הכוללת 98.7 אחוז הצלחה.
* על רשת הקלט nn0.txt המודל שלנו wnet1.json השיג בתוצאה הכוללת X אחוז הצלחה.