

# תרגיל בית 3 – מבוא ללמידה

מגיש: טל רוזנצוויג

ת.ז: 307965806

## שאלה 1:

תוצאת הדיוק שקיבלתי עבור ביצוע אימון האלגוריתם על קבוצת האימון וביצוע בדיקה שלו על קבוצת המבחן היא:

```
(HW1) C:\Users\talro\OneDrive\מיכמסח\IntroToAi\Projects\tal's project>python ID3.py  
0.9469026548672567
```

## שאלה 2:

הוכחה:

על מנת להוכיח את הטענה יש להראות כי העץ הנוצר מקבוצת אימון **לא מנורמלת**, זהה במדויק לעץ הנוצר מקבוצת אימון **מנורמלת**.

טענת עזר: נרמול  $MinMax$  אינו משנה את סדר הגודל של הערכים.

הוכחה: יהיו  $x, y$  ערכים לא מנורמלים כך ש-  $x < y$  - ו-  $x' = \frac{x - \min}{\max - \min}, y' = \frac{y - \min}{\max - \min}$  הערכים

שלהם לאחר נרמול  $Minmax$  בהתאמה.

נשים לב כי  $\max > \min$  ולכן מתקיים  $\max - \min > 0$  ולכן  $x' < y'$ .

בכיוון השני יהיו ערכים מנורמלים  $x' < y'$  ומכיוון שהמכנה שלהם חיובי, ו-  $\min = \text{const}$  אז

מתקיים  $x < y$ .

טענה זו גוררת את הטענה הבאה: ערכים שהיו קטנים, גדולים מערך הסף לפני נרמול יהיו קטנים, גדולים בהתאמה **מערך הסף המנורמל אחרי** הנרמול.

נניח בשלילה כי לאחר נרמול ובניית עץ  $ID3$  על קבוצת האימון **התקבל עץ שונה** מהעץ שנבנה לפני הנרמול, אזי צריכה להיות דוגמת אימון  $e \in \text{train\_data}$  כך שבסיום בניית העץ שייכת לעלה אחר מהעץ שנבנה לפני הנרמול.

מאחר ו- $e$  שייכת לעלה אחר, על פי אלגוריתם בניית  $ID3$  מתקיים כי לפחות באחד מצמתי העץ המפרידים את הדוגמאות לפי תכונות, הדוגמא  $e$  החליפה כיוון – המשמעות היא שערך התכונה שלה החליף סדר ביחס לערך הסף של הצומת בסתירה לכך שנרמול  $MinMax$  משמר סדר.

כעת לאחר שהוכחנו כי עבור קבוצת אימון מנורמלת וקבוצת אימון לא מנורמלת יבנו בדיוק **אותם עצים**, עבור כל קבוצת מבחן נקבל **אותו דיוק**.

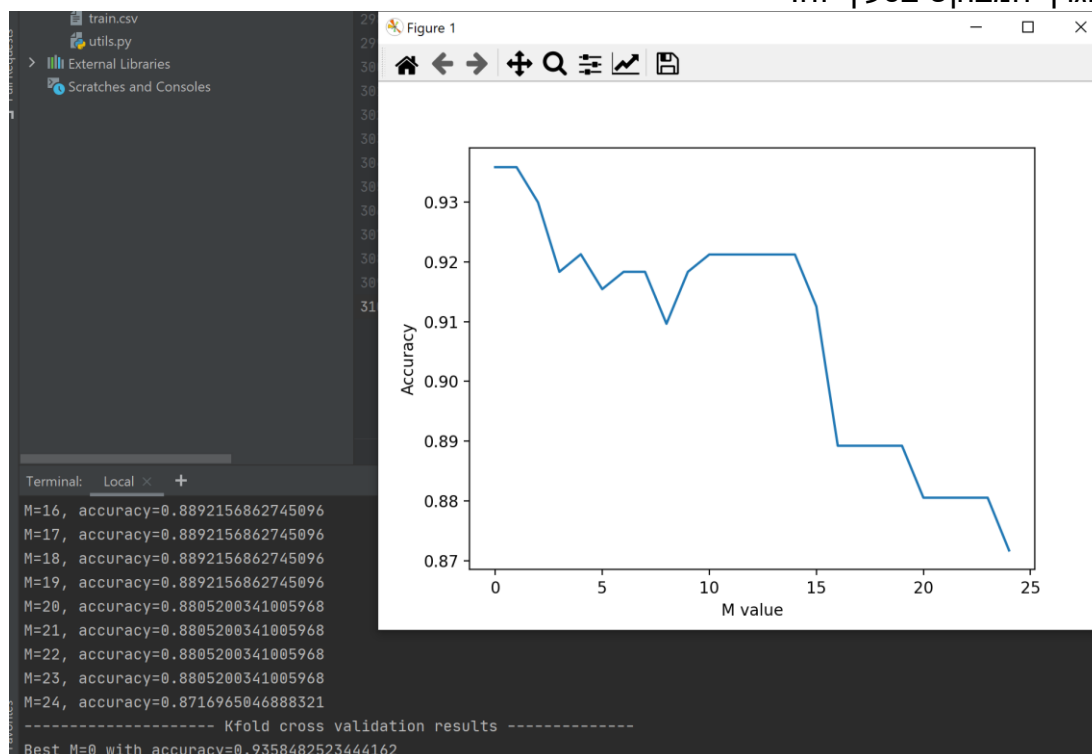
### שאלה 3:

#### סעיף 3.1:

גיזום עצי החלטה נעשה כדי להקטין את העץ ולהחליש את התאמת היתר. הגיזום מחליף חלק מצמתי העץ בעלים עם אבחון לפי הרוב בקבוצת האב, ושיטה זו מונעת התאמת יתר לדוגמאות האימון וגורמת לכך שהמסווג יכול לענות טוב יותר על דוגמאות שלא ראה באימון.

#### סעיף 3.3:

הגרף המבוקש בסעיף זה:



גרף זה מתאר כיצד הפרמטר  $M$  משפיע על הדיוק.

לפי הגרף, ניתן לראות כי עבור  $M = 0$  מתקבל ערך הדיוק הטוב ביותר. כמו כן, ניתן לראות בגרף כי ערכים גדולים של  $M$  גורמים לירידה בדיוק מפני שהם גוזמים ענפים בעלי חשיבות לצורך חישוב תוצאת הדיוק. תופעה זו יכולה להיות מוסברת גם מגודל קבוצת האימון הקטנה שלנו שנעשית קטנה יותר לאחר חלוקה לחלקים ב- $kfold$ .

#### סעיף 3.4:

הגיזום עם  $M = 0$  לא שיפר את הביצועים ביחס להרצה ללא גיזום בשאלה 1 וזאת ניתן לראות בתצלום הבא:

```
C:\Users\talro\miniconda3\envs\HW1\python.exe "C:/Users/talro/OneDrive/תסמכים/IntroToAi/Projects/tal's project/ID3.py" -v
The accuracy of ID3Classifier=0.9469026548672567

C:\Users\talro\miniconda3\envs\HW1\python.exe "C:/Users/talro/OneDrive/תסמכים/IntroToAi/Projects/tal's project/ID3.py" -v
The accuracy of ID3PruneClassifier=0.9469026548672567
```

שמתי לי כי עבור ערכים שונים כן ניתן לקבל דיוק טוב יותר עבור קבוצת המבחן, למשל עבור  $M = 8$  אך אני מבין שערכים אלו מותאמים **לקבוצת המבחן הזו בלבד** וערך זה אינו יתאים לקבוצות אחרות.

## שאלה 4:

### סעיף 4.1:

ערך ה- $loss$  של  $ID3$  הוא 0.0212.

```
C:\Users\talro\miniconda3\envs\HW1\python.exe "C:/Users/talro/OneDrive/תסמכים/IntroToAI/Projects/tal's project/ID3.py" -v
The accuracy of ID3Classifier=0.9469026548672567
loss without cost optimizing=0.021238938053097345
```

### סעיף 4.2:

על מנת לגרום ל- $ID3$  ללמוד מסווג שממזער את פונקציית ה- $loss$  השתמשתי באלגוריתם הבא:

1. חלוקת הדטה לשני חלקים:  $training\ data$  &  $validation\ data$ .
2. בניית עץ  $ID3$  רגיל ולאחר מכן גיזום שלו לפי השלבים הבאים:
3. יורדים לעומק רקורסיה ובמעלה מהעלים לשורש נבקר בכל צומת ונבצע:
  - a. חישוב של  $loss$  על ה- $validation\ data$  שחלחל לצומת הזה בעץ.
  - b. חישוב של  $loss$  על ה- $validation\ data$  שחלחל לצומת הזה בעץ, במידה ונגזום אותו וניתן לו אבחון חיובי.
  - c. חישוב של  $loss$  על ה- $validation\ data$  שחלחל לצומת הזה בעץ, במידה ונגזום אותו וניתן לו אבחון שלילי.
  - d. נבדוק איפה ה- $loss$  הכי נמוך ונבחר באופציה זו, במידה ולא גזמנו נמשיך רקורסיבית על הבנים.

לאחר שעברנו על כל העץ קיבלנו עץ שנבנה מ- $training\ data$  ובאמצעות ה- $validation\ data$  מזערנו את ה- $loss$  אל מול האלגוריתם הרגיל.

### סעיף 4.3:

ביצעתי ניסויים על מנת לקבוע את גודל קבוצת הוולידציה, באמצעות  $kfold$  עם 5  $splits$  רצתי על כמה ערכים שונים וקיבלתי את התוצאה הבאה:



מכיוון שיש לאלגוריתם הזה, סוג של "אקראיות" עקב חלוקת קבוצת הוולידציה, לאחר כמה ניסיונות קיבלתי את התוצאה הבאה: (הטובה ביותר אצלי).

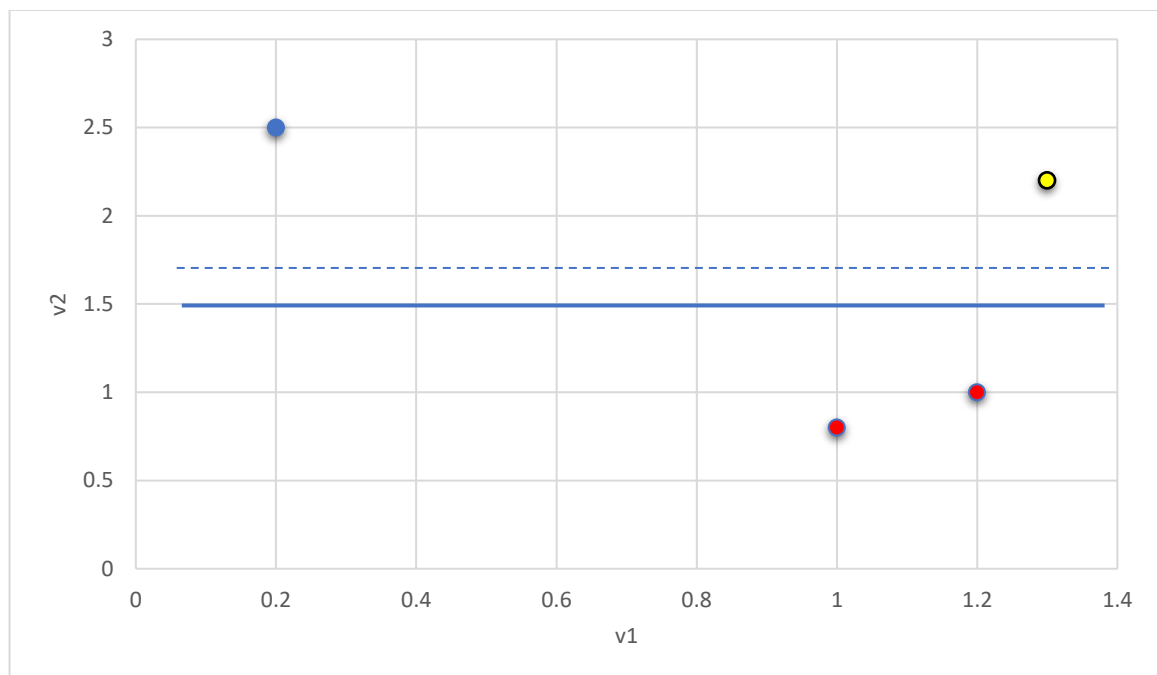
```
(HW1) C:\Users\talro\OneDrive\מיכחטמ\IntroToAi\Projects\tal's project>python CostSensitiveID3.py  
0.001769911504424779
```

## שאלה 5:

סעיף א':

מקרא:

מסווג ID3 מסומן בגרף בקו מקווקוו ומסווג המטרה מסומן בגרף בקו רציף בצבע כחול.



## הסבר

ניתן לראות כי בגרף קיימת דוגמא המסווגת כחיובית ומסומנת בכחול, וקיימות שתי נקודות המסווגות כשליליות והן מסומנות באדום. דוגמת המבחן נמצאת בנקודת (1.3, 2.2) והיא מסומנת בצהוב.

עבור דוגמת המבחן זו ניתן לראות בבירור כי מתקיים לכל ערך  $K$  שייבחר כי למידת עץ ID3 תניב מסווג אשר עונה נכון עבור כל דוגמת מבחן אפשרית, אך למידת KNN מניבה מסווג שעבורו דוגמת מבחן זו הוא יטעה לכל ערך  $K$  שייבחר.

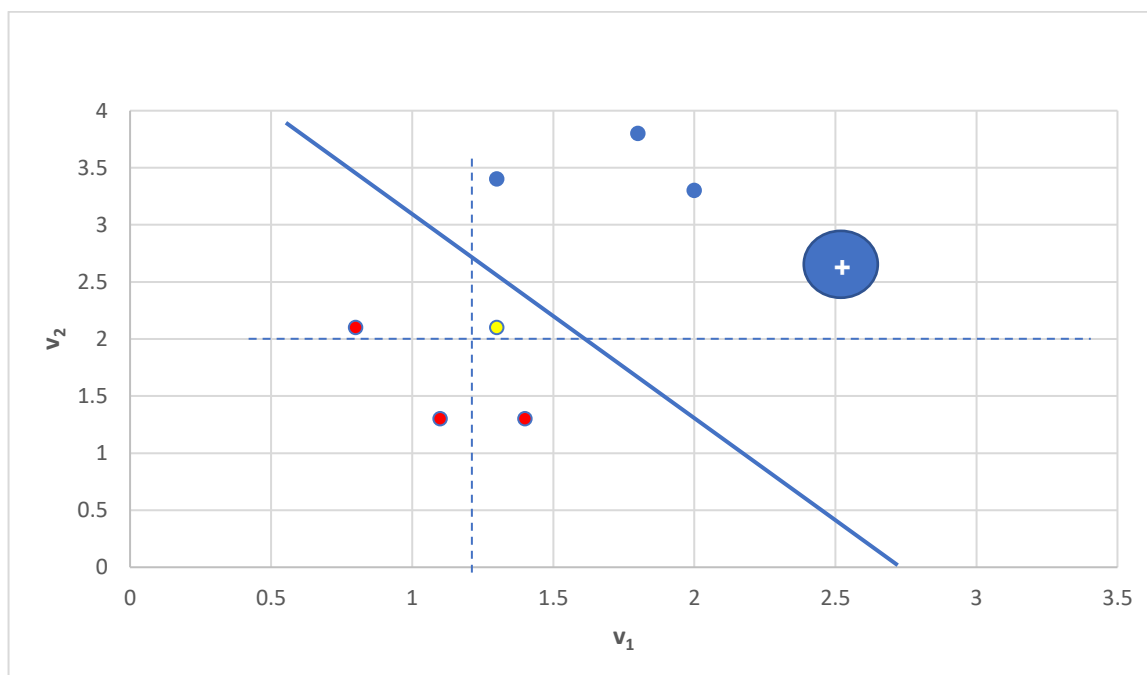
בדוגמת מבחן זו אנו רואים כי לכל ערך  $K$  שייבחר, מתקיים כי מסווג KNN יחזיר תשובה שלילית עבור דוגמת המבחן כאשר הסיווג הנכון עבור דוגמה זו. לעומתו, מסווג ID3 יסווג נכון לכל ערך של  $K$  שייבחר לכל דוגמת מבחן אפשרית (התקבל מסווג המטרה).

למשל, עבור  $k=1$  מסווג KNN יחזיר תשובה שלילית ומסווג ID3 יחזיר תוצאה חיובית, בהתאם למסווג המטרה, כלומר הוא יסווג את הערך הנכון.


## סעיף ב'

מקרא:

מסווג ID3 מסומן בגרף בקו מקווקוו ומסווג המטרה מסומן בגרף בקו רציף בצבע כחול.



## הסבר

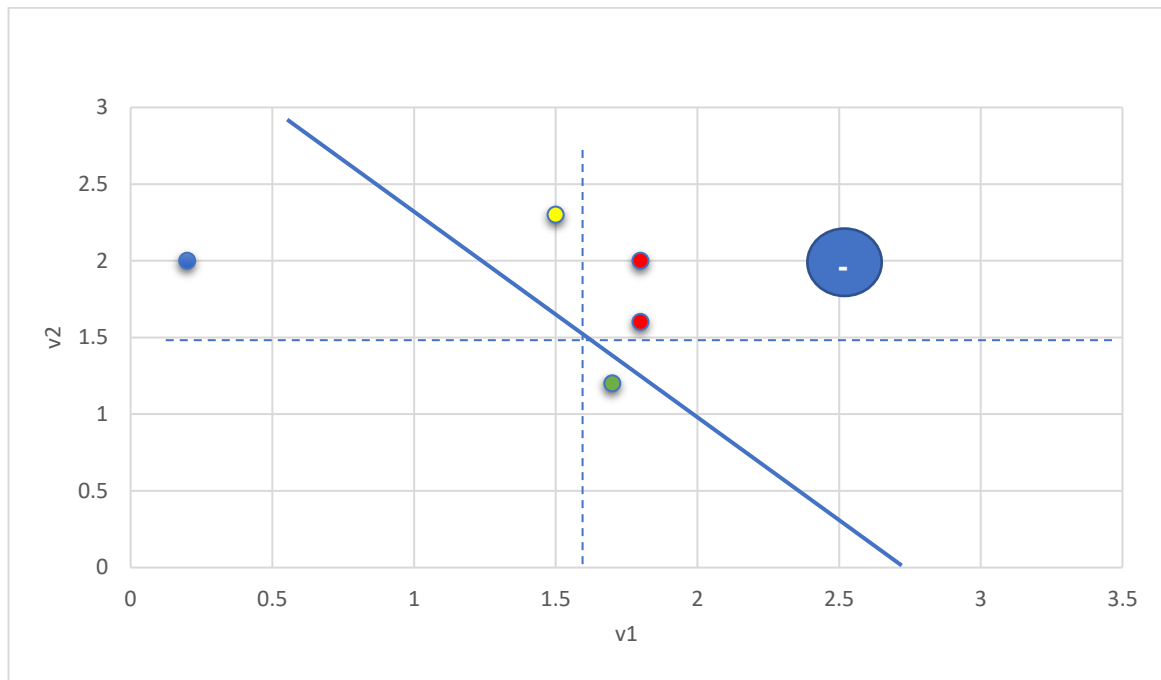
ניתן לראות כי בגרף קיימות שלוש דוגמאות המסווגות כחיוביות ומסומנות בכחול, וקיימות שלוש דוגמאות המסווגות כשליליות והן מסומנות באדום. דוגמת המבחן נמצאת בנקודת (1.3, 2.1) והיא מסומנת בצהוב. כמו כן, נשים לב כי עבור הרבעון המסומן ב-  מבטא שערכו של מסווג ID3 ברבעון זה הוא חיובי ובכל שאר שלושת הרבעונים הוא שלילי.

ניתן לראות בבירור כי מתקיים עבור  $K = 3$  כי למידת מסווג  $KNN$  תניב מסווג אשר עונה נכון עבור כל דוגמת מבחן אפשרית, אך למידת עץ ID3 מניבה מסווג שעבור דוגמת המבחן הצהובה הוא יטעה. בדוגמת מבחן זו אנו רואים כי עבור ערך  $K = 3$  מתקיים כי מסווג ID3 יחזיר ערך חיובי בהתאם לרבעון אליו הוא שייך למרות שדוגמת המבחן שלילית על פי מסווג המטרה. לעומתו, מסווג  $KNN$  יחזיר ערך שלילי ואכן זהו הערך הנכון.

## סעיף ג'

מקרא:

מסווג ID3 בגרף בקו מקווקוו ומסווג המטרה מסומן בגרף בקו רציף בצבע כחול.



## הסבר

ניתן לראות כי בגרף קיימות שלוש דוגמאות המסווגות כחיוביות ומסומנות בכחול, וקיימות שתי דוגמאות המסווגות כשליליות והן מסומנות באדום.

קיימות שתי דוגמאות מבחן:

(1) דוגמת מבחן ראשונה הנמצאת בנקודת (1.5, 2.3) והיא מסומנת בצהוב.

(2) דוגמת מבחן שנייה הנמצאת בנקודת (1.7, 1.2) והיא מסומנת בירוק.

כמו כן, נשים לב כי עבור הרבעון המסומן ב-  מבטא שערכו של מסווג ID3 ברבעון זה הוא שלילי ובכל שאר שלושת הרבעונים הוא חיובי.

עבור הדוגמא הראשונה נקבל:

מסווג מטרה	מסווג KNN	מסווג ID3	ערך K
-	-	+	3

עבור הדוגמא השנייה נקבל:

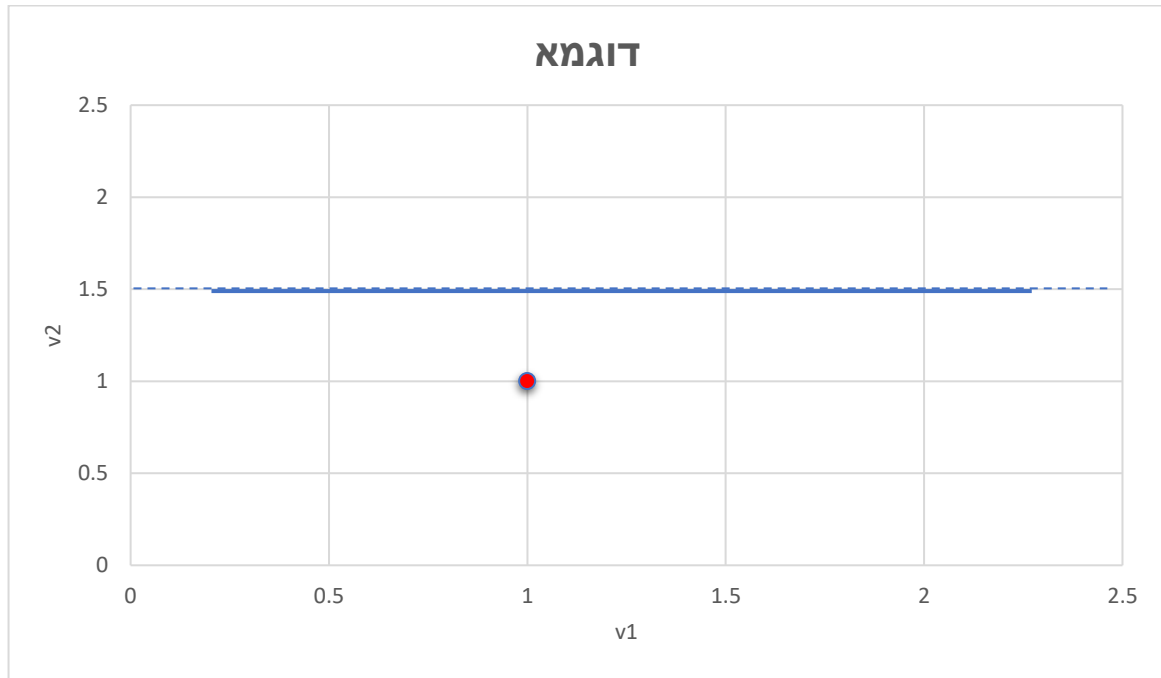
מסווג מטרה	מסווג KNN	מסווג ID3	ערך K
+	-	+	3

הראיתי כי קיימות 2 דוגמאות מבחן, על הראשונה מסווג ה-ID3 טועה ועל השנייה טועה מסווג ה-knn.

## סעיף ד'

מקרא:

מסווג ID3 מסומן בגרף בקו מקווקוו ומסווג המטרה מסומן בגרף בקו רציף בצבע כחול.



## הסבר

ניתן לראות כי בגרף קיימת דוגמא אחת המסווגת כחיובית ומסומנת בכחול, ודוגמא אחת המסווגת כשלילית ומסומנת באדום.

מסווג ID3 יסווג כערך שלילי עבור החלק שקטן שווה ל-  $v_2 \leq 1.5$  ועבור  $v_2 > 1.5$  מסווג ID3 יסווג כערך חיובי. (נבחר להיות מסווג המטרה עצמו)

עבור  $K = 1$  וקבוצת האימון שהוצגה, למידת מסווג KNN מסוים תניב מסווג אשר יחזיר תוצאה נכונה עבור כל דוגמת מבחן אפשרית מכיוון שיש 2 דוגמאות והמרחקים שלהם ממסווג המטרה שווים.

## שאלה 6:

### סעיף 6.1

כתבתי פונקציה לביצוע מציאת  $hyper\ parameters$  הכי טובים  $(N,k,p)$  ניתן להפעיל אותה באמצעות הדגל `find_hyper` - בעת הרצת הקובץ. הפונקציה מבצעת  $kfold$  על קבוצת האימון ובודקת ערכי הפרמטרים לפי רשימות מסופקות או דיפולטיביות. לאחר מכן אני לוקח ממוצע של הדיוק על כל ה- $splits$ .  
לעיתים רחוקות מתקבל דיוק של 0.98 על קבוצת המבחן אך לעיתים יותר קרובות מתקבל הדיוק:

The accuracy for KNNForest=0.9734513274336283

## שאלה 7:

### סעיף 7.1

אסביר בקצרה את שני השיפורים שביצעתי בחלק זה:

#### שיפור בסיסי:

באלגוריתם KNN הרגיל, מודדים מרחק אוקלידי בין  $centroids$  על דוגמאות לא מנורמלות. זה גורם לכך שדוגמאות בעלות יחידות שונות יכולות להשפיע ביתר כוח על המרחק שלא בצדק. יכולים להיות שני וקטורים בעלי מרחק קטן מאוד אך בעצם הם מאוד שונים.  
לכן באלגוריתם שלי, אני מנרמל את כלל התכונות בקבוצת האימון לפי נרמול סטנדרטי שנלמד בהרצאה ואת קבוצת המבחן לפי הערכים שנרמלתי בקבוצת האימון. הנרמול מקנה לי מרחק אמין יותר בין דוגמאות.

#### שיפור ראשון:

באלגוריתם KNN הרגיל, המסווג מצא את ה- $k$  עצים שה- $centroid$  שלהם הכי קרוב ל- $centroid$  של קבוצת המבחן ועליהם בחן את קבוצת המבחן.

באלגוריתם המשופר שלי, לכל דוגמת המבחן אני מוצא את  $k$  העצים שה- $centroid$  שלהם הכי קרוב אליה ובוחן עליהם את דוגמת המבחן הספציפית הזו.

#### שיפור שני:

באלגוריתם KNN הרגיל, המסווג קבע את החיזוי לפי תוצאת הרוב מבין  $k$  העצים שנבחרו לסיווג. באלגוריתם KNN המשופר שלי, אני משתמש בפונקציה יורדת  $e^{-distance}$  כדי לחשב "ציון" של החיזוי לכל עץ. זאת אומרת שאני לוקח בחשבון לא רק את הפרדיקציה שעץ יחיד נתן אלא גם את המרחק שלו מדוגמת המבחן. ככל שהעץ רחוק יותר ה"ציון" נמוך יותר, זאת אומרת עצים קרובים יותר "נחשבים" יותר. אני סוכם את כלל הציונים של כל חיזוי ומחזיר את החיזוי בעל הציון הגבוה.

#### שיפור שלישי:

באלגוריתם KNN הרגיל, מדדנו מרחק אוקלידי רגיל, באלגוריתם המשופר שלי אני מודד מרחק אוקלידי ממושקל על פי משקולות שחישבתי עבור התכונות.

חישוב המשקולות מתבצע באמצעות פונקציה שכתבתי בקובץ `ImprovedKNNForest.py` וניתן להריץ אותה עם הדגל `-feature_weights`.

הפונקציה מבצעת  $kfold$  לכל תכונה, ובכל  $split$  מתבצע אימון על קבוצת אימון ומבחן על קבוצת המבחן ולאחר מכן מתבצע אימון על קבוצת אימון כך שעבור התכונה שנבחרה ביצעתי פרמוטיציה על כל הערכים שלה.

ממוצע ההפרשים בדיוק הוא המשקולת של התכונה. לאחר מציאת כל המשקולות נרמלתי אותן לערכים בין 0 ל-1.

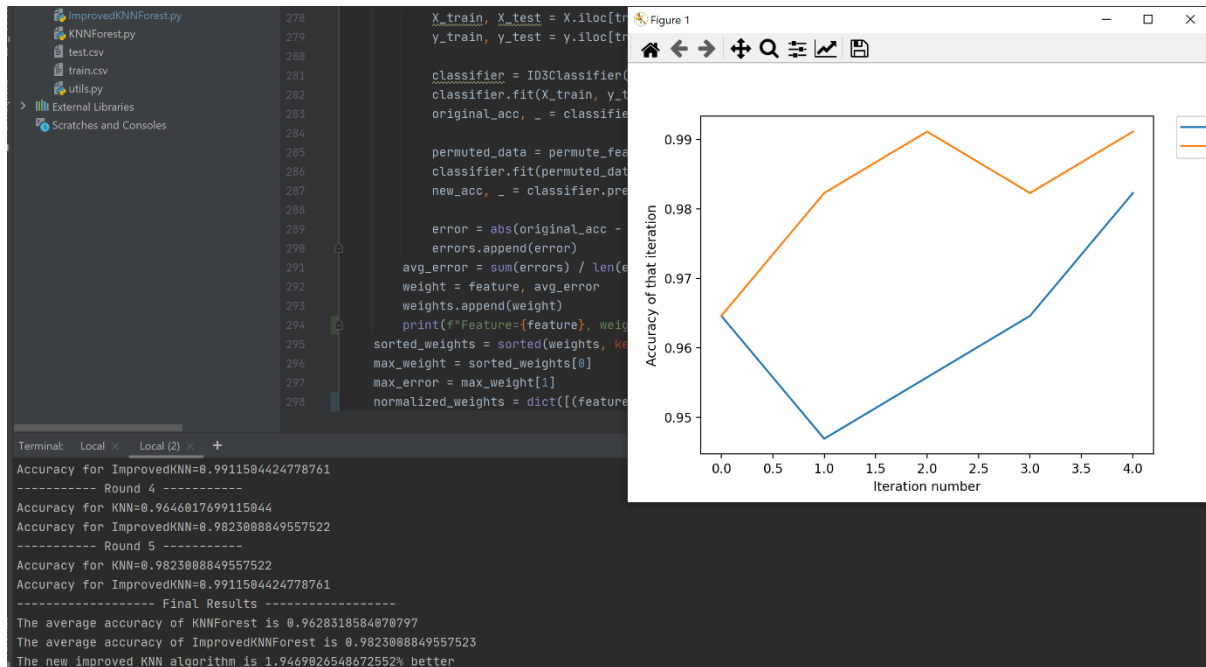
ההיגיון בדרך החישוב הוא שאם לאחר "ערבוב" הערכים של תכונה הדיוק לא השתנה – התכונה חשובה פחות, ואם לאחר "ערבוב" הערכים השתנה הדיוק, לתכונה זו יש חשיבות לחישוב.

חישבתי משקולות אלו באופן חד פעמי והן נמצאות באופן קבוע כחלק מהמסווג.  
(החישוב אורך בסביבות ה-25 דקות)



## סעיף 7.2:

הפרמטרים שהשתמשתי בהם הם אותם hyper parameters שמצאתי בסעיף 6, באמצעות אותו ניסוי.  
הפרמטרים הנוספים הם המשקולות שמצאתי באמצעות ניסוי שהסברתי בסעיף הקודם.



הדיוק המקסימלי מתקבל לעיתים רחוקות כ- 1.0 אך לרוב האלגוריתם נמצא בסביבות בין 0.98 ל- 0.99

```
C:\Users\talro\miniconda3\envs\HW1\python.exe "C:/Users/talro/OneDrive/מסמכים/IntroToAi/Projects/tal's project/ImprovedKNNForest.py" -v
The accuracy for ImprovedKNNForest=0.9911504424778761
```