<u>פרוייקט מסכם בבינה - תום כהן 204576946</u>

<u>שאלה 1:</u>

```
C:\Users\HP\AppData\Local\Programs\Python\Python37\python.exe C:/Users/HP/Pycha
ID3 accuracy is: 0.9469026548672567

Process finished with exit code 0
```

<u>שאלה 2:</u>

נוכיח את הטענה,נסמן בT את העץ של המסווג הרגיל וב'T את העץ של המסווג עם פונקציית הנרמול.
נראה כי שני העצים זהים עד כדי ערך הthreshold השייך לתכונה.
ואז נראה שכל דוגמאת מבחן עושה את אותו המסלול בעץ T וב'T ולכן בגלל שהעצים זהים עד כדי ערך threshold השייך לתכונה, נקבל שלכל דוגמת מבחן הסיווג לפי העץ T שווה לסיווג לפי העץ 'T וזה יוכיח כי פונקציית הנרמול מינימקס לא משנה את דיוק המסווג הנלמד.

<u>טענת עזר:</u> תהי קבוצת דוגמאות מבחן, נסמן את האלגוריתם ללא הנרמול בA ועם הנרמול ב'A , ואת f,b כתכונה והסף שהאלגוריתם A בחר בשביל להפריד את התכונות ואת f',b' כתכונה והסף שהאלגוריתם 'A בחר כדי להפריד אותן
אזי f=f' ולכל דוגמה e מתקיים: f(e)>b אממ f'(e)>b' .

נשים לב כי הטרנסופומציה מX לערכו לאחר נרמול מינימקס היא: הורד X_{min} (חסר מX מספר שלילי) ולאחר מכן הכפל את מה שייצא במספר החיובי $\frac{1}{X_{min}-X_{\max}}$. שתי הפעולות משמרות אי שיוויון ולכן מתקיים:

$$x1 \leq x2 \text{ } if \text{ } and \text{ } only \text{ } if \text{ } normalize(x1) \leq normailze(x2)$$

לכן לפי מבנה Id3 שמימשנו לכל תכונה הדוגמאות ימויינו לפי אותו סדר. סדר הדוגמאות הוא הדבר היחיד שמשפיע על חישוב הIG כשרוצים לבחור את הסף לפיצול, לכן לכל תכונה נבחר סף מיטבי שיפריד בין הדוגמאות בשני האלגוריתמים בדיוק באותה צורה, ולכן לכל קבוצה נבחר את אותה התכונה f לפיצול הקבוצה בשני העצים.

בנוסף בגלל ששניי הספים מפצלים את הדוגמאות בדיוק באותו אופן נקבל שלכל דוגמה e מתקיים: f(e)>b אממ f'(e)>b' .
בנוסף ממבנה אלגוריתם ID3 ובגלל שהסדר בערכי הX נשמר ומהגדרת פונקציית נרמול minimax נקבל ש b'= normalize(b)

<u>טענת עזר 2:</u> לכל f,b שנבחרו על ידי A ולכל e דוגמת מבחן מתקיים f(e)>b אממ f(e')>b' עבור f',b' המתאימים שנבחרו על ידי 'A כאשר e היא דוגמת המבחן המנורמלת :
לפי טענה קודמת פונקציית הנרמול משמרת אי שיוויון לכן: $f(e) > b \text{ } iff \text{ } normalize(f(e)) > normalize(b)$
ולפי מה שהראנו קודם b'= normalize(b) וf(e') היא סימון לnormalize((f(e)) נקבל ש: $f(e) > b \text{ } iff \text{ } f(e') > b'$

נראה באינדוקציה על עומק T כי לכל רמה בעץ העצים T ו'T זהים עד כדי ערכי threshold .

<u>בסיס:</u>

<u>i=1:</u>

לפי טענת העזר נבחרה אותה תכונה בשני העצים להפרדת הדוגמאות ועבור אותה תכונה הדוגמאות פוצלו באותו אופן בין הבנים.

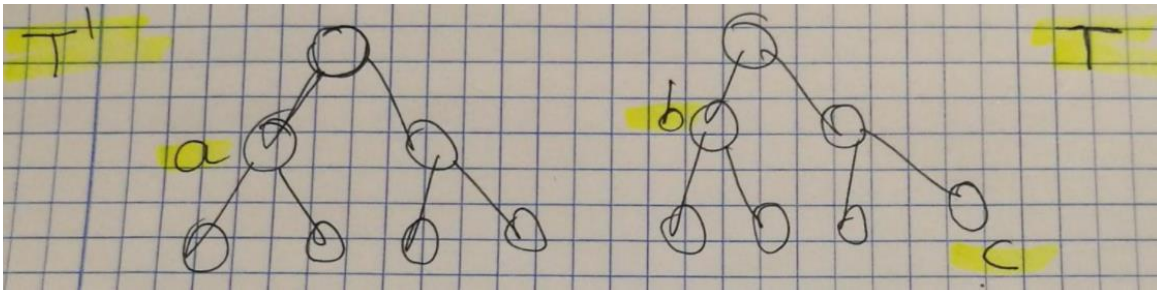
<u>צעד:</u>

נניח את נכונות הטענה עד הרמה ה-i-1 ונראה עבור הרמה הi :

צומת ברמה הi הוא בן של צומת ברמה ה-i-1 ולכן לפי הנחת האינדוקציה אותן דוגמאות מבחן נמצאות בכל צומת ברמה הi ושוב לפי טענת העזר נבחרה אותה תכונה לכל צומת ברמה הi ולכל צומת ברמה זו הדוגמאות פוצלו באותו אופן בין הבנים שלה.

לכן נקבל ששניי העצים זהים עד כדי ערכי threshold, (בפרט נובע שבשני העצים קיימים אותם עלים עם אותם הסיווגים).

תהי דוגמאת מבחן, נראה באינדוקציה על אורך מסלול החיפוש בשני העצים שהצמתים בשני העצים T ו'T בעלי אותו מיקום ושנעשתה בהם את אותה הפנייה בעץ. לצורך הבהרה הצמתים a וb באותו מיקום בשני העצים והצמתים a וc לא.



<u>בסיס:</u>

<u>i=1:</u>

בשני העצים מתחילים בשורש ועושים את אותה פנייה לפי טענת עזר 2.

<u>צעד:</u>

נניח את נכונות הטענה עד הצומת ה-i-1 במסלול ונראה עבור הצומת הi:

הצומת הi הוא הבן של הצומת ה-i-1 לפי הנחת האינדוקציה בשני העצים הצמתים i-1 זהים במיקום ונעשתה בהם אותה פניה ולכן גם הצומת הi בשני המסלולים בעל אותו מיקום. מטענת עזר 2 תתבצע בשני העצים אותה פניה.

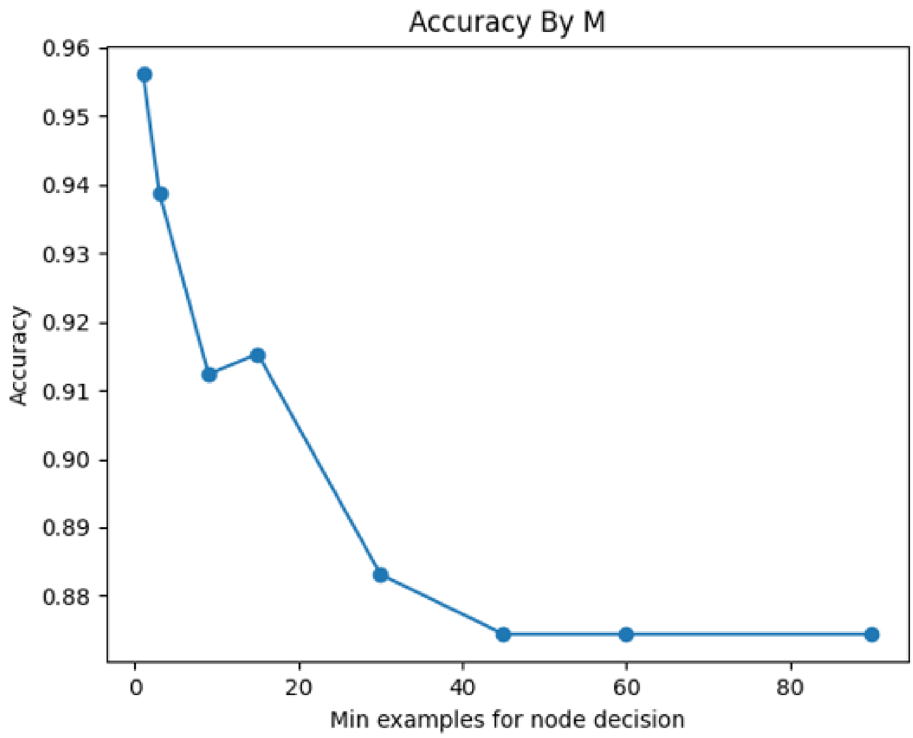
לכן בסופו של דבר בשני העצים הדוגמא תגיע לאותו עלה עם אותו סיווג (כי העצים זהים עד כדי threshold), ולכן תסווג באופן זהה על ידי שני העצים.

שאלה 3:

סעיף 3.1:

באופן כללי גיזום מביא לעצי החלטות קטנים יותר אותם אנו מעדיפים לפי עקרון התער של אוקמה וכי בעץ קטן יותר יש תמיכה סטטיסטית גדולה יותר לכל עלה. הגיזום מנסה למנוע את תופעת הover fiting בה דיוק גדול יותר על קבוצת האימון גורם לדיוק קטן יותר על קבוצת המבחן (נגרם בדר"כ ע"י דוגמאות רועשות). לכן אנחנו מעוניינים בעץ לא עקבי עם קבוצת האימון. לכן אנו גוזמים את העץ לפני שמפרידים את הדוגמאות לחלוטין אחת מהשניה לפי סיווגן, בתקווה להקטין את שגיאת המבחן.

סעיף 3.3:



עבור גיזום עם M=1 , כלומר ללא גיזום בכלל קיבלתי את הדיוק המקסימלי.
כנראה שהדוגמאות לא רועשות, ושאין הרבה או בכלל תכונות מיותרות ולכן לא סובלים מoverfiting שגיזום מוקדם מקטין - אם סיווג של כל עלה נקבע תמיד לפי לפחות 10 דוגמאות (הסיווג השכיח ב10 האלו) לעומת ללא גיזום מוקדם בו סיווגים של עלים יכולים להיקבע גם לפי דוגמא בודדת, אז הסיכוי שנטעה בסיווג של דוגמאת מבחן בגלל רעש קטן יותר, כי יותר סיכוי שדוגמא אחת רועשת מאשר 6 (במקרה שקובעים סיווג של צומת עם 10 דוגמאות לפי הסיווג השכיח בה, צריך במינימום לפחות 6 דוגמאות מאותו סיווג) .
כאן דווקא בניגוד לעקרון התער של אוקומה עץ החלטות עמוק יותר הוא יעיל יותר וכנראה שיש צורך במס' רב של תכונות כדי להפריד את הדוגמאות וכדי להחליט שצומת בעץ היא עלה בעל סיווג כלשהוא.

סעיף 3.4 :

```
ID3 accuracy is: 0.9469026548672567
Id3 accuracy with best M is: 0.9469026548672567
```

השורה הראשונה היא הדיוק עם ID3 ללא גיזום השורה השנייה היא הדיוק עם ID3 עם גיזום עם הM המיטבי שהוא M=1 ולכן בפועל לא גוזמים ויוצאת אותה תוצאה ולא מקבלים שיפור ביחס לריצה ללא גיזום בשאלה 1.

שאלה 4- למידה מכוונת מחיר:

סעיף 4.1:

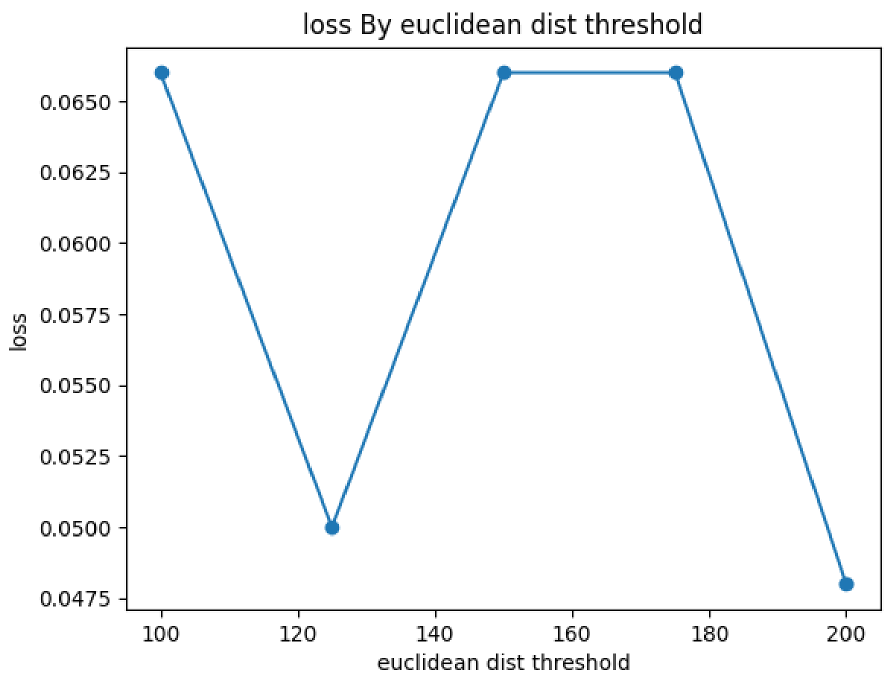
```
ID3 loss before improvment is 0.021238938053097345

Process finished with exit code 0
```

סעיף 4.2:

נעשה שלושה שינויים בסעיף זה כדי ללמוד מסווג שימזער את פונקציית הLoss:

1) לפי הfaq בסעיף זה מותר לבצע בסעיף זה עיבוד מוקדם של הדאטא, אפשר לראות מהגדרת הLoss שככל שעושים יותר שגיאות כך הLoss גדל, נבצע עיבוד מוקדם של הדאטא כדי להקטין את הoverfitting לקבוצת האימון ובך להקטין את מס' השגיאות ואת הLoss. העיבוד המוקדם ייתבצע באופן הבא: עבור חסם מינימלי על מרחק אוקלידי בין דוגמאות (מרחק אוקלידי כמו בסעיף של הKnnForest), דוגמאות שיהיו במרחק קטן מהחסם ובעלי סיווג שונה, יימחקו מקבוצת האימון(שתיהן) והאלגוריתם לא ייתאמן עליהן. ההיגיון מאחורי זה אומר שאם שתי דוגמאות ממש דומות ואחד חולה ואחד לא אז יש סיכוי גדול שמדובר בדוגמאות רועשות ולכן נמחק אותן מקבוצת האימון כדי למנוע Over Fiting.

את הערך המיטבי לחסם האוקדלידי מצאתי על ידי ניסויים המפורטים בגרף הבא:

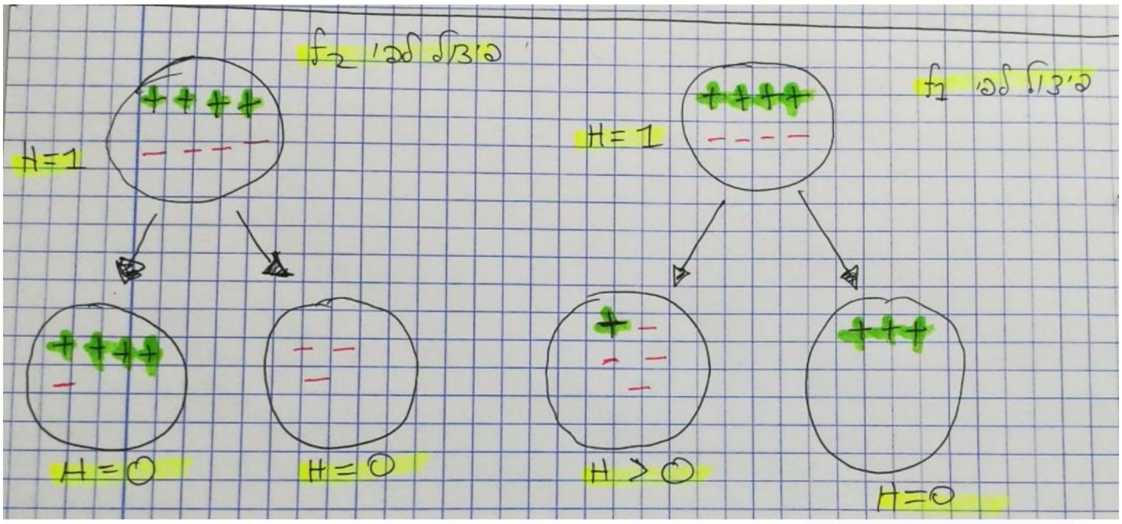


לכל ערך חסם אוקלידי שבדקתי בניסוי השתמשי ב50 האינדקסים הראשונים של הtrain לחישוב הLoss ובשאר האינדקסים כקבוצת האימון שממנה אני מסיר דוגמאות דומות עם סיווגים שונים.
לבסוף השתמשי בחסם האוקלידי 125 שעבורו קבילתי loss מינמלי לסינון **כל** קבוצת האימון לפני הלמידה עליה.

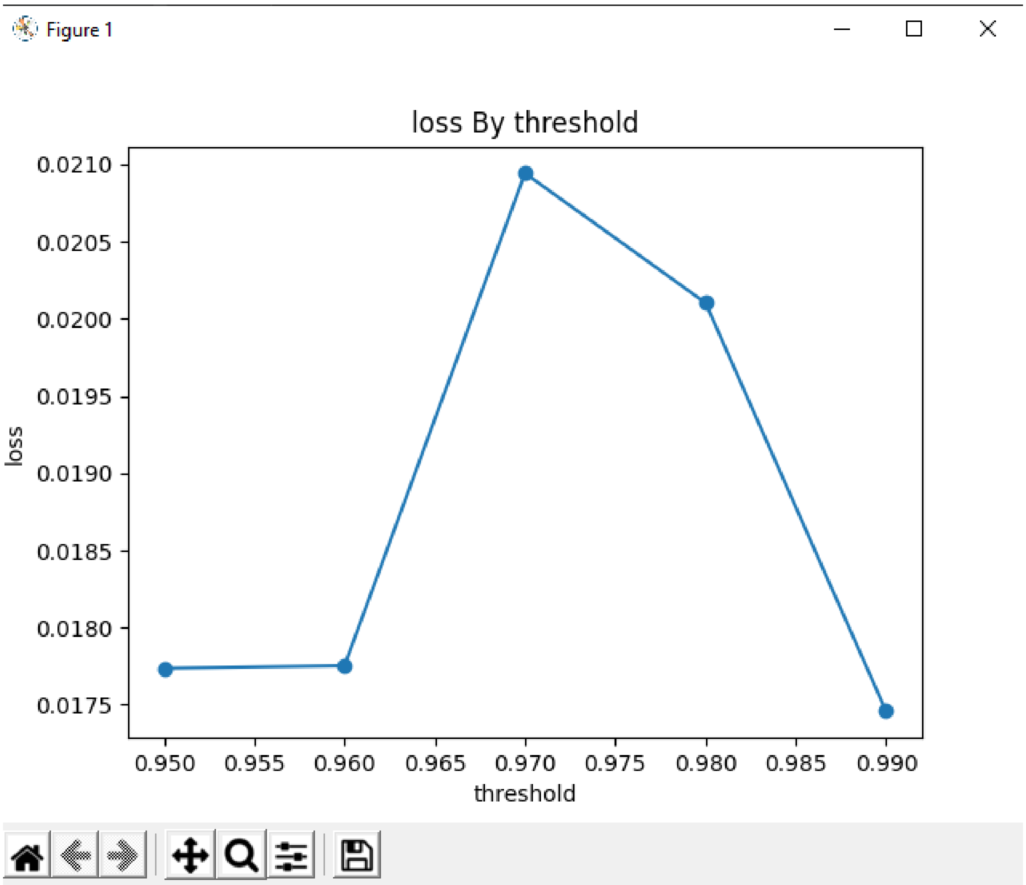
2) מהגדרת הLoss כל שגיאת FN מגדילה אותו הרבה יותר, לכן כדי למזער את הLoss נמזער את השגיאות האלו באופן הבא:

2.1 – כשאנחנו עושים שגיאת FN אנחנו אומרים על מישהו חולה שהוא בריא, לפי ההיגיון אם יש צומת עם הרבה דוגמאות של חולים ודוגמא אחת בריאה, אז בגלל הדימיון בין הדוגמאות (הגיעו לאותה צומת) כנראה שהאחת שבריאה היא דוגמה רועשת ולכן יש לסווג את כל הצומת כעלה עם סיווג חולה ולא להפריד שוב בעזרת עוד תכונה.
נגדיר חסם threshold, נשנה את הפונקציה isConsistentNode הקובעת אם node יהפך לעלה בעל הסיווג של רוב הדוגמאות בצומת, כך שאם החלק היחסי של הדוגמאות החיוביות (חולים) גדול מthreshold אז נקבע את הצומת כעלה, לא נבצע את אותו טכניקה עבור הבריאים כדי להמנע מדוגמאות רועשות הגורמות לשגיאת FN אותה אנחנו מנסים למזער . נבחר חסם מעל ל0.95 ולכן העלה יסווג כחולה במידה ועבר את החסם (מעל 0.95 כדי לא לסווג בריאים באמת כחולים ולהגדיל את הFP שגם מגדיל את הLoss), ועד לערך 0.99 כדי שבאמת תהיה השפעה לשינוי שעשינו.

2.2- לפי אותו היגיון הנ"ל , נשתמש באותו threshold באותה צורה כך שאם יש node שאחוז החולים בו גדול מthreshold אז נחזיר 0 עבור הnode (כאילו כולם היו חולים). למשל בדוגמא הבאה נעדיף לפצל לפי f2 והבן השמאלי שלה יהיה עלה, ובמקרה הגרוע נקבל שגיאת FP (בגלל השינוי הקודם שעשינו) שערכה בLoss קטן ובמקרה הטוב הדוגמא השלילית רועשת וכך נמנע את המצב שהיינו מפרידים שוב את הדוגמא השלילית משאר הדוגמאות החיוביות , מסווגים אותה כשלילית ומגדילים את הסיכוי לקבל שגיאת FN יקרה(שוב לפי ההיגיון שאם הרבה דוגמאות חולות ואחת בריאה זה כנראה רעש).



לצורך קביעת ערך הthreshold המיטבי עשיתי kfold על קבוצת האימון עם k=5 וקיבלתי ש0.99 הוא הthreshold המיטבי:



<u>סעיף 4.3:</u>

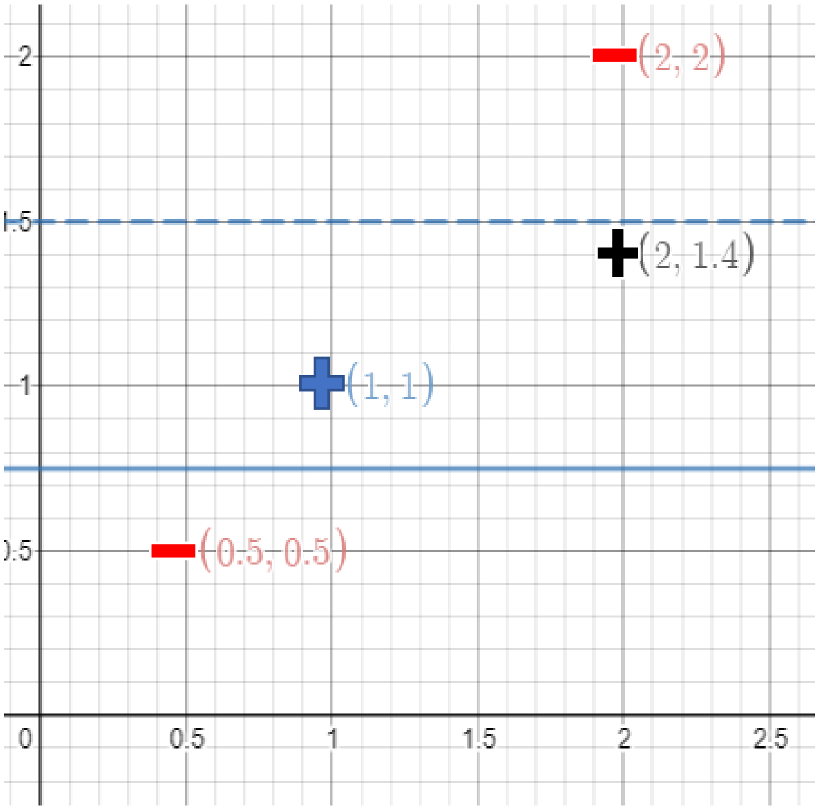
אפשר לראות ששיפרנו את הLoss :

```
C:\Users\HP\AppData\Local\Programs\Python\Python37
Loss after improvement is: 0.007079646017699116

Process finished with exit code 0
```

שאלה 5:

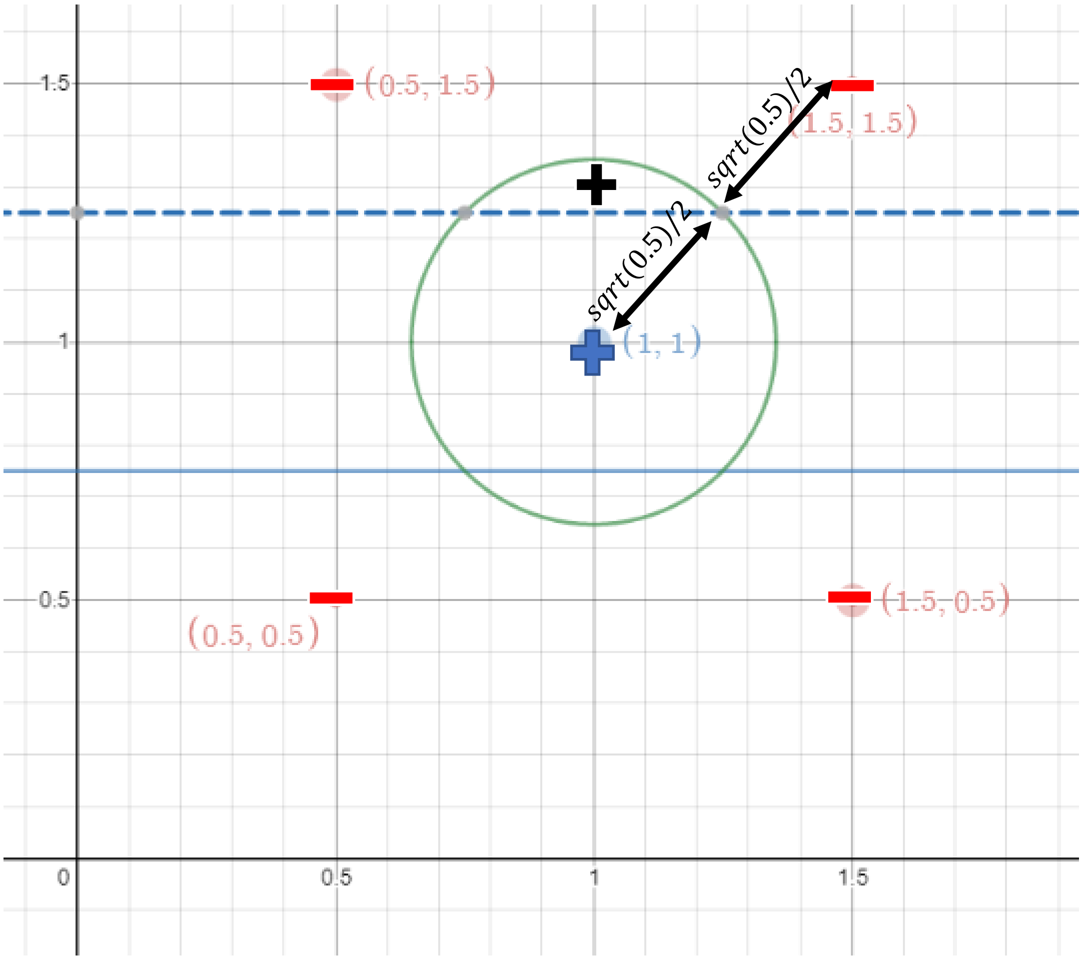
בסעיפים הבאים אני מניח שייתכנו רק K אי זוגיים, כי ווידאתי זאת מול כותב התרגיל.
בשרטוטים הבאים המסווג של ID3 הוא האזור שמופיע בין הקווים הכחולים, עבור נק' שעל קו הם מסווגות חיוביות אמ"ם הקו מלא.
ציר ה-X מתאר את הערך V1 של תכונה וציר הY את ערך הV2 .

א)



הנק' השחורה היא הנק' עבורה KNN יטעה וID3 ייצדק.
המסווג (שהוא גם מסווג המטרה) מסווג דוגמה כחיובית אמ"ם היא מקיימת $0.75 \leq X.v2 < 1.5$
ערכי k אפשריים הם 1 ו3 כי הקבוצה מגודל 3 ואפשר לראות שעבור שניהם הדוגמא תסווג שלילית אפילו שהיא חיובית.

ב)



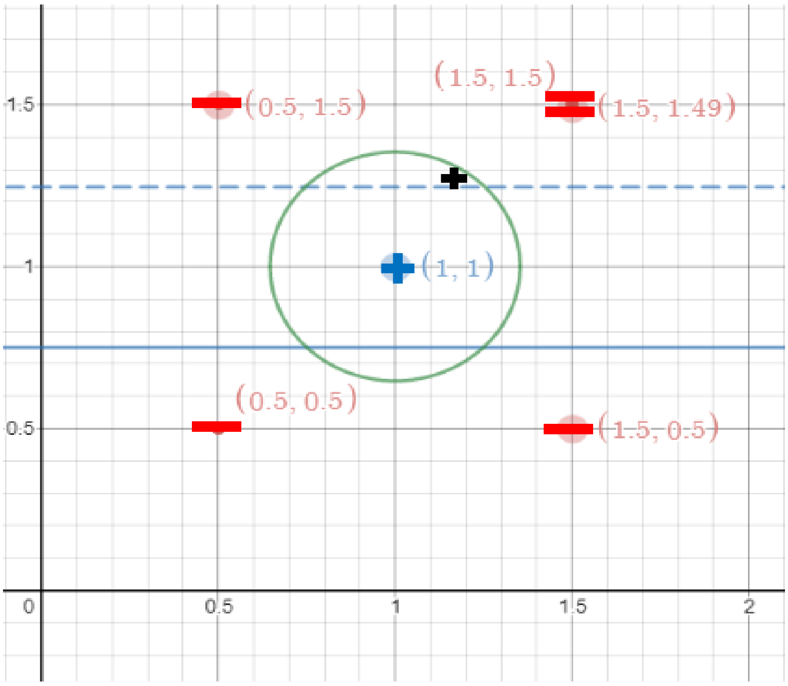
הנק' השחורה היא הנק' עבורה KNN ייצדק וID3 יטעה.

ID3 מסווג דוגמה כחיובית אמ"ם היא מקיימת $0.75 \leq X.v2 < 1.25$

מסווג המטרה הוא עיגול בקוטר $sqrt(0.5)/2$ שמרכזו בנק' (1,1) .

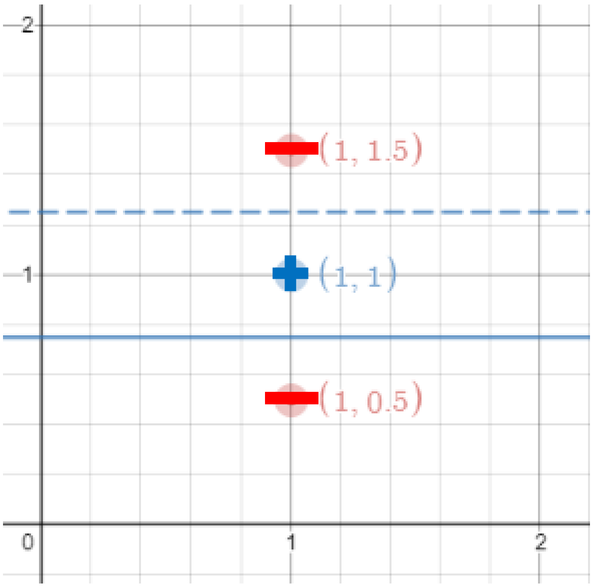
עבור ערך k=1 נקבל שknn הוא מסווג המטרה ואילו ID3 יסווג את הנק' החיובית השחורה כשלילית.

ג)



מסווג המטרה הוא עיגול בקוטר $sqrt(0.5)/2$ שמרכזו בנק' (1,1) בדיוק כמו בציור הקודם.
ID3 מסווג דוגמה כחיובית אמ"ם היא מקיימת $0.75 \leq X.v2 < 1.245$
לכן ייטעה עבור הנק' המסומנת ב+ שחור ויסווגה כשלילית במקום כחיובית, KNN עבור K=3 ייטעה גם בסיווג הנק' האחרונה מכיוון שיש 2 דוגמאות עם סיווג שלילי שקרובות אליה (1.5,1.49),(1.5,1.5) .

ד)



ID3 מסווג דוגמה כחיובית אמ"ם היא מקיימת $0.75 \leq X.v2 < 1.25$ וזהו גם מסווג המטרה.
Knn עבור *k=1* יסווג כל דוגמת מבחן עם הסיווג הנכון כי כל מה שנופל בתוך מסווג המטרה יהיה יותר קרוב לדוגמא *(1,1)* שהיא בעלת סיווג חיובי וכל מה שבחוץ יהיה יותר קרוב לדוגמאות עם הסיווג השלילי (זה מתקיים גם עבור מקרי הקצה של נק' על שני הקווים, עקב הצורה בה הגדרתם את שבירת השיוויון עבור נק' במרחק זהה מדוגמת מבחן כלשהיא ב*KNN*)

שאלה 6:

הניסוי שעשיתי כדי למצוא את קבוצת הפרמטרים הוא:
לכל שלשה *n,k,p* כאשר *n* רץ מ2 עד 9 , *k* רץ מ2 עד 9 ו*p* רץ מ 0.3 עד 0.7 בקפיצות של 0.5 , עשיתי *k-fold*
על קבוצת האימון וחישבתי את הדיוק לשלשה זו, בחרתי בסוף בשלשה שמיקסמה את הדיוק:
(n,k,p) =(9,9,0.7)
את הניסוי אפשר למצוא בפונצקיה *experiments()*.

הדיוק שקיבלתי על קבוצת המבחן הוא:

```
C:\Users\HP\AppData\Local\Programs\Py
0.9557522123893806

Process finished with exit code 0
```

אפשר לראות שיש שיפור ביחס לדיוק של *ID3*.

שאלה 7:

בסעיף זה המטרה היא לשפר את הדיוק של *knn-decision-tree* , לכן ניצור שלושה *knn-decision-tree* שיהוו יער של יערות (וועדה של ועדות).
כשנרצה לסווג דוגמת מבחן נחזיר את הסיווג השכיח ביותר מבין הסיווגים של 3 היערות.
כל יער ייתן משקל שונה ל*K* העצים הקרובים ביותר שייבחר-
כך נייצר כאן שיפור שאינו רק שינוי של הפרמטרים ב*Knn decision tree* מסעיף קודם-
(כי אם כל יער מסווג דוגמא באותה דרך וצירפנו 3 יערות אז בגדול פשוט ייצרנו יער גדול יותר, עד כדי הבדלים של קבוצת המבחן שנבחרה לכל יער וזה שמפעילים *Knn* בתוך כל יער ואז עושים בוחרים את הסיווג השכיח ביותר מבין הסיווג של כל היערות)

היער הראשון לא ימשקל את הסיווגים של *K* העצים שנבחרו – (יער זהה ליער בסעיף הקודם)
היער השני ייתן משקל גדול יותר לסיווג של עץ ככבל שהוא עמוק פחות-
לפי ההגייון שככל שהעץ עמוק פחות לכל עלה יש תמיכה סטטיסטית גדולה יותר (לפי עקרון התער של אוקומה), אז כבר בשלב האימון חישבתי את העומק של כל עץ ונתתי לכל עץ משקל גדול יותר ככל שהוא עמוק פחות- בפונקציה *update_tree_weights* של המחלקה *DepthWeightKnnForest*.
לאחר מכן כדי לקבוע את סיווג היער עבור דוגמה:
איתחלתי שני קאונרטים ל0 אחד לקביעה שהדוגמא חולה ואחת בריאה ואז אם עץ של היער סיווג דוגמא כ*X* הוספתי את המשקל של העץ לקאונטר המתאים. לבסוף החזרתי את הסיווג של הועדה כסיווג המתאים לקאונטר הגדול יותר. למעשה כל הלוגיקה הזאת מופיעה בפונקציה *calculateMajorityClass* ו*ForestClassifier* של המחלקה הנ"ל.
היער השלישי ייתן משקל גדול יותר לסיווג של עץ שקרוב יותר לדוגמא, לפי ההיגיון שאם עץ קרוב יותר לדוגמא ב*centroid* שלו אז הוא נבנה מאוכלסייה של דוגמאות שדומה יותר לדוגמא ולכן יידע לסווג אותה בצורה טובה יותר.
אפשר למצוא את הלוגיקה שמבצעת את המשקול הזה בפונצקייה *forestClassifier* של המחלקה *DistanceWeightedKnnForest* .

היערות שונים זה מזה- כל יער קובע את הסיווג של הדוגמא בדרך אחרת, ובסוף אנחנו לוקחים את החלטת הרוב לכן הגיוני שבצורה זו נשפר את הדיוק לפי אותו היגיון שהרבה **עצים שונים** משפרים דיוק לעומת עץ אחד ולכן משתמשים ביער החלטה (בסעיף זה יצרנו יער של יערות), ואכן ראיתי שהדיוק משתפר ביחס לסעיף קודם.

בנוסף בגלל שהמטרה בסעיף זה היא לשפר את הדיוק: בכל היערות בכל עץ שניצור בדומה לסעיף 4 נשנה את הפונקציה *isConsistentNode* כך שתסווג צומת כעלה אם אחוז הדוגמאות החולות **או** (זה השוני מסעיף

4) הבריאות בו גדול מ *threshold* . צעד זה נובע מאותו היגיון שהוסבר בשלב 4 לפיו אם יש הרבה מאוד דוגמאות מאותו סיווג ומעט מאוד עם סיווג שונה, וכל הדוגמאות דומות אחת לשניה בתכונות שלהן כי הן באותו צומת אז הדוגמאות בעלות הסיווג שנמצא במיעוט הן כנראה דוגמאות רועשות.
בדרך זו הדיוק יגדל מ2 סיבות:

א. הקטנו את ה*overfitting* לקבוצת האימון.
ב. בגלל שיצרנו עלה, לצומת שלפני השינוי היה מפוצל הקטנו את עומק העץ וככה קיבלנו עץ פחות עומק לפי העקרון של אוקומה ולכן כל עלה יהיה בעל תמיכה סטטיסטית גדולה יותר.

* השתמשתי באותו *threshold* שקיבלתי מסעיף 4- 0.99

<u>ניסויים:</u>

עשיתי את אותו ניסוי שעשיתי בסעיף הקודם לכל סוג אחר של יער- הניסויים מופיעים בקובץ, גם כאן יצא לי שהשלשה האופטימלית היא *(n,k,p) = (9,9,0.7)* . למרות זאת בגלל שאני רוצה ליצור יערות שונים זה מזה כמו שהסברתי קודם למרות שיצא לכל יער בנפרד פרמרטר *P* אופטימלי 0.7 נתתי לכל יער פרמטר *p=0.3* מינמלי ביער המשופר כדי שהיערות יתאמנו על דאטא שונה ככל האפשר (לפי ההיגיון שועדה של מומחים בה לכל המומחים בה יש את אותו רקע מדעי היא לא וועדה טובה לעומת וועדה בה כל מומחי יכול להביא את היתרון היחסי שלו).
בנוסף עשיתי את הניסויים הבאים על ידי הרצות ידניות (לא מופיע בקובץ):

- ניסיתי למשקל בין ההחלטות של 3 היערות- לא ראיתי שזה משפר, אז השארתי לכל החלטה של יער משקל 1.
- עשיתי ניסוי כדי לקבוע את המשקול של *k* העצים ב *DistanceWeightedKnnForest* בסוף הגעתי לצורת המשקול הבאה: לעץ ה*i* הקרוב ביותר מתוך העצים (הקרוב ביותר הוא העץ ה0 והרחוק ביותר הוא העץ *k-1*) אני נותן משקל של $\frac{k-i}{k}$. כך ככל שעץ קרוב יותר לדוגמאת מבחן הוא מקבל משקל גדול יותר.

להלן הדיוק שאחנו מקבלים עם היער המשופר, אפשר לראות שהשתפרתי ביחס ליער הרגיל:

```
C:\Users\HP\AppData\Local\Programs\P
0.9823008849557522

Process finished with exit code 0
```

תודה, היה אחלה קורס :), באמת.