חלק א – יבש שאלות פתוחות

1- חיפוש מיודע

א- *IDA עדיף על פני אלג' *A לפי מדד ביצועי שהוא צריכת הזיכרון, *IDA בעל צריכת זיכרון לינארית באורך מסלול הפתרון להבדל מ- *A בו צריכת הזיכרון פרופורציונלית למספר הצמתים שנוצר ב- OPEN איחוד CLOSE.

ב- מספר האיטירציות המקסימלי של אלג' ID-DFS הינו מספר הקריאות לאלגוריתם הבסיס DFS_l מהפונקציה החיצונית, מספר זה הינו d+1 (עומק הפתרון +1), כיוון שאנחנו קוראים ל- DFS_l כל עוד לא מצאנו פתרון בשכבה הנוכחית וקוראים לו עם עומק חדש, כלומר קוראים לו פעם ראשונה עם עומק 0, פעם שנייה עם עומק 1, וכולי עד הפעם d+1 בה נשלח עומק d (שזה הוא גם עומק הפתרון) נסיים הריצה שם כיוון שהוא ימצא אותו.
סה"כ קראנו ל- DFS_l, d+1 פעמים כלומר d+1 איטרציות.
מספר זה קטן משמעותית ממספר האיטרציות המקסימלי ב- *IDA, נשים לב ש- *IDA קורא ל- DFS_f כל פעם עם הגבלת f חדשה לכן יכול להיות שהוא כל פעם מפתח רק צומת אחד חדש ואז מבצע איטרציה חדשה עם f_limit חדש לכל צומת. כלומר בשונה מ- ID-DFS שהוא מפתח שכבה שלמה בכל איטרציה, *IDA יכול לבצע k איטרציות כדי לפתח שכבה אחת כאשר k הינו מספר הבנים באותה שכבה. (בכל הגדלה של f_limit הוא יפתח צומת אחד חדש בלבד)

ג- (i) כפי שלמדנו אלג' *IDA מבצע מספר איטרציות מקסימלי כמספר ערכי f_limit האפשריים כלומר מספר ערכי f_limit האפשריים לאלג'.
A1 מעדכן f_limit כך שיגדל לפחות 1/k כל פעם, כלומר בכל איטרציה יהיה לנו f_limit חדש בהפרש 1/k מהקודם לפחות, לכן אם עלות הפתרון האופטימלי הינה c_d^* אז A1 ימצא פתרון כאשר f_limit הינו c_d^* כיוון שבכל איטרציה יגדל ב- 1/k לפחות, לכן לכל היותר A1 יבצע c_d^* / (1/k) כלומר kc_d^* איטרציות על בעיה כלשהי במרחב D.
(ii) נשים לב כי f תגדל כל פעם בלפחות 1/k. אם כן, צמתים באותה השכבה כך שהערך ה- f שלהם גדול בפחות מ- 1/k מערך ה- f של הצומת האחרון שהוביל ל- f_limit חדש, גם הם יפתחו באיטרציה הבאה (זה בשונה מ- *IDA שהיה מפתח כל אחד לחוד), עכשיו הבעיה תהיה אם הצומת הזה (שנפתח באופן "מיותר") יוביל למסלול למטרה בעלות גבוהה יותר ו- *IDA מתחיל בו הרצת DFS, עלות מסלול זה תהיה יותר ממחיר הפתרון האופטימלי בלכל היותר 1/k כי אחרת אנחנו לא נמשיך בלפתח המסלול כי אז נתקל בצומת שיוחרג מ- f_limit בהכרח. לכן האפשרות היחידה להחזרת עלות אחרת מ- c_d^* הינה עבור מסלול למטרה בעלות c_d^* + 1/k לכל היותר (לא כולל), לכן נקבל כי :

$$\varepsilon(A1,D) = path\_cost(A1(D)) - c_d^* < c_d^* + 1/k - c_d^* < 1/k$$

ד- (i) לפי דרך חישוב ה- f, כל הצמתים שערכי f שלהם בהפרש פחות מ- 1/k יקבלו את אותה ערך f^q חדש ויפתחו באותה איטרציה בשונה מ- *IDA שהיה עושה איטרציה חדשה לכל אחד מהם.
כל ערך f בהפרש גדול מ- 1/k (ושווה) יוביל לאיטרציה חדשה, מכאן ניתן להסיק כי f_limit בכל איטרציה חדשה יגדל בלפחות 1/k כמו בסעיף ג. כיוון שערכי g לא השתנו עדיין מחיר הפתרון האופטימלי c_d^* ובשביל להגיע אליו ב- f_limit כדי לסיים הריצה, וכיוון ש- f_limit תגדל כל פעם ב- 1/k בכל איטרציה לפחות, מספר האיטרציות המקסימלי שידרשו הינו (1/k) / c_d^* כלומר kc_d^* איטרציות.
(ii) כיוון ש- f_limit יגדל כל פעם ב- 1/k לפחות, גם בו צמתים באותה שכבה שה- f שלהם בהפרש פחות מ- 1/k יפתחו באותה איטרציה (בשונה מ- *IDA) אם לאחד מהם יהיה מסלול למטרה יתכן שנלך אליו גם אם הוא לא האפטימלי (תלוי בהרצת DFS) לכן ייתכן שנחזיר מסלול בעל עלות > c_d^*, כיוון שאנחנו מפתחים רק צמתים בהפרש f פחות מ- 1/k, גם כאן זה יהיה ההפרש מעלות האופטימלית, שוב יותר מזה היה חורג מה- f_limit ולא היה ממשיך לפתח.
סה"כ נקבל עלות מקסימלית עבור האלג' החדש A2 של c_d^* + 1/k לכל היותר (ולא שווה כלומר עלות > c_d^* + 1/k). לכן:

$$\varepsilon(A2,D) = path\_cost(A2(D)) - c_d^* < c_d^* + 1/k - c_d^* < 1/k$$

ה- (i) מתקיים כי במרחב D^q ערכי w (העלות) יקטנו לכל היותר בפחות מ- 1/k כלומר כל שתי עלות של קשת בהפרש של פחות מ- 1/k יקבלו אותה עלות חדשה, וכנ"ל לגבי ערכי היוריסטיקה הם יקטנו לכל היותר בפחות מ- 1/k, לכן כל שני ערכי היוריסטיקה בהפרש של פחות 1/k יקבלו אותו ערך עבור היוריסטיקה החדשה.

$f^q = w^q + h^q$ כלומר עבור כל צומת מתקיים כי ה- f שלו יקטן בלכל היותר ב- 1/k + 1/k = 2/k (לא כולל) וצמתים בעלי f מקורי בהפרש קטן מ- 2/k יקבלו ערך f^q חדש זהה.

נריץ *IDA על המרחב החדש, עכשיו כל איטרציה חדשה f_limit יגדל בלפחות 2/k אחרת הצמתים היו נפתחים באותה איטרציה.

נסתכל על מצב מטרה G במרחב D, היוריסטיקה קבילה בפרט מתקיים h(G)=0 ו- g(G) = c_d^* (העלות המקסימלית עבור G).

כל קשת במסלול ל- G בעלות => 1/k (בגרף המקורי), בגרף החדש היא תרד בפחות מ- 1/k לכן העלות החדשה המקסימלית של g(G) הינה $Q_k(c_d^*)$ (זה באופן מקסימלי בין אם הוא לא משתנה כי ערכי g על המסלול לא השתנו או כי רק אחד (קשת אחת) ירד בפחות מ- 1/k).

לכן העלות המקסימלית החדשה שתהיה במרחב הינה $Q_k(c_d^*)$ ו- f_limit צריכה להגיע לערך זה (לכל היותר), כפי שאמרנו בכל איטרציה f_limit תגדל בלפחות 2/k לכן מספר איטרציות מקסימלי שיבצע A3 הינו:

$(Q_k(c_d^*))$ / (2/k) = (k$Q_k(c_d^*)$) / 2

(ii) לפי החישוב החדש של פונקציית העלות מתקיים כל עלות כל קשת תרד מקסימום בפחות מ- 1/k (לא כולל 1/k). רוצים לדעת אורך המסלול האופטימלי המקסימלי האפשרי (לדעת כמה קשתות יש בו באופן מקסימלי ולחשב כמה תרד העלות האופטימלית גם כן באופן מקסימלי), במרחב D מתקיים כי עלות הפתרון האופטימלי הינה c_d^* וכל קשת במרחב הינה בעלות 1/k לפחות לכן לכל היותר יהיה לנו (1/k) / c_d^* כלומר kc_d^* קשתות במסלול זה, נשים לב כי מרחב D^q זה לא נכון כיוון שאנחנו משנים את העלות, אם עלות כל קשת הינה 1/k בדיוק אז העלות הכוללת למסלול לא משתנה, אבל אם לפחות אחד מהם יותר מעלות 1/k יכול להיות שהיא תרד, על כן, אורך מסלול מקסימלי (מספר הקשתות) במרחב D^q הינו $Q_k(\mathrm{k}c_d^*)$ (זה יהיה לכל היותר באופן הכי מקסימלי),כיוון כל קשת מהם תרד בפחות מ- 1/k לכל היותר, סה"כ ירד קטן ממש מ- (1/k)* $Q_k(\mathrm{k}c_d^*)$ מעלות הפתרון האופטימלי במרחב D, לכן עלות חדשה תהיה לכל הפחות (לא כולל): c_d^* - $(1/\mathrm{k}) * Q_k(kc_d^*)$ ויכול להיות יותר מזה (להתקרב מעלות האופטימלית של מרחב D של c_d^*) לכן הפרש זה מפתרון האופטימלי של מרחב D יניב למקסימום הפרש, לכן :

ε(A3,D) = path_cost(A3(D)) – c_d^* > c_d^* - $(1/\mathrm{k}) * Q_k(kc_d^*) - c_d^*$ > - $(1/\mathrm{k}) * Q_k(kc_d^*)$

לכן אם נסתכל על $\varepsilon(\mathrm{A3,D})$ מבחינת גודל חיובי (הפרש העלויות), זה יהיה חסם עליון על ההפרש בין שני המחירים.

2- **חיפוש מרובה סוכנים**

א- סיבה אפשרית להתנהגות זו הינה שהיוריסטיקה לא מתחשבת בצעדים קודמים שהנחש עשה ותרמו לו (לגודלו) כלומר הגדילו הסיכוי לו לנצח.
נניח שאנחנו נמצאים בצעד נוכחי שבו נאכל פירה וגודל הנחש יוגדל ב- 1, אלג' 1 שמסתכל רק על עומק 2 כלומר צעד אחד בלבד של הנחש יחזיר ∞ וכך הנחש יגדל (וסיכוי ניצוחו יגדל), אבל נניח שצעד שאחרי זה הנחש ימות (או יתרחק מהפירות), אלג' 2 שמסתכל על עומק 4 מסתכל על שני צעדים של הנחש, ורואה שהנחש ימות/יתרחק מפירות ומחזיר ערך היוריסטיקה בהתאם בלי לקחת בחשבון צעד קודם שכן תרם ושיפר הנחש מבחינת גודל שהוא מדד חשוב לניצחון, לכן יכול להיות שנאכל בצעד ראשון וצעד שני נמות או נתרחק מפירות ועדיין זה יהיה יותר טוב מפשוט התקדמות בלי אכילה כדי לא למות או להתקרב מפירות עתידיות.
לכן החזרת ∞– בהיוריסטיקה במקרה של מוות לא הגיונית והיה צריך לקחת בחשבון השיפורים הקודמים שנעשו ולהחזיר ערך מתאים לשיפור זה למשל.
נשים לב, שאלג' 1 מסתכל רק על צעד נוכחי של הנחש לכן אם הוא יוכל לאכול הוא פשוט אוכל וזהו ולא חושב יותר מדי, לכן עבור היוריסטיקה זו (h1) אלג' 1 היה יותר טוב.
באופן כללי, עבור היוריסטיקות אחרות הלקחות בחשבון מדדים אחרים שהשתפרו במהלך המשחק (החשובים לניצחון), כפי שלמדנו בכיתה הסתכלות על עומק גדול בו חושבים קדימה על צעדים עתידיים אפשריים ולקיחתם בחשבון יכול לעזור לנו מאוד בלתכנן ניצחון לכן אלג' 2 במקרה זה היה מנצח רוב הזמן.

ב- כן, כיוון שהיוריסטיקה הנ"ל מתחשבת בצעדים קודמים שנעשו ומתחשבת בתרומתם לנחש דרך התחשבות במדד length_snake לכן במקרה שניצחנו בעומק d, בהרצת מינימקס בעומק D>d אנחנו מתחשבים באורך הנחש שהגיע אליו בצעדים d/2 הראשונות שלו (וכנראה ניצח בהם), גם אם הוא מת (ימות) בעומק יותר מ- d או מתרחק מפירות בעומק יותר מ- d, אנחנו לא נחזיר ערך אקראי גרוע עבור המצב הסופי בלבד, (שאסטרטגיית צעדים אלו תוביל אליו) אלה פשוט בחישוב ההיוריסטיקה אומרים שהצעד הנוכחי לא תרם לנחש אבל כך אורכו יהיה, שהולך להגיע אליו (בין אם הוא ימות עכשיו או מתרחק מפירות) ועדיין יכול להיות שהוא ינצח באורך זה (מהלך צעדים זה). לכן אם משחק מינימקס באורך d מסתיים בניצחון כלומר אורך הנחש מספיק כדי לנצח, אנחנו לפחות נגיע לאורך זה בחישוב מינימקס עבור עומק D>d כי אנחנו מתחשבים באורך שיגיע אליו בצעדי d/2 (עומק d) ולא נחזיר ערך עבור המצב הסופי שיהיה בעומק D בלבד, ולכן גם כן ננצח.

3- **מערכות לומדות**

א- נגדיר קבוצת האימון הבאה: (כך ש- $x_i = (v_{1,i}, v_{2,i})$ כך ש- $v_{1,i}, v_{2,i}$ תכונות רציפות של Salary ∈ [0,10000] ו- Age ∈ [0,100] בהתאמה.

Salary	Age	Res סיווג
5500	21	1 (+)
6000	19	0 (-)
7000	30	1 (+)
4000	21	0 (-)

כך שנגדיר הסיווג באופן הבא :

$$\begin{cases} \text{If (salary > 5000 \&\& age > 20)} & 1 \\ \text{else} & 0 \end{cases}$$

אם ננרמל הדוגמאות יהיה לנו :

Salary	Age	Res סיווג
1/2	2/11	1 (+)
2/3	0	0 (-)
1	1	1 (+)
0	2/11	0 (-)

עבור knn כך ש- k=1 תהי דוגמא מבחן (5000,21) ננרמל אותה (1/3,2/11) נחשב מרחק אוקלידי בינה לבין הדוגמאות ונקבל שדוגמאת המבחן קרובה ביותר על הדוגמא (1) לכן נסווג אותה כמו דוגמא (1) לכן תקבל סיווג 1 (+), למרות שבפועל ה- salary שלה הינו 5000 ו- 5000>5000 אינו נכון ולכן צריכה לקבל סיווג אפס (-).

עבור ID3: כפי שלמדנו, העץ בוחר תכונות לפיצול לפי ה- IG כלומר נבחר התכונה הממקסמת את ה- IG שלנו, נשים לב שהתכונות רציפות ולכן צריך עבור כל תכונה לבחור לפי איזה סף נפצל אותם.

נתחיל בזה שה-

$$\text{Entropy(res)} = \text{Entropy(E)} = -\sum_{i=1}^{n} p(xi) * \log(p(xi)) = -1/2 * \log(1/2) – 1/2 * \log(1/2) = 1$$

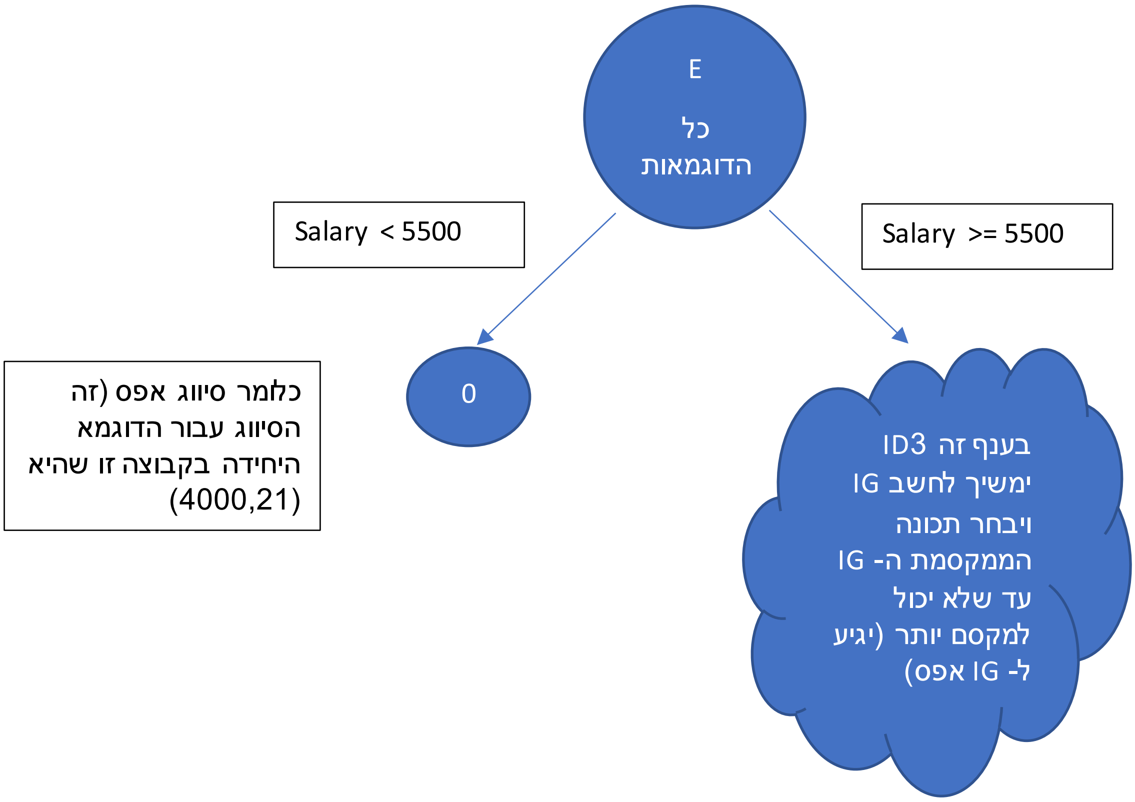
נחשב IG של התכונה Salary, נחקור את ערכי הסף האפשריים:

ערכי הסף	E(salary,res)	IG
5500	3/4 E(2,1) + 1/4 E(1,0) = 3/4 * 0.9182	1 – 0.68865
6000	2/4 E(1,1) + 2/4 E(1,1) = 1	0
7000	1/4 E(1,0) + 3/4 E(1,2) = 3/4 * 0.9182	1 – 0.68865
4000	4/4 E(2,2) = 1	0

נחשב IG של התכונה Age, נחקור את ערכי הסף האפשריים:

ערכי הסף	E(Age,res)	IG
21	3/4 E(2,1) + 1/4 E(1,0) = 3/4 * 0.9182	1 – 0.68865
19	4/4 E(2,2) = 1	0
30	1/4 E(1,0) + 3/4 E(1,2) = 3/4 * 0.9182	1 – 0.68865
21	3/4 E(2,1) + 1/4 E(1,0) = 3/4 * 0.9182	0

קיבלנו ערכי IG זהים כלומר אנחנו יכולים למקסם את ה- IG שלנו בכמה דרכים (כמה פיצולים אפשריים) אם כן ה- ID3 באופן הזה יוכל לבחור תכונה אקראית עם ערך סף אקראי מאלה שממקסימים את ה- IG והוא יפצל באופן הבא: (נניח שבחר תכונה Salary עם ערך סף 5500).



עכשיו נסתכל על דוגמא מבחן של (5400,21) מתקיים כי 21>20 && 5400>5000 לכן צריך בפועל סיווג 1 (+), אבל ID3 מסתכל בפיצול הראשון על ה- Salary ורואה כי 5400<5500 לכן מסווג אותה כ- אפס (-).

ב- נגדיר קבוצת האימון הבאה: (כך ש- $x_i = (v_{1,i}, v_{2,i})$ כך ש- $v_{1,i}, v_{2,i}$ תכונות רציפות של [0,10000] ∈ Salary ו- [0,100] ∈ Age בהתאמה.

סיווג Res	Age	Salary
1 (+)	19	4000
1 (+)	21	5500
1 (+)	22	7000
1 (+)	18	5000

הסיווג שלנו יוגדר להיות באופן הבא:

$$\begin{cases} \text{If (true) } 1 \\ \text{else } 0 \end{cases}$$

כלומר מסווג הכל כחיובי.

עבור KNN כך K=4 תהי דוגמא מובחן כלשהי (x,y) כיוון ש- K=4 וגודל קבוצת האימון הוא 4, אז 4 הדוגמאות הקרובות הינם 4 הדוגמאות בסט האימון ו- KNN יבחר לפי החלטת הרוב שלהם, כיוון שלכולם תיוג חיובי, הוא יסווג אותה כחיובית, ואכן זה הסיווג האמיתי שלה לפי אופן פעולת המסווג שהצגנו שמסווג הכל כחיובי.

עבור ID3, צריך לבנות העץ, ולבחור תכונה וערך סף לפיצול ראשון, כיוון שכל סט האימון מתויג כחיובי חישוב ה- IG עבור כל תכונה/ערך סף יניב ל- IG=0 כיוון שאין אף ערך שיפצל הדוגמאות, אם כן, יהיה עץ בעל עלה יחיד, כך שהסיווג חיובי תמיד. ואכן אז סיווג דוגמת המבחן לפי ID3 יניב לסיווג כחיובית, ושוב זה הסיווג האמיתי שלה לפי המסווג שהצגנו.

<u>**חלק ב – רטוב וכתיבת דוח**</u>

<u>**משימה: תכנות אלגוריתם למידה**</u>

מצורף הקבצים

קובץ dataset_preparation מכין את ה- data, מבצע את הטעינה מקבצי ה-csv מחלק הדוגמאות לקבצי train ו- test כפי שנבקש.
לפי תעודת הזהות שלי בחרתי dataset_4 ו- dataset_10.

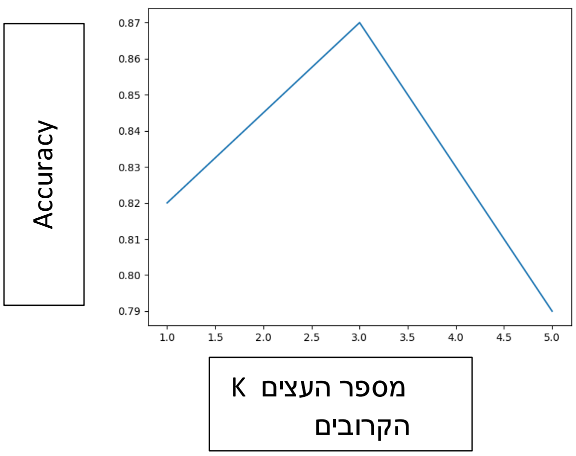
קובץ decision_tree בונה עץ החלטה, דרך הפונקציה decision_tree_build ששולחים לה הדאטא (קבוצת האימון) ובונה עץ החלטה כפי שלמדנו בכיתה, חלקת הדוגמאות נעשית דרך שאילת שאלות על ערכי הפיצ'רים ופיצול בהתאם, בחרתי כל פעם לשאול שאלה על התכונה/ערך סף שממזער את האנטרופיה (אי הוודאות). בנוסף יש את הפונקציה predict שנשלח אליה דוגמא ועץ החלטה ומסווגת הדוגמא לפי עץ ההחלטה ששלחנו (לפי השאלות בו) ומחזירה את הסיווג.

קובץ k_decision_tree, הפעלת הפונקציה decision_tree_build (מקובץ decision_tree) N פעמים כדי ללמד N עצי החלטה כפי שנתבקש. זה נעשה דרך הפונקציה n_decision_tree, כך שכל פעם מגרילים p*200 דוגמאות מסט האימון כדי ללמוד מהם בדיוק כפי שנתבקש.
בנוסף יש הפונקציה k_decision_tree אליה נשלח דוגמא ו- N עצי ההחלטה שלנו והיא מוצאת K עצי החלטה הקרובים אליה לפי מדד centroid שהוגדר, ומחשבת סיווג הדוגמא לפי כל אחד מה-K עצים ומחזירה הסיווג הסופי לפי החלטת הרוב.
בנוסף כאשר חישבתי ממוצע כל פיצ'ר ב- centroid נרמלתי הערכים, כדי שחישוב המרחק בין הדוגמא ל- centroid עבור כל עץ יהיה יותר נכון/מדויק ולא יושפע מדוגמאות בעלת ערכים גבוהים באיזשהו פיצ'ר. (כאשר נירמלתי קיבלתי דיוק יותר גבוה עבור הניסויים בחלק 2).

<u>**משימה: הרצת ניסויים**</u>

1- הערכים הדיפולטיביים שבחרתי הינם:
N=7, K=3, P=0.5

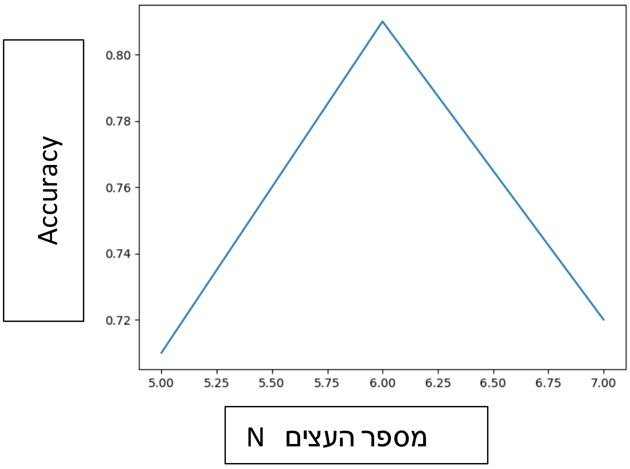
2- עבור ה- dataset_4:
ניסוי 1: N=7, K=1, P=0.5
N=7, K=3, P=0.5
N=7, K=5, P=0.5



ניתן לראות שאנחנו מקבלים דיוק מקסימלי של 0.87 כאשר K=3 ודיוקים פחות טובים עבור K=1 ו- K=5, התנהגות זו יכולה להיות כאשר אנחנו מסתכלים גם על עצים "רחוקים" כאשר K=5 כלומר דוגמאות לא רלוונטיות לנו (בדומה לאיך שלמדנו בכיתה בהגדלת K נמנע overfitting אבל תעלה

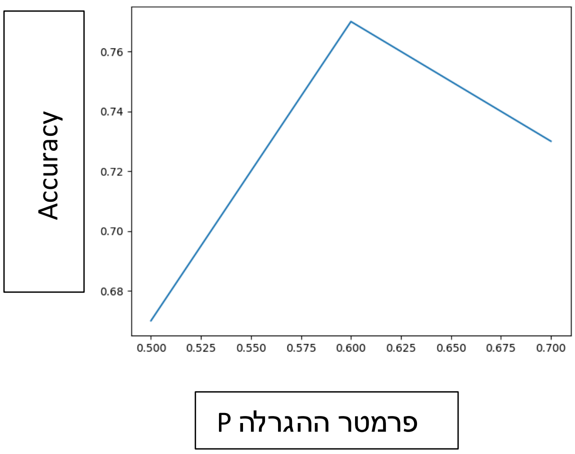
שגיאת המבחן/תוריד הדיוק בשל הסתכלותה על דוגמאות רחוקים). עבור K=1 יכול להיות כי אנחנו מסתכלים על רק דוגמא יחידה קרובה ומסווגים רק לפיה, ומתעלמים מדוגמאות אחרות (עצי החלטה אחרים) שגם הם קרובים לנו וצריך להתייחס אליהם (יש צורך בהכללה).

ניסוי 2: N=5, K=3, P=0.5
N=6, K=3, P=0.5
N=7, K=3, P=0.5



למידת יותר עצים תניב לדיוק יותר גבוה, הייתי משערכת את זה, כי אז נלמד יותר עצים, ועבור כל עץ מספר דוגמאות שווה, כך שבחרנו אותם באופן אקראי, ובנוסף נבחר מהם K העצים הכי קרובים לדוגמא ונסווג לפיהם. נשים לב שזה לא הכרחי עדיין יכול להיות שמספר עצים רב לא דווקא ישפר את הדיוק יכול להיות שבהגרלת הדוגמאות עבור מספר עצים גדול נקבל עצים כמעט זהים ואז לא בהכרח הדיוק ישתפר. (הכל תלוי בהגרלת הדוגמאות והדאטא המתקבל לכל עץ)

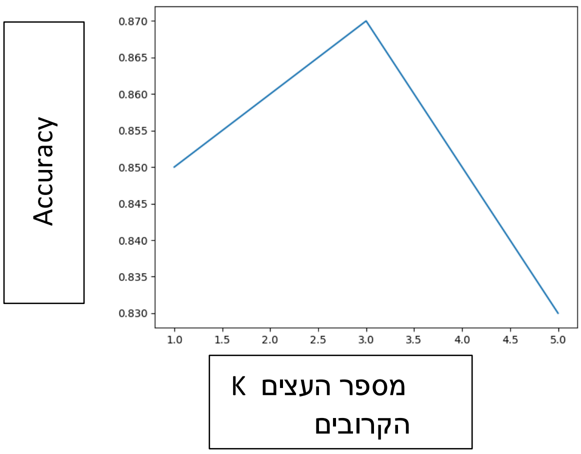
ניסוי 3: N=7, K=3, P=0.5
N=7, K=3, P=0.6
N=7, K=3, P=0.7



P הינו מדד לדוגמאות שאנו מגרילים כדי שעצי ההחלטה ילמד מהם, הייתי משערכת שכל שה- P גדול הדיוק יגדל גם כן כי אז הוא לומד על יותר דוגמאות, ניתן לראות בגרף שעבור 0.7 הדיוק ירד קצת, וזה יכול להיות בשל overfitting כלומר דיוק גבוה מאוד על קבוצת האימון כך שמוריד הדיוק על קבוצת המבחן, בנוסף יכול להיות כי הגרלת אחוז גדול מסט האימון עבור כל עץ החלטה יניב לעצי החלטה זהים (כמעט לומדים אותן דוגמאות) ולכן שוב הדיוק לא ישתפר (לא נרוויח יותר דיוק מלמידת כמה עצים).

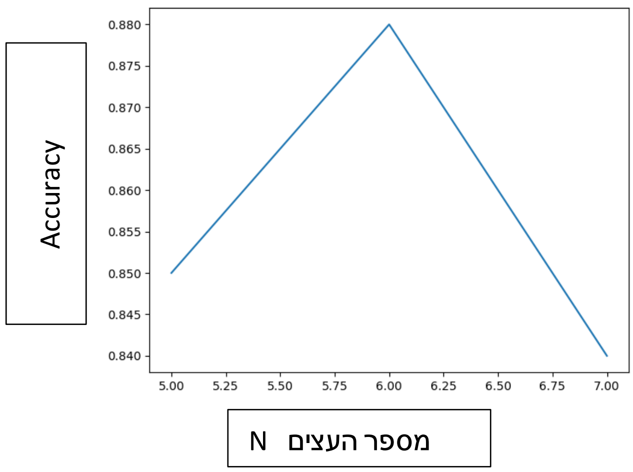
עבור ה- dataset_10: (ערכי הפרמטרים נבחרו באופן זהה, רציתי לנסות ולבדוק פעולתם גם על סט דאטא זה כדי ללמוד התנהגות פרמטרים אלה על יותר דאטא ולבחור אלו שמניבים רוב הסיכוי לדיוק גדול יותר).

ניסוי 1: N=7, K=1, P=0.5
N=7, K=3, P=0.5
N=7, K=5, P=0.5



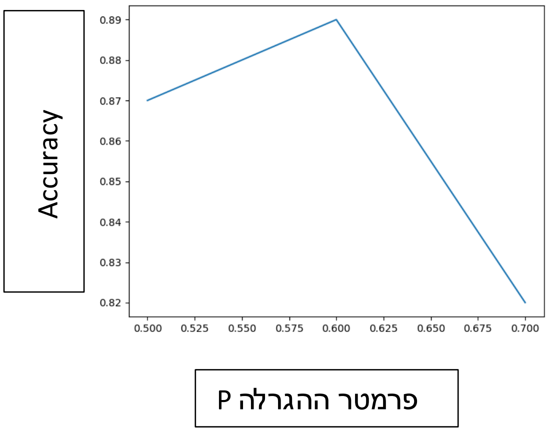
גם ב- dataset זה ניתן לראות כי הסתכלות עול יותר עצים קרובים ולהכליל אותם תניב לדיוק יותר גבוה, אבל הכללה יותר מדי תוריד הדיוק בשל הסתכלות על עצי החלטה רחוקים ולא רלוונטיים, כנ"ל אם נסתכל רק על עץ החלטה הכי קרוב ולסווג רק לפיו, בלי להתייחס לעצים אחרים שגם הם קרובים.

ניסוי 2: N=5, K=3, P=0.5
N=6, K=3, P=0.5
N=7, K=3, P=0.5



גם ב- dataset הזה ניתן לראות כי למידת יותר עצים תניב לדיוק יותר גבוה, יותר עצים כלומר יותר הגרלות של דוגמאות כל פעם ולמידה לפיהם ואז בחישוב K העצים הקרובים יהיה לנו סט גבוה ונכלל יותר של דוגמאות, עבור למידת 7 עצים הדיוק ירד קצת לפי המספרים בציר y וזה יכול לקרות אם הגרלת דוגמאות בכל עץ הניבה למשל לאותן דוגמאות כמעט בכל עץ כך שלא באמת היה לנו יותר עצים לסווג לפיהם ואז הדיוק היה פחות טוב וזה יכול לקרות כי הכל תלוי בהגרלת הדוגמאות. לכן יותר עצים תניב להגרלות דומות ברוב העצים כך שלא באמת נרוויח מלמידת יותר עצי החלטה (גודל הדאטא שעובדים עליו הינו 200 לא כך מספיק גדול לכן באיזשהו שלב יהיה לנו מספיק עצים רלוונטיים לכלל הדוגמאות ולא צריך יותר).

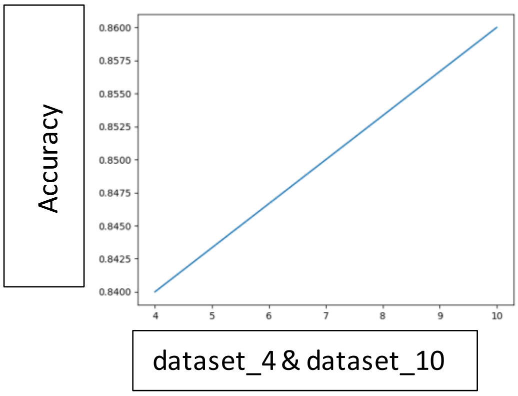
ניסוי 3: N=7, K=3, P=0.5
N=7, K=3, P=0.6
N=7, K=3, P=0.7



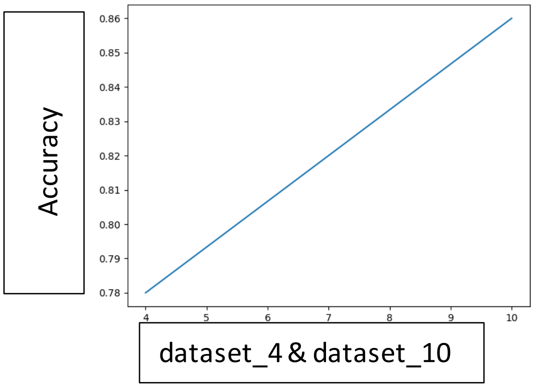
גם ב- dataset הזה ניתן לראות כי הגרלות יותר דוגמאות תניב ליותר דיוק, אבל למידה על מלא דוגמאות תוכל להוריד הדיוק כנראה בשל overfitting בדומה לקודם, כלומר כך שנכסה קבוצת האימון ונלמד אותה מאוד טוב אבל נשלם עבור דיוק קבוצת המבחן, או בשל שכל עץ מגריל ובוחר רוב הדוגמאות מסט האימון כך שנקבל עצי החלטה זהים כמעט כך שלא יועיל לנו בלמידה (כמעט אותם עצים).

<u>בחירת פרמטרים אופטמליים:</u>

עכשיו ננסה לבחור פרמטרים אופטימליים שנקבע אותם להמשך התרגיל:
נבדוק ה accuracy עבור K=6, N=3, P=0.6 , בחרתי לבדוק פרמטרים אלה יחד כי הם הניבו לדיוקים הכי טובים בניסויים למעלה, בנוסף מבחינה הגיונית אנו רוצים מספרים ממוצעים של פרמטרים, למשל עבור K לא להסתכל על הרבה עצים קרובים כי אז נכליל סיווגים לא רלוונטיים ולא רק על עץ יחיד קרוב כי אז לא נקבל דיוק בהסתכלות רק על עץ יחיד, בנוסף רוצים ללמוד N=6 עצים ולא יותר, רוצים ללמוד מספר עצים ממוצע כדי שיהיה לנו עצים שונים ולהרוויח את זה בסיווג ולא ללמוד יותר מדי ולקבל כמעט אותם עצים. בנוסף בחרתי פרמטר הגרלה של 0.6 כדי לא ללמוד יותר מדי דוגמאות מסט האימון לכל עץ שוב בשל לא לקבל overfitting ולא לקבל כמעט אותן דוגמאות בכל עץ בנוסף לא להגריל מעט דוגמאות אלה לנצל כמות הדוגמאות שיש לנו. לכן אלו הפרמטרים שכנראה יניבו לפתרון אופטימלי (כמעט).
בממוצע, אלה שהינבו לרוב הסוכוי ליותר דיוק מנסיונות שעשיתי עבור כל מיני פרמטרים יחד.



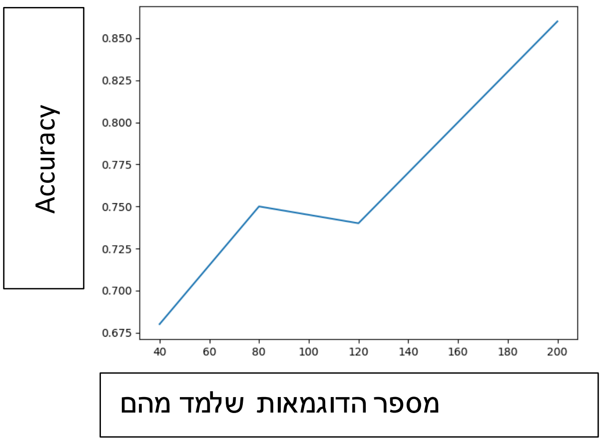
כלומר קיבלנו דיוק של 0.84 ו- 0.86 עבור dataset_4 ו- dataset_10 בהתאמה. הדיוקים אלה הכי טובים מהניסיונות שעשיתי.
עבור הרצה אחרת עם אותם פרמטרים קיבלתי:



כלומר קיבלנו דיוק של 0.78 ו- 0.86 עבור dataset_4 ו- dataset_10 בהתאמה. כנראה הדיוק הכולל הינו משהו בתחומים האלה. (כי זה תלוי בהגרלת הדוגמאות כל פעם).

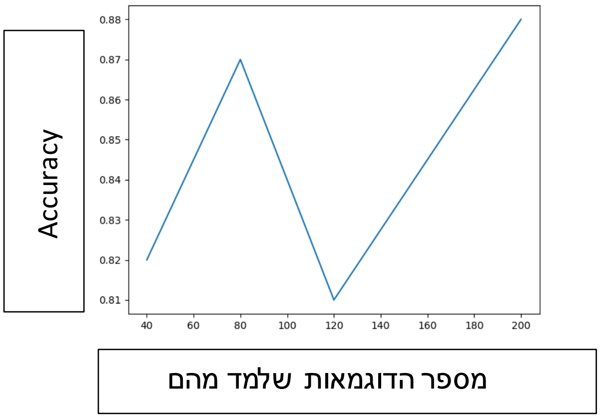
מצורף הקובץ params.py אשר בהרצתו מריץ ניסיונות אלה ומדווח על תוצאתם.

3- ראשית נקבע הפרמטרים שבחרנו בסעיף קודם, ונחלק הdata ל-5 חלקים שווים כך שכל פעם נסתכל על 40 דוגמאות חדשות.
עבור dataset_4:



הדיוק יגדל עם למידת יותר דוגמת כי המסווג יכליל יותר דוגמאות. עבור 120 דוגמאות ניתן לראות שהדיוק ירד קצת וזה יכול להיות בשל לקיחת בחשבון למשל דוגמאות רועשות כך שיש להם השפעה על הדיוק בסט של 120 דוגמאות והשפעה זו תירד/לא תשפע בהסתכלות על יותר דוגמאות, נשים לב כי לפי בחירת P עבור סט 120 למשל לא נגריל דוגמאות ויהיה לנו 6 עצי החלטה עבור אותו סט וכנראה יהיו זהים כך שמספרם לא באמת ישנה הדיוק.
עבור 200 דוגמאות אנחנו נגריל דוגמאות מהסט לפי הפרמטרים האופטימליים מקודם כיוון שבחרנו הפרמטרים כך שלא יהיה לנו overfitting והשתדלתי לבחור הפרמטרים הכי טובים לכן ניתן לראות שהמסווג מנצל 200 הדוגמאות שיש לו ולומד מהם בצורה טובה ומניב דיוק גדול.

עבור dataset_10:



גם בדאטא סט זה ניתן לראות שלמידת יותר דוגמאות יניב דיוק יותר טוב, גם פה הדיוק ירד עבור 120 דוגמאות אולי יש דוגמאות רועשות בדומה לקודם וכאשר נסתכל על סט יותר גדול השפעתם תרד. ושוב למידה עבור 200 דוגמאות תניב לדיוק המקסימלי, כיוון שבחרנו פרמטרים הכי אופטימליים בכך שננצל הדאטא שלנו באופן מקסימלי כדי לסווג נכון.

מצורף הקובץ exp.py אשר בהרצתו מריץ ניסיונות אלה ומדווח על תוצאתם.