תרגיל בית 3 – MDP,RL ומבוא ללמידה

עברו על כלל ההנחיות לפני תחילת התרגיל.

הנחיות כלליות:

- תאריך ההגשה: 25/08/2024 ב־23:59
- את המטלה יש להגיש **בזוגות בלבד.**
- יש להגיש מטלות מוקלדות בלבד. פתרונות בכתב יד לא ייבדקו.
- ניתן לשלוח שאלות בנוגע לתרגיל בפיאצה בלבד.
- המתרגל האחראי על תרגיל זה: **תומר מלניק.**
- בקשות דחיה מוצדקות (מילואים, אשפוז וכו') יש לשלוח למתרגל האחראי (אור רפאל בידוסה) בלבד.
- במהלך התרגיל ייתכן שנעלה עדכונים, למסמך הנ"ל – תפורסם הודעה בהתאם.
- העדכונים הינם מחייבים, ועליכם להתעדכן עד מועד הגשת התרגיל.
- שימו לב, התרגיל מהווה כ- 10% מהציון הסופי במקצוע ולכן העתקות תטופלנה בחומרה.
- התשובות לסעיפים בהם מופיע הסימון ✍️ צריכים להופיע בדוח.
- לחלק הרטוב מסופק שלד של הקוד
- אנחנו קשובים לפניות שלכם במהלך התרגיל ומעדכנים את המסמך הזה בהתאם. גרסאות עדכניות של המסמך יועלו לאתר. הבהרות ועדכונים שנוספים אחרי הפרסום הראשוני יסומנו כאן בצהוב. ייתכן שתפורסמנה גרסאות רבות – אל תיבהלו מכך. השינויים בכל גרסה יכולים להיות קטנים.

שימו לב שאתם משתמשים רק בספריות הפייתון המאושרות בתרגיל (מצוינות בתחילת כל חלק רטוב)
לא יתקבל קוד עם ספריות נוספות

מומלץ לחזור על שקפי ההרצאות והתרגולים הרלוונטיים לפני תחילת העבודה על התרגיל.

חלק א' – MDP ו־RL (67 נק')

רקע

בחלק זה נעסוק בתהליכי החלטה מרקובים, נתעניין בתהליך עם **אופק אינסופי** (מדיניות סטציונרית).

חלק א' - חלק היבש ✍️

שאלה 1

1. בתרגול ראינו את משוואת בלמן כאשר התגמול ניתן עבור המצב הנוכחי בלבד, כלומר $R: S \to \mathbb{R}$, למתן תגמול זה נקרא "תגמול על הצמתים" מכיוון שהוא תלוי בצומת שהסוכן נמצא בו.
בהתאם להגדרה זו הצגנו בתרגול את האלגוריתמים Value iteration ו-Policy Iteration למציאת המדיניות האופטימלית.

כעת, נרחיב את ההגדרה הזו, לתגמול המקבל את המצב הנוכחי, הפעולה לביצוע והמצב הבא שהסוכן הגיע אליו בפועל (בין אם הסוכן בחר לצעוד לכיוון הזה ובין אם לא), כלומר: $R: S \times A \times S' \to \mathbb{R}$, למתן תגמול זה נקרא "תגמול על הקשתות".

א. (2 נק') התאימו את הנוסחה של התוחלת של התועלת מהתרגול, עבור התוחלת של התועלת המתקבלת במקרה של "תגמול על הקשתות", אין צורך לנמק.

ב. (2 נק') כתבו מחדש את נוסחת משוואת בלמן עבור המקרה של "תגמול על הקשתות", אין צורך לנמק.

ג. (3 נק') נסחו את אלגוריתם Value Iteration עבור המקרה של "תגמול על הקשתות".

ד. (3 נק') נסחו את אלגוריתם Policy Iteration עבור המקרה של "תגמול על הקשתות".

הערה: בסעיפים ג' ו־ד' התייחסו גם למקרה בו $\gamma = 1$, והסבירו מה לדעתכם התנאים שצריכים להתקיים על הסביבה\mdp על מנת שתמיד נצליח למצוא את המדיניות האופטימלית.

שאלה 2

נתונים שני אנשים – "סוחט" ו-"קורבן". בכל שלב ה"סוחט" יכול:

1) "לפרוש" – לפרוש עם רווחי הסחיטה.
2) "לסחוט" – לדרוש תשלום של 1₪ בהסתברות p , ה"קורבן" יענה לדרישה. ובהסתברות $1-p$ ה"קורבן" יסרב לשלם וידווח למשטרה.

הנחות:

- לאחר שה"קורבן" דווח למשטרה, ה"סוחט" מאבד את כל הרווחים שנצברו ואינו יכול לסחוט שוב.
- לאחר שה"סוחט" מגיע לרווחים מצטברים של n₪ הוא פורש מיד.
- מטרת הסוחט היא למקסם את סכום הכסף שהוא מרוויח.
- אופק סופי, ניתן להניח שגדול מאוד $\gamma = 1$

1. (4 נק') נסחו את הבעיה כבעיית MDP עם המצבים $i = 0,1,\dots,n$ ומצב סיום T. (0 הוא מצב התחלתי) באופן ספציפי, כתבו את המצבים, הפעולות בכל מצב, ההסתברויות המעבר והתגמולים.

הערה: התגמולים חייבים להיות אי-שליליים.

2. (2 נק') האם ניתן לנסח את הבעיה כבעיית MDP עם מצב יחיד ומצב סיום? נמקו.
3. (2 נק') האם ניתן לנסח את הבעיה כבעיית MDP כאשר חלק מהתגמולים שלילים? נמקו.

4. (9 נק') נתון כי n=3.

כעת נרצה למצוא מדיניות אופטימליות ומה התועלת של המצב ההתחלתי כפונקציה של p.

בתשובתכם מצאו עבור אילו ערכי p נקבל כל מדיניות – מצאו את a ו-b כך שהמדיניות בטווח הנתון לא תשתנה. מלאו את הערכים החסרים בטבלה שבעמוד הבא **ונמקו** היטב את תשובתכם.

הערה: כאשר המדיניות של מצב i יכולה לקבל יותר מפעולה אחת יש לציין את כל הפעולות.

$\boldsymbol{a} =$ _____

$\boldsymbol{b} =$ _____

ערכי p	מדיניות	תועלות
$0 < p < \boldsymbol{a}$	$\pi_1(0) =$ _____ $\pi_1(1) =$ _____ $\pi_1(2) =$ _____ $\pi_1(3) =$ _____	$V^{\pi_1}(0) =$ _____
$\boldsymbol{a} < p < \boldsymbol{b}$	$\pi_2(0) =$ _____ $\pi_2(1) =$ _____ $\pi_2(2) =$ _____ $\pi_2(3) =$ _____	$V^{\pi_2}(0) =$ _____
$\boldsymbol{b} < p < 1$	$\pi_3(0) =$ _____ $\pi_3(1) =$ _____ $\pi_3(2) =$ _____ $\pi_3(3) =$ _____	$V^{\pi_3}(0) =$ _____

חלק ב' - היכרות עם הקוד

חלק זה הוא רק עבור היכרות הקוד, עבורו עליו במלואו וודאו כי הינכם מבינים את הקוד.

mdp.py – **אתם לא צריכים לערוך כלל את הקובץ הזה.**

בקובץ זה ממומשת הסביבה של ה-mdp בתוך מחלקת MDP. הבנאי מקבל:

- board - המגדיר את המצבים האפשריים במרחב ואת התגמול לכל מצב, תגמול על הצמתים בלבד.
- terminal_states – קבוצה של המצבים הסופיים (בהכרח יש לפחות מצב אחד סופי).
- transition_function – מודל המעבר בהינתן פעולה, מה ההסתברות לכל אחת מארבע הפעולות האחרות. ההסתברויות מסודרות לפי סדר הפעולות.
- gamma – discount factor המקבל ערכים - $\gamma \in (0,1)$.
  בתרגיל זה לא נבדוק את המקרה בו $\gamma = 1$.

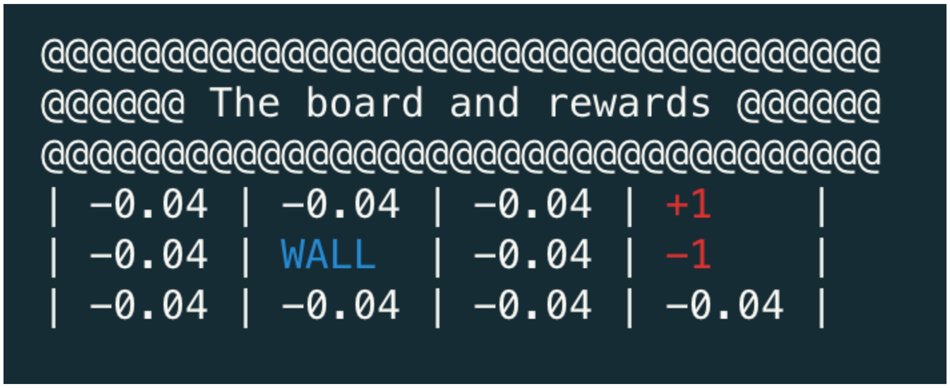
הערה: קבוצת הפעולות מוגדרת בבנאי והיא קבועה לכל לוח שיבחר.

למחלקת MDP יש מספר פונקציות שעשויות לשמש אתכם בתרגיל.

- print_rewards() – מדפיסה את הלוח עם ערך התגמול בכל מצב.
- print_utility(U) – מדפיסה את הלוח עם ערך התועלת U לכל מצב.
- print_policy(policy) – מדפיסה את הלוח עם הפעולה שהמדיניות policy נתנה לכל מצב שהוא לא מצב סופי.
- step(state, action) – בהינתן מצב נוכחי state ופעולה action מחזיר את המצב הבא באופן דטרמיניסטי. עבור הליכה לכיוון קיר או יציאה מהלוח הפונקציה תחזיר את המצב הנוכחי state.
- load_mdp(board,termina_states,transition_function,gamma) – פונקציה סטטית המקבלת שמות של קבצים המכילים את מבנה הmdp, ואת ערכה של gamma ומחזירה אוביקט (mdp

הערה: פונקציות ההדפסה שלנו משתמשות בספרית פייתון בשם termcolor בשביל לסמן מצבים סופיים באדום וקירות בכחול. אנו ממליצים להשתמש ב־integrated terminal שנמצא בIDE שלכם על מנת לראות את ההדפסות בפורמט ברור.

להלן דוגמה להדפסה:



```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@ The board and rewards @@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
| -0.04 | -0.04 | -0.04 | +1    |
| -0.04 | WALL  | -0.04 | -1    |
| -0.04 | -0.04 | -0.04 | -0.04 |
```

חלק ג' – רטוב

כל הקוד צריך להיכתב בקובץ mdp_rl_implementation.py

מותר להשתמש בספריות:

All the built-in packages in python, numpy, matplotlib, argparse, os, copy, typing, termcolor, random

עליכם לממש את הפונקציות הבאות:

- (רטוב 7 נק'): value_iteration(mdp, U_init, epsilon) – בהינתן ה-mdp, ערך התועלת ההתחלתי U_init, וחסם העליון לשגיאה מהתוחלת של התועלת האופטימלי epsilon מריץ את האלגוריתם value iteration ומחזיר את U המתקבל בסוף ריצת האלגוריתם. הערך עבור קירות הינו None. **TODO**
- (רטוב 7 נק'): get_policy(mdp, U) – בהינתן ה-mdp וערך התועלת U (המקיים את משוואת בלמן) מחזיר את המדיניות (במידה וקיימת יותר מאחת, מחזיר אחת מהן). המדיניות עבור קירות ומצבים סופיים הינה None. **TODO**
- (רטוב 7 נק'): policy_evaluation(mdp, policy) – בהינתן ה-mdp, ומדיניות policy מחזיר את ערכי התועלת לכל מצב. הערך עבור קירות הינו None. **TODO**
- (רטוב 7 נק'): policy_iteration(mdp, policy_init) - בהינתן ה-mdp, ומדיניות התחלתית policy_init, מריץ את האלגוריתם policy iteration ומחזיר מדיניות אופטימלית. המדיניות עבור קירות ומצבים סופיים הינה None. **TODO**

הערה: ניתן לפתור את סעיף זה גם לפני ההרצאה והתרגול בנושא RL, כל המידע הדרוש מופיע במסמך.

אבוי! תומר, המתרגל ה"אחראי" על התרגיל, איבד את קבצי ה־board וה־transition_function וכעת אין אנו יודעים את R ואת P. למזלו, הראל - מומחה ה־RL™ , סיפר לו שהוא יודע לבצע <u>סימולציות</u>, כלומר לבצע רצף של פעולות אקראיות במרחב ה־MDP ועל הדרך לתעד את ה־Rewards המתקבלים, הפעולות שנבחרו והפעולות שהתבצעו בפועל. סימולציה אחת, אשר נקראת גם אפיזודה (episode), מוגבלת עד ל־k צעדים ומורכבת מרצפים מהצורה:

$$state, reward, action, actual\_action$$

הרצף האחרון שהתבצע באפיזודה יהיה מהצורה:

$$state, reward, None, None$$

שכן קיבלנו עליו תגמול אך לא בצענו בו פעולה.

הראל הציע לתומר להשתמש באלגוריתם RL – model based, כלומר להשתמש בסימולציות על מנת ללמוד <u>ולהעריך</u> את פונקציית התגמולים ואת מודל המעבר, ולאחר מכן להשתמש באלגוריתמים הקודמים (Value Iteration ו־Policy Iteration) על מנת לקבל את המדיניות האופטימלית ביחס ל־MDP המוערך.

לדוגמה, אם הרצנו אפיזודה אחת בלבד וקיבלנו:

$$A, 3, Up, Up$$

$$B, 2, Up, Down$$

$$A, 3, Down, Down$$

$$C, 5, Up, Up$$

$$A, 3, None, None$$

ראשית, נוכל לדעת כי $R(A) = 3, R(B) = 2, R(C) = 5$.

שנית, ניתן לראות כי מתוך כל שלוש הפעמים שהפעולה שנבחרה הייתה Up, היא אכן בוצעה פעמיים ואילו פעם אחת בוצעה הפעולה $Down$. לכן, מהרצה של אפיזודה אחת, נוכל לשערך כי אם בחרנו Up מודל המעבר מבצע את הפעולה Up בהסתברות $2/3$ ומבצע את הפעולה $Down$ בהסתברות $1/3$.

בנוסף, בכל הפעמים שהפעולה שנבחרה הייתה $Down$ (פעם אחת בלבד) היא אכן בוצעה. לכן, מהרצה של אפיזודה אחת נוכל לשערך כי אם בחרנו $Down$ מודל המעבר מבצע את הפעולה $Down$ בהסתברות $1/1 = 1$.

אלגוריתם זה נקרא ADP.

הערה: בתרגיל זה אתם התבקשתם ללמוד את $\Pr[actual\_action|chosen\_action]$, זו הגדרה לא שגרתית. בדרך כלל באלגוריתם ADP לומדים את $\Pr[s'|s,a]$, קרי את מודל המעבר עצמו, באופן דומה.

סיפקנו לכם בקובץ simulator.py סימולטור שיאפשר לכם לממש את האלגוריתם, אנא קראו אותו וראו דוגמת הרצה.

- (רטוב 7 נק') adp_learner(sim, num_episodes, num_rows, num_cols, actions)– בהינתן הסימולטור, מספר האפיזודות שיש להריץ ,מספר השורות, מספר העמודות ורשימת הפעולות החוקית מריץ את האלגוריתם adp ומחזיר את הR ו-P אשר התקבלו בסיום הריצה. התגמול עבור קירות הינו $None$. ניתן להניח כי num_episodes מספיק גדול כך שנספיק לראות את כל המצבים ואת כל הפעולות לאורך כלל הסימולציה. **TODO**

- ✍️ (יבש 3 נק') הריצו את הadp_leaner שלוש פעמים עם num_episodes=10,100,1000 ולאחר מכן השתמשו ב־policy_iteration על מנת להוציא את המדיניויות האופטימליות עבור ערכים אלו. צרפו את תוצאות ההרצה. האם המדיניות זהות? איזו מדיניות קרובה יותר למדיניות האידיאלית? מה היתרונות והחסרונות של הגדלת מספר הepisodes?
- ✍️ (יבש 2 נק') הציעו גרסת anytime לאלגוריתם הadp שלנו. כתבו pseudo-code עבור ההצעה שלכם, נמקו כיצד הפתרון שלכם עומד בעקרון אלגוריתמי anytime שלמדנו.

חלק ב' - מבוא ללמידה (33 נק')

✍️ חלק א' – חלק היבש (16 נק')

kNN – נעים להכיר

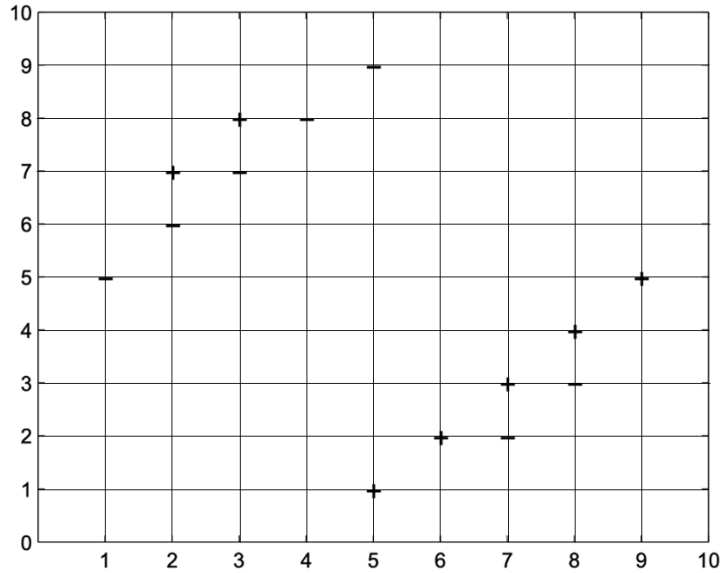
בחלק זה תכירו אלגוריתם למידה בשם kNN, או בשמו המלא k-Nearest Neighbors, כאשר ה־k הוא למעשה פרמטר!
יהי סט אימון עם n־דוגמות, $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$, כאשר $\forall i: x_i \in \mathbb{R}^d, y_i \in \mathcal{Y}$.
כלומר הדוגמות הינן וקטורים d־ממדיים והתגיות הינן מדומיין כלשהו, הבעיה היא בעיית קלסיפיקציה (סיווג).
אם לא נאמר אחרת, הקלסיפיקציה תהיה בינארית, כלומר $\mathcal{Y} = \{-, +\}$.
עבור כל דוגמה בסט האימון, ניתן להסתכל על הכניסה ה־i בווקטור כעל הfeature ה־i של הדוגמה, קרי כל דוגמה x_i מיוצגת על ידי d־ערכים: $f_1(x_i), f_2(x_i), \dots, f_d(x_i)$.
תהליך ה"אימון" של האלגוריתם הוא טריוויאלי – פשוט שומרים את סט האימון במלואו.
תהליך הסיווג הוא גם פשוט למדי – כאשר רוצים לסווג דוגמה <u>מסט המבחן</u> מסתכלים על k השכנים הקרובים ביותר שלה במישור הd־ממדי <u>מבין הדוגמות בסט האימון</u>, ומסווגים את הדוגמה על פי הסיווג הנפוץ ביותר בקרב k השכנים.
על מנת להימנע משוויון בין הסיווגים, נניח בדרך כלל כי k־אי זוגי, או שנגדיר היטב שובר שוויון.
אם לא נאמר אחרת, במקרה של שוויון בקלסיפיקציה בינארית, נסווג את הדוגמה כחיובית $+$.

<u>שאלות הבנה</u>

א. (3 נק') כאמור, בתהליך הסיווג אנו בוחרים עבור הדוגמה את הסיווג הנפוץ ביותר של k השכנים הקרובים ביותר, אולם עלינו להגדיר את פונקציית המרחק עבור קביעת סט שכנים זה.
שתי פונקציות מרחק נפוצות הינן מרחק אוקלידי ומרחק מנהטן.
1) עבור איזה ערכים של d, k נקבל שאין תלות בבחירה בין פונקציות המרחק הנתונות (נמקו)
2) עבור בעיית קלסיפיקציה בינארית תנו דוגמה <u>פשוטה</u> לערכי d, k, סט אימון ודוגמת מבחן בה השימוש בכל אחת מפונקציות המרחק הנ"ל משנה את סיווג דוגמה המבחן.

מעתה, אלא אם כן צוין אחרת, נשתמש במרחק <u>אוקלידי</u>.

נתונה קבוצת האימון הבאה, כאשר $d = 2$:



3) (1 נק') איזה ערך של k עלינו לבחור על מנת לקבל את הדיוק המרבי <u>על קבוצת האימון</u>? מה יהיה ערך זה? (דוגמה יכולה להיות שכנה של עצמה).
4) (1 נק') עבור איזה ערך של k נקבל מסווג $majority$ של קבוצת האימון? קרי כל דוגמת מבחן תקבל את הסיווג הנפוץ של כלל קבוצת האימון?
5) (2 נק') נמקו מדוע שימוש בערכי k גדולים או קטנים מדי יכול להיות גרוע עבור קבוצת הדגימות הנ"ל.
6) (2 נק') שרטט את גבול ההחלטה של 1-nearest neighbor עבור הגרף

מתפצלים ונהנים

(7 נק') כידוע, בעת סיווג של דוגמת מבחן על ידי עץ החלטה, בכל צומת בעץ אנו מחליטים לאיזה צומת בן להעביר את דוגמת המבחן על ידי ערך סף v שמושווה לfeature של הדוגמה. לפעמים ערך הסף <u>קרוב מאוד</u> לערך הfeature של דוגמת המבחן. היינו רוצים להתחשב בערכים "קרובים" לערך הסף בעת סיווג דוגמת מבחן, ולא לחרוץ את גורלה של הדוגמה לתת־עץ אחד בלבד; לצורך כך נציג את האלגוריתם הבא:

יהיו עץ החלטה T, דוגמת מבחן $x \in \mathbb{R}^d$, ווקטור $\varepsilon \in \mathbb{R}^d$ המקיים $\forall i \in [1, d]: \varepsilon_i > 0$.
כלל אפסילון־החלטה שונה מכלל ההחלטה הרגיל שנלמד בביתה באופן הבא:
נניח שמגיעים לצומת בעץ המפצל לפי ערכי התכונה i, עם ערך הסף v_i.
אם מתקיים $|x_i - v_i| \leq \varepsilon_i$ אזי ממשיכים **בשני** המסלולים היוצאים מצומת זה, ואחרת ממשיכי לבן המתאים בדומה לכלל ההחלטה הרגיל. לבסוף, מסווגים את הדוגמה x בהתאם לסיווג הנפוץ ביותר של הדוגמאות הנמצאות בכל העלים אליהם הגענו במהלך הסיור על העץ (במקרה של שוויון – הסיווג ייקבע להיות $True$).

יהא T עץ החלטה לא גזום, ויהא T' העץ המתקבל מ־T באמצעות גיזום מאוחר שבו הוסרה הרמה התחתונה של T (כלומר כל הדוגמות השייכות לזוג עלים אחים הועברו לצומת האב שלהם).
הוכיחו\הפריכו: **בהכרח** קיים ווקטור ε כך שהעץ T עם כלל אפסילון־החלטה והעץ T' עם כלל ההחלטה הרגיל יסווגו כל דוגמת מבחן ב$\mathbb{R}^d$ בצורה זהה.

חלק ב' - היכרות עם הקוד

רקע

חלק זה הוא רק עבור היכרות הקוד, עבורו עליו במלואו וווודאו כי הינכם מבינים את הקוד.
בחלק של הלמידה, נעזר ב $dataset$, הדאטה חולק עבורכם לשתי קבוצות: קבוצת אימון train.csv וקבוצת מבחן test.csv.
בכלל, קבוצת האימון תשמש אותנו לבניית המסווגים, וקבוצת המבחן תשמש להערכת ביצועיהם.

בקובץ utils.py תוכלו למצוא את הפונקציות הבאות לשימושכם:
load_data_set, create_train_validation_split, get_dataset_split
אשר טוענות/מחלקת את הדאטה בקבצי ה־csv למערכי np.array (קראו את תיעוד הפונקציות).

הדאטה של ID3 עבור התרגיל מכיל מדדים שנאספו מצילומים שנועדו להבחין בין גידול שפיר לגידול ממאיר. כל דוגמה מכילה 30 מדדים כאלה, ותווית בינארית **diagnosis** הקובעת את סוג הגידול (0=שפיר, 1=ממאיר). כל התכונות (מדדים) רציפות . העמודה הראשונה מציינת האם האדם חולה (M) או בריא (B). שאר העמודות מציינות כל תכונות רפואיות שונות של אותו אדם (התכונות מורכבות ואינכם צריכים להתייחס למשמעות שלהן כלל).

תיקיית $ID3-dataset$:
- תיקיה זו אלו מכילה את קבצי הנתונים עבור $ID3$.

קובץ $utils.py$:
- קובץ זה מכיל פונקציות עזר שימושיות לאורך התרגיל, כמו טעינה של $dataset$ וחישוב הדיוק.
- בחלק הבא יהיה עליכם לממש את הפונקציות $l2\_dist$ ו$accuracy$. קראו את תיעוד הפונקציות ואת ההערות הנמצאות תחת התיאור **TODO**.

קובץ $unit\_test.py$:
- קובץ בדיקה בסיסי שיכול לעזור לכם לבדוק את המימוש.

קובץ $DecisionTree.py$:
- קובץ זה מכיל 3 מחלקות שימושית לבניית עץ $ID3$ שלנו.
  - המחלקה $Question$: מחלקה זו מממשת הסתעפות של צומת בעץ. היא שומרת את התכונה ואת הערך שלפיהם מפצלים את הדאטה שלנו.
  - המחלקה $DecisionNode$: מחלקה זו מממשת צומת בעץ ההחלטה.
    הצומת מכיל שאלה $Question$ ואת שני הבנים $true\_branch, false\_branch$ כאשר $true\_branch$ הוא הענף בחלק של הדאטה שעונה $True$ על שאלת הצומת (הפונקציה $match$ של ה$Question$ מחזירה $True$).
    ו־ $false\_branch$ הוא הענף בחלק של הדאטה שעונה $False$ על שאלת הצומת (הפונקציה $match$ של ה$Question$ מחזירה $False$).
  - המחלקה $Leaf$: מחלקה זו מממשת צומת שהוא עלה בעץ ההחלטה. העלה מכיל לכל אחד מהמחלקות בדאטה את מספר הדוגמאות בעלה עבור כל מחלקה (למשל: $\{'B' : 5, 'M' : 6\}$).

קובץ $ID3.py$:

- קובץ זה מכיל את המחלקה של $ID3$ שתצטרכו לממש חלקים ממנה, עיינו בהערות ותיעוד המתודות.

קובץ $ID3\_experiments.py$:

- קובץ הרצת הניסויים של ID3, הקובץ מכיל את הניסויים הבאים, שיוסברו בהמשך:
  $cross\_validation\_experiment, basic\_experiment$

חלק ג' – חלק רטוב ID3 (17 נק')

עבור חלק זה מותר לכם להשתמש בספריות הבאות:
All the built in packages in python, sklearn, pandas ,numpy, random, matplotlib, argparse, abc, typing.

אך כמובן שאין להשתמש באלגוריתמי הלמידה, או בכל אלגוריתם או מבנה נתונים אחר המהווה חלק מאלגוריתם למידה אותו תתבקשו לממש.

1. (5 נק') השלימו את הקובץ utils.py ע"י מימוש הפונקציות $l2\_dist$ ו $accuracy$.
קראו את תיעוד הפונקציות ואת ההערות הנמצאות תחת התיאור **TODO**.
(הריצו את הטסטים המתאימים בקובץ $unit\_test.py$ לוודא שהמימוש שלכם נכון).
שימו לב! בתיעוד ישנן הגבלות על הקוד עצמו, אי־עמידה בהגבלות אלו תגרור הורדת נקודות.
בנוסף, שנו את ערך הID בתחילת הקובץ מ־123456789 למספר תעודת הזהות של אחד מהמגישים.

2. (12 נק') אלגוריתם ID3:

a. השלימו את הקובץ ID3.py ובכך ממשו את אלגוריתם $ID3$ כפי שנלמד בהרצאה. **TODO**
שימו לב שכל התכונות רציפות. אתם מתבקשים להשתמש בשיטה של חלוקה דינמית המתוארת בהרצאה. כאשר בוחנים ערך סף לפיצול של תכונה רציפה, דוגמאות עם ערך השווה לערך הסף משתייכות לקבוצה עם הערכים הגדולים מערך הסף. במקרה שיש כמה תכונות אופטימליות בצומת מסוים בחרו את התכונה בעלת האינדקס המקסימלי.
כלל המימוש הנ"ל צריך להופיע בקובץ בשם $ID3.py$, באזורים המוקצים לכך.
(השלימו את הקוד החסר אחרי שעיינתם והפנמתם את הקובץ $DecisionTree.py$ ואת המחלקות שהוא מכיל).

b. ממשו את $basic\_experiment$ שנמצאת ב $ID3\_experiments.py$ **TODO**
והריצו את החלק המתאים ב $main$ ציינו בדו"ח את הדיוק שקיבלתם. ✍️

הוראות הגשה

- ✓ הגשת התרגיל תתבצע אלקטרונית בזוגות בלבד.
- ✓ הקוד שלכם ייבדק (גם) באופן אוטומטי ולכן יש להקפיד על הפורמט המבוקש. הגשה שלא עומדת בפורמט לא תיבדק (ציון 0).
- ✓ המצאת נתונים לצורך בניית הגרפים אסורה ומהווה עבירת משמעת.
- ✓ הקפידו על קוד קריא ומתועד. התשובות בדוח צריכות להופיע לפי הסדר.
- ✓ יש להגיש קובץ zip יחיד בשם AI3_<id1>_<id2>.zip (ללא סוגריים משולשים) שמכיל:

- קובץ בשם AI_HW3.PDF המכיל את תשובותיכם לשאלות היבשות.
- קבצי הקוד שנדרשתם לממש בתרגיל <u>**ואף קובץ אחר:**</u>
  - קובץ – utils.py
  - בחלק של עצי החלטה – ID3.py, ID3_experiments.py
  - בחלק של mdp וRL- mdp_rl_implementation.py