תרגיל בית 3 – MDP ומבוא ללמידה

מגישים- רות גוטקוביץ ת.ז 209287085

ניקיטה ליסוקון ת.ז 332684190

עברו על כלל ההנחיות לפני תחילת התרגיל.

הנחיות כלליות:

- תאריך ההגשה: 26/01/23 ב23:59
- 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: **אור רפאל בידוסה.**
- בקשות דחיה מוצדקות (מילואים, אשפוז וכו') יש לשלוח למתרגל האחראי (ספיר טובול) 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 (30 נק')

רקע

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

חלק א' - חלק היבש ✍️

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

   T(S) = neighbors of S

$$U^{\pi}(s) = E_{\pi}\left[\sum_{t=0}^{\infty} \sum_{a \in \mathbb{A}(S_t)} \sum_{s' \in \mathrm{T}(S_t)} \gamma^t R(S_t, a, s') | S_0 = s \right]$$

   ב. (1 נק') כתבו מחדש את נוסחת משוואת בלמן עבור המקרה של "תגמול על הקשתות", אין צורך לנמק.

$$U(s) = \max_{a \in \mathbb{A}(s)} \sum_{s' \in \mathrm{T}(s)} P(s'|a,s) R(s,a,s') + \gamma \max_{a \in \mathbb{A}(s)} \sum_{s'} P(s'|a,s) U(s')$$

ג. (2 נק') נסחו את אלגוריתם Value Iteration עבור המקרה של "תגמול על הקשתות".

רעיון כללי: הפעלת משוואת בלמן מסעיף ב' באופן איטרטיבי:

1)ננחש תועלת התחלתית לכל קשת

2)עבור כל מצב נחשב את הצד הימני במשוואת בלמן ובצד השמאלי במשוואת בלמן נמצא תועלת מקסימלית של קשתות ממצב הנוכחי

3)נעדכן את כל התועלות של הקשתות על פי החישובים שביצענו

4)נבדוק תנאי עצירה. אם מתקיים, נחזיר את פונקציית התועלת שהתקבלה

*פסאודו קוד של אלגוריתם:

T(S) = neighbors of S

function VALUE-ITERATION(mdp, ϵ) **returns** a utility function
inputs: mdp, an MDP with states S, actions $A(s)$, transition model $P(s' \mid s, a)$,
rewards $R(s)$, discount γ
ϵ, the maximum error allowed in the utility of any state
local variables: U, U', vectors of utilities for states in S, initially zero
δ, the maximum change in the utility of any state in an iteration

repeat
$U \leftarrow U'$; $\delta \leftarrow 0$
for each state s **in** S **do**
$U'[s] \leftarrow \max_{a \in \mathbb{A}(s)} \sum_{s' \in \mathrm{T}(s)} P(s'|a,s)R(s,a,s') + \gamma \max_{a \in \mathbb{A}(s)} \sum_{s'} P(s'|a,s)U(s')$
if $|U'[s] - U[s]| > \delta$ **then** $\delta \leftarrow |U'[s] - U[s]|$
Until $\delta <= \epsilon(1-\gamma)/\gamma$
return U

אם $\gamma = 1$, בתאוריה ניתן לבצע אינסוף צעדים ולקבל אינסוף תגמולים. אם נגדיר תגמולים שליליים על קשתות אז נבריח אלגוריתם לבצע כמות סופית של צעדים (קשת בעלת תגמול הגבוה ביותר היא בעלת תגמול שלילי הכי קרוב לאפס).

ניתן גם להגביל כמות צעדים המותרת באלגוריתם, אבל נתון שמערכת שלנו בעלת אופק אינסופי

ד. (2 נק') נסחו את אלגוריתם Policy Iteration עבור המקרה של "תגמול על הקשתות".

רעיון כללי: שיפור מדיניות MEU באופן איטרטיבי:

1)ננחש מדיניות התחלתית π_0

2)נחשב את התועלת המתקבלת על ידי מדיניות זו, עבור כל קשת

3)נעדכן את המדיניות להיות מדיניות MEU הנובעת מהתועלות על קשתות המתקבלות

*פסאודו קוד של אלגוריתם נשאר אותו קוד כמו בתרגול כי שינוי נוצר רק בפונקציה U:

function POLICY-ITERATION(mdp) **returns** a policy
  inputs: mdp, an MDP with states S, actions $A(s)$, transition model $P(s' \mid s, a)$
  local variables: U, a vector of utilities for states in S, initially zero
    π, a policy vector indexed by state, initially random

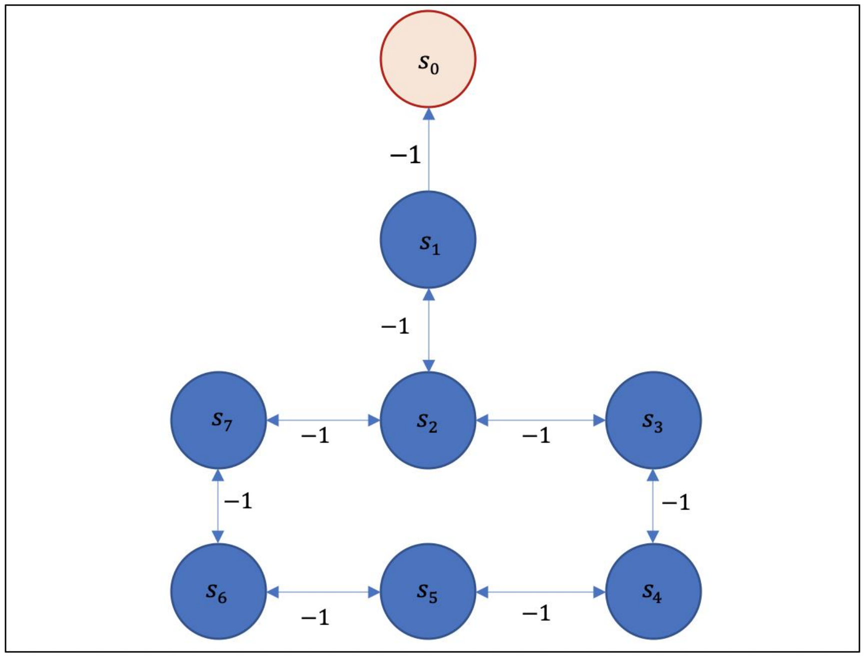
  repeat
    $U \leftarrow$ POLICY-EVALUATION(π, U, mdp)
    $unchanged? \leftarrow$ true
    for each state s **in** S **do**
      if $\max_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\ U[s'] > \sum_{s'} P(s' \mid s, \pi[s])\ U[s']$ **then do**
        $\pi[s] \leftarrow \operatorname{argmax}_{a \in A(s)} \sum_{s'} P(s' \mid s, a)\ U[s']$
        $unchanged? \leftarrow$ false
  until $unchanged?$
  return π

אם $\gamma = 1$, בתאוריה ניתן לבצע אינסוף צעדים ולקבל אינסוף תגמולים. אם נגדיר תגמולים שליליים על קשתות אז נכריח אלגוריתם לבצע כמות סופית של צעדים (קשת בעלת תגמול הגבוה ביותר היא בעלת תגמול שלילי הכי קרוב לאפס).

ניתן גם להגביל כמות צעדים המותרת באלגוריתם, אבל נתון שמערכת שלנו בעלת אופק אינסופי

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

נתון הגרף הבא:



נתונים:

- $\gamma = 1$ (Discount factor).
- אופק אינסופי.
- $S = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$ – קבוצת המצבים – מתארים את מיקום הסוכן בגרף.
- $S_G = \{s_0\}$ – קבוצת המצבים הסופיים.
- קבוצת הפעולות לכל מצב (על פי הגרף), לדוגמא: $A(s_2) = \{\uparrow, \rightarrow, \leftarrow\}$.
- תגמולים ("תגמול על הקשתות"):

$$\forall s \in S \backslash S_G, a \in A(s), s' \in S: \quad R(s, a, s') = -1$$

- מודל המעבר הוא דטרמיניסטי, כלומר כל פעולה מצליחה בהסתברות אחת.

ה. (יבש 4 נק') הרץ את האלגוריתם Value iteration שכתבת על הגרף הנתון. ומלא את הערכים בטבלה הבאה, כאשר $\forall s \in S: U_0(s) = 0$. (ייתכן שלא צריך למלא את כולה).

	$U_0(s_i)$	$U_1(s_i)$	$U_2(s_i)$	$U_3(s_i)$	$U_4(s_i)$	$U_5(s_i)$	$U_6(s_i)$	$U_7(s_i)$	$U_8(s_i)$
s_1	0	-1	-1	-1	-1	-1	-1		
s_2	0	-1	-2	-2	-2	-2	-2		
s_3	0	-1	-2	-3	-3	-3	-3		
s_4	0	-1	-2	-3	-4	-4	-4		
s_5	0	-1	-2	-3	-4	-5	-5		
s_6	0	-1	-2	-3	-4	-4	-4		
s_7	0	-1	-2	-3	-3	-3	-3		

ו. (יבש 4 נק') הרץ את האלגוריתם Policy iteration שכתבת על הגרף הנתון. ומלא את הערכים בטבלה הבאה, כאשר המדיניות ההתחלתית π_0 מופיעה בעמודה הראשונה בטבלה. (ייתכן שלא צריך למלא את כולה).

	$\pi_0(s_i)$	$\pi_1(s_i)$	$\pi_2(s_i)$	$\pi_3(s_i)$	$\pi_4(s_i)$	$\pi_5(s_i)$	$\pi_6(s_i)$	$\pi_7(s_i)$	$\pi_8(s_i)$
s_1	↑	↑	↑	↑					
s_2	↑	↑	↑	↑					
s_3	←	←	←	←					
s_4	↑	↑	↑	↑					
s_5	→	→	→	→					
s_6	→	→	↑	↑					
s_7	↓	→	→	→					

חלק ב' - היכרות עם הקוד

חלק זה הוא רק עבור היכרות הקוד, עבורו עליו במלואו ווודאו 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.

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 4 נק'): value_iteration(mdp, U_init, epsilon) – בהינתן ה-mdp, ערך התועלת ההתחלתי U_init, וחסם העליון לשגיאה מהתוחלת של התועלת האופטמילי epsilon מריץ את האלגוריתם value iteration ומחזיר את U המתקבל בסוף ריצת האלגוריתם. **TODO**

- (רטוב 4 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). **TODO**

- (רטוב 3 נק') q_learning (mdp, init_state,...) – בהינתן ה־mdp, מצב התחלתי init_state, ושאר הפרמטרים הדרושים עבור האלגוריתם, מריץ את האלגוריתם Qlearning ומחזיר את הQtable אשר התקבלה בסיום הריצה. **TODO**

  שימו לב! נזכיר כי אלגוריתם Qlearening הינו אלגוריתם ActiveRL-modelfree ועל כן לא אמור היה לקבל את הMDP כפרמטר אלא לקבל סימולטור של הסביבה.

  לא ניתנים לנו פונקציית המעברים של הסביבה והתועלות מתקבלות כפלט מהסביבה כתוצאה מסימלוץ ריצה.

  - עליכם להתחיל מטבלת Qtable המלאה באפסים.
  - כזכור לכם מההרצאה, עדכון ערך תא ב־Qtable מבוצע על ידי הנוסחה הבאה:

$$Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$$

  בתרגיל זה α הינו הפרמטר המועבר לפונקציה בשם learning_rate.

  - כזכור מההרצאה – עבור אלגוריתם זה נצטרך לבצע סימולציות, ביצוע כל סימולציה נקרא "אפיזודה" (episode).

  כל סימולציה תתחיל מ־init_state (המועבר כפרמטר לפונקציה) ולאחר מכן תבצע רצף פעולות – אשר נגמר כאשר הגענו למצב סופי או לאחר max_steps צעדים – הקצר

מבניהם.

בהינתן שאנו נמצאים במצב s בסימולציה עלינו לבחור פעולה על פי כלל־החלטה.

עבור סעיף זה נשתמש בכלל החלטה בשם $\varepsilon - greedy$:

בכל פעם שנרצה לבצע פעולה בסימולציה נגריל ערך המתפלג יוניפורמית בתחום $[0,1]$.

אם הערך שקיבלנו גדול ממש מ־ε נבחר את הפעולה המניבה ערך מקסימלי למצב s על פי ה־Qtable הנוכחי (אם יש כמה פעולות עם ערך מקסימלי נבחר אחת באופן שרירותי).

אם הערך שקיבלנו קטן או שווה ל־ε נבחר פעולה רנדומלית באופן יוניפורמי מכל הפעולות.

בסיום כל אפיזודה (ולא בסיום כל צעד של הסימולציה) נעדכן את ערך ה־ε לפי הקוד הבא:

```
# Reduce epsilon (because we need less and less exploration)
epsilon = min_epsilon + (max_epsilon - min_epsilon)*np.exp(-decay_rate*episode)
```

כאשר episode זה מספר האפיזודה שבעת סיימנו להריץ (מתחילים מ־0).

אתם רשאים להעתיק אותו.

זהו המקום היחיד בו נשתמש בפרמטרים max\min_epsilon ,decay_rate.

יש להתחיל את הריצה עם ערך ה־ε המועבר לפונקציה בפרמטר epsilon.

- (רטוב 2 נק') q_table_policy_extraction(mdp,qtable) –

בהינתן ה-mdp, והטבלה Qtable החזר את המדיניות המתאימה לטבלה.

אם ישנן כמה פעולות עם ערך מקסימלי, בחר אחת שרירותית.

- ✍️ (יבש 3 נק') אור הריץ את האלגוריתם Qlearning על mdp עבורו לכל מצב יש ערך reward חיובי, וQtable המאותחלת לאפסים עבור כל מצב ופעולה.

בסיום הרצת האלגוריתם הוא הדפיס את טבלת הQtable וראה כי חלק מהערכים של המצבים הינם 0 עבור פעולות מסוימות. הסבר כיצד מקרה זה ייתכן.

מצב זה ייתכן כאשר מבצעים מספר לא מספיק גדול של סימולציות. זאת אומרת אף פעם לא הגענו למצבים האלה בסימולציות שביצענו לכן הם לא שונו מערכם התחלתי אפס

main.py – דוגמת הרצה לשימוש בכל הפונקציות.

בתחילת הקובץ אנו טוענים את הסביבה משלושה קבצים:
board, terminal_states, transition_function
ויוצרים מופע של הסביבה (mdp).

- שימו לב, שברגע הקוד ב-main לא יכול לרוץ מכיוון שאתם צריכים להשלים את הפונקציות הרלוונטיות ב-mdp_rl_implementation.py .
- בנוסף, על מנת לראות את הלוח עם הצבעים עליכם להריץ את הקוד בIDE לדוגמה PyCharm.

הסעיפים הבאים הינם בונוס (5 נקודה לציון התרגיל)

על מנת לקבל את הבונוס יש לממש את **שתי הפעולות** – אחרת, יש להשאיר את הפעולות לא ממומשות.

- policy_evaluation(mdp, policy) – בהינתן ה-mdp, ומדיניות policy מחזיר את ערכי התועלת לכל מצב. **TODO**

- policy_iteration(mdp, policy_init) - בהינתן ה-mdp, ומדיניות התחלתית policy_init, מריץ את האלגוריתם policy iteration ומחזיר מדיניות אופטימלית. **TODO**

חלק ב' - מבוא ללמידה (70 נק')

✍ חלק א' – חלק היבש (28 נק')
ענו על חלק זה במלואו בדוח.

1. (12 נק') נגדיר דאטה סט $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$ שבו n דוגמאות מתויגות עם סיווג בינארי $y_i \in \{0,1\}$.
כל דוגמה היא וקטור תכונות המורכב משתי תכונות רציפות $x_i = (v_{i_1}, v_{i_2})$.
הניחו כי קיים מסווג מטרה $f(x): R^2 \rightarrow \{0,1\}$ שאותו אנו מעוניינים ללמוד (הוא אינו ידוע לנו) וכן שהדוגמאות ב־D עקביות עם מסווג המטרה (כלומר שאין דוגמאות רועשות ב־D).
בסעיפים הבאים, עבור KNN, הניחו פונק' מרחק אוקלידי.
כמו כן, הניחו שאם בעת סיווג של נקודה קיימות נקודות במרחב כך שעבורן יש מספר דוגמאות במרחק זהה, קודם מתחשבים בדוגמאות עם ערך v_1 מקסימלי ובמקרה של שוויון בערך של v_1, מתחשבים קודם בדוגמאות עם ערך v_2 מקסימלי.
הניחו כי אין דוגמאות זהות לחלוטין (כלומר גם עם ערך v_1 זהה וגם עם ערך v_2 זהה).
בKNN נקודה אינה שכנה של עצמה.
בכל סעיף, **הציגו מקרה המקיים את התנאים המוצגים בסעיף, הסבירו במילים, וצרפו תיאור גרפי (ציור המתאר את המקרה (הכולל לפחות תיאור מסווג המטרה והדוגמאות שבחרתם)** . סמנו דוגמאות חיוביות בסימן '+' (פלוס) ודוגמאות שליליות בסימן '-' (מינוס). בכל אחת מתתי הסעיפים הבאים אסור להציג מסווג מטרה טריוויאלי, דהיינו שמסווג כל הדוגמאות כחיוביים או כל הדוגמאות כשליליים.

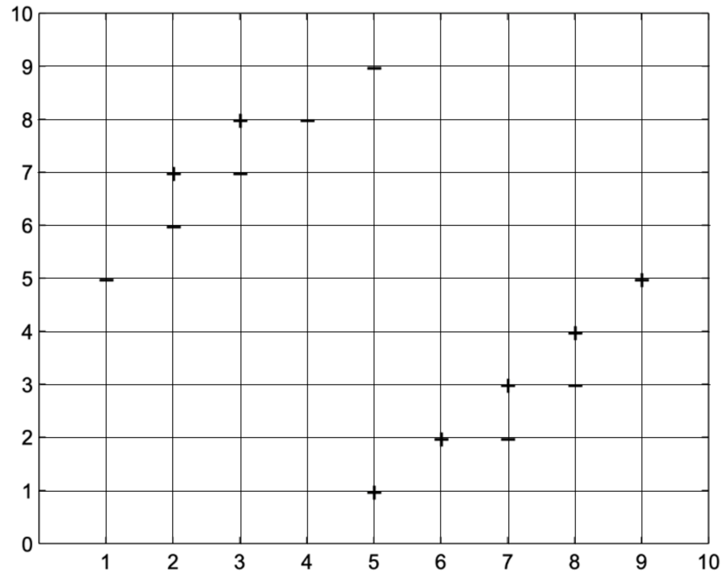
א. (3 נק') הציגו מסווג מטרה $f(x): R^2 \rightarrow \{0,1\}$ וקבוצת אימון בעלת לכל היותר 10 דוגמאות כך שלמידת עץ ID3 תניב מסווג אשר עונה נכון עבור כל דוגמת מבחן אפשרית (כלומר יתקבל מסווג המטרה), אך למידת KNN תניב מסווג שעבורו קיימת לפחות דוגמת מבחן אחת עליה הוא יטעה, לכל ערך K שייבחר.

ב. (3 נק') הציגו מסווג מטרה $f(x): R^2 \rightarrow \{0,1\}$ וקבוצת אימון בעלת לכל היותר 10 דוגמאות כך שלמידת מסווג KNN עבור ערך K מסוים תניב מסווג אשר עונה נכון עבור כל דוגמת מבחן אפשרית (כלומר יתקבל מסווג המטרה), אך למידת עץ ID3 תניב מסווג אשר עבורו קיימת לפחות דוגמת מבחן אפשרית אחת עליה הוא יטעה.

ג. (3 נק') הציגו מסווג מטרה $f(x): R^2 \rightarrow \{0,1\}$ וקבוצת אימון בעלת לכל היותר 10 דוגמאות כך שלמידת מסווג KNN עבור ערך K מסוים תניב מסווג אשר עבורו קיימת לפחות דוגמת מבחן אפשרית אחת עליה הוא יטעה, וגם למידת עץ ID3 תניב מסווג אשר עבורו קיימת לפחות דוגמת מבחן אחת אפשרית עליה הוא יטעה.

ד. (3 נק') הציגו מסווג מטרה $f(x): R^2 \rightarrow \{0,1\}$ וקבוצת אימון בעלת לכל היותר 10 דוגמאות כך שלמידת מסווג KNN עבור ערך K מסוים תניב מסווג אשר עונה נכון עבור כל דוגמת מבחן אפשרית (כלומר יתקבל מסווג המטרה), וגם למידת עץ ID3 תניב מסווג עונה נכון עבור כל דוגמת מבחן אפשרית (כלומר יתקבל מסווג המטרה).

2. (9 נק') בשאלה נשתמש במסווג k-nearest neighbour באמצעות מרחק אוקלידי, במשימת סיווג בינארי.
אנו מגדירים את הסיווג של נקודת המבחן להיות הסיווג של רוב ה-k השכנים הקרובים ביותר (שימו לב שבשאלה זו נקודה **יכולה** להיות שכנה של עצמה).
במקרה של שוויון נחזיר True.



א. (3 נק') איזה ערך של k ממזער את שגיאת האימון עבור קב' הדגימות הנ"ל? מהי שגיאת האימון כתוצאה מכך? שרטטו את גבול ההחלטה של k-nearest neighbor עבור הk הנ"ל.
ב. (3 נק') נמקו מדוע שימוש בערכי k גדולים או קטנים מדי יכול להיות גרוע עבור קבוצת הדגימות הנ"ל.
ג. (3 נק') קראו על Leave-One-Out Cross Validation בקישור הבא:
https://www.statology.org/leave-one-out-cross-validation/
אילו ערכים של k ממזערים את שגיאת Leave-One-Out Cross Validation עבור קב' הדגימות? מהי השגיאה שנוצרה?

3. (7 נק') 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 בכיתה באופן הבא:
נניח שמגיעים לצומת בעץ המפצל לפי ערכי התכונה i, עם ערך הסף v_i.
אם מתקיים $|x_i - v_i| \le \varepsilon_i$ 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 בעל 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.

תשובה- נפריך את הטענה-

נניח בשלילה כי הטענה נכונה. אזי יהי 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. תהי דוגמת מבחן $x \in \mathbb{R}^d$. אזי לפי ההנחה מתקיים כי העץ T והעץ T' יסווגו את דוגמת המבחן בצורה זהה. כלומר, בעת המעבר על העץ T סיווגנו את הדוגמה x בהתאם לסיווג הנפוץ ביותר של הדוגמאות הנמצאות בכל העלים בהם ביקרנו בעת המעבר על העץ. נחלק למקרים:

- נניח כי בעת המעבר בעץ T ביקרנו בצומת בו מתקיים- $|x_i - v_i| \le \varepsilon_i$ אזי בצומת זה מופעל כלל אפסילון- החלטה שלפיו אנו ממשיכים בשני המסלולים היוצאים מצומת זה. אזי הסיווג בעץ T יקבע לפי הסיווג הנפוץ ביותר של כלל הדוגמאות בכל העלים שעברנו בהם במהלך הסיור על העץ. לכן, בעץ T נקבל סיווג שונה מהעץ T'.
- בעת נניח כי בעת המעבר על העץ T לא ביקרנו באף צומת בו מתקיים- $|x_i - v_i| \le \varepsilon_i$. אזי לכל צומת V שעברנו בו במהלך הסיור בעץ הפעלנו את כלל החלטה הרגיל. לפי הנלמד בכיתה, עץ שעבר גיזום הינו "טוב" יותר מהעץ מהעץ ללא הגיזום (כלומר הדיוק בסיווג השתפר) אזי ניתן לומר כי בשני העצים דוגמת המבחן x לא תסווג בצורה זהה.

כלומר בשני המקרים נקבל כי הטענה אינה נכונה ולכן לא 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, עברו עליו במלואו וודאו 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 למערבי 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).

<u>תיקיית $ID3-dataset$:</u>
- תיקיה זו אלו מכילה את קבצי הנתונים עבור $ID3$.

<u>קובץ $utils.py$:</u>
- 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**.

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

<u>קובץ $DecisionTree.py$:</u>
- קובץ זה מכיל 3 מחלקות שימושית לבניית עץ $ID3$ שלנו.
  - <u>המחלקה $Question$:</u> מחלקה זו מממשת הסתעפות של צומת בעץ. היא שומרת את התכונה ואת הערך שלפיהם מפצלים את הדאטה שלנו.
  - <u>המחלקה $DecisionNode$:</u> 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 $False$).
  - <u>המחלקה $Leaf$:</u> מחלקה זו מממשת צומת שהוא עלה בעץ ההחלטה. העלה מכיל לכל אחד מהמחלקות בדאטה את מספר הדוגמאות בעלה עבור כל מחלקה (למשל: $\{‘B’: 5, ‘M’: 6\}$).

<u>קובץ $ID3.py$:</u>
- קובץ זה מכיל את המחלקה של $ID3$ שתצטרכו לממש חלקים ממנה, עיינו בהערות ותיעוד המתודות.

<u>קובץ $ID3\_experiments.py$:</u>
- קובץ הרצת הניסויים של ID3, הקובץ מכיל את הניסויים הבאים, שיוסברו בהמשך:
  $cross_{validation_{experiment}}, basic_{experiment}$

חלק ג' – חלק רטוב ID3 (42 נק')

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

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

4. (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.

5. (25 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 <u>האינדקס המקסימלי</u>.
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. ✍

תשובה- הדיוק שהתקבל עבור הרצת חלק זה הינה- 94.69%

6. (8 נק') **גיזום מוקדם.**
פיצול צומת מתקיים כל עוד יש בו יותר דוגמאות מחסם המינימום m, כלומר בתהליך בניית העץ מבוצע "גיזום מוקדם" כפי שלמדתם בהרצאות. שימו לב כי פירוש הדבר הינו שהעצים הנלמדים אינם בהכרח עקביים עם הדוגמאות. לאחר סיום הלמידה (של עץ יחיד), הסיווג של אובייקט חדש באמצעות העץ שנלמד מתבצע לפי רוב הדוגמאות בעלה המתאים.

✍ **a.** (3 נק') הסבירו מה החשיבות של הגיזום באופן כללי ואיזה תופעה הוא מנסה למנוע?

תשובה- חשיבות הגיזום היא הגבלת <u>התאמת יתר (overfitting)</u> בכך נתעלם מדגימות שגורמות לרעש. הגיזום נעשה ע״י הסתכלות על כל קבוצת מדגם, ואם יש בה לכל היותר m דגימות יוצאות דופן משאר הקבוצה, נהפוך את הקבוצה לעלה המתאים לסיווג תת הקבוצה הגדולה. בכך אנו מעלים את שגיאת האימון אבל מקטינים את שגיאת המבחן.

.b (5 נק') עדכנו את המימוש בקובץ $ID3.py$ כך שיבצע גיזום מוקדם כפי שהוגדר בהרצאה. הפרמטר $min\_for\_pruning$ מציין את המספר המינימלי בעלה לקבלת החלטה, קרי יבוצע גיזום מוקדם אם ורק אם מספר הדוגמות בצומת קטן שווה לפרמטר הנ"ל. **TODO**

.c סעיף זה בונוס (5 נקודה לציון התרגיל):

שימו לב, זהו סעיף יבש ואין צורך להגיש את הקוד שכתבתם עבורו.

בצעו כיוונון לפרמטר M על קבוצת האימון:

1. בחרו לפחות חמישה ערכים שונים לפרמטר M.
2. עבור כל ערך, חשבו את הדיוק של האלגוריתם על ידי $\mathrm{K-fold\ cross\ validation}$ על קבוצת האימון בלבד.

כדי לבצע את חלוקת קבוצת האימון ל- K קבוצות יש להשתמש בפונקציה sklearn.model_selection.KFold עם הפרמטרים $\mathrm{n\_split} = 5$, $\mathrm{shuffle} = \mathrm{True}$ ו־$\mathrm{random\_state}$ אשר שווה למספר תעודת הזהות של אחד מהשותפים.

✍ .i השתמשו בתוצאות שקיבלתם כדי ליצור גרף המציג את השפעת הפרמטר M על הדיוק. צרפו את הגרף בדו"ח. (לשימושכם הפונקציה $util\_plot\_graph$ בתוך הקובץ $utils.py$).

✍ .ii הסבירו את הגרף שקיבלתם. לאיזה גיזום קיבלתם התוצאה הטובה ביותר ומהי תוצאה זו?

תם סעיף הבונוס, הסעיף הבא הינו סעיף <u>חובה</u>.

✍ **.d** (4 נק') השתמשו באלגוריתם ID3 עם הגיזום המוקדם כדי ללמוד מסווג מתוך **כל** קבוצת האימון ולבצע חיזוי על קבוצת המבחן.
השתמשו בערך ה־M האופטימלי שמצאתם בסעיף c. (ממשו $best\_m\_test$ שנמצאת ב $ID3\_experiments.py$ והריצו את החלק המתאים ב $main$). ציינו בדו"ח את הדיוק שקיבלתם. האם הגיזום שיפר את הביצועים ביחס להרצה ללא גיזום בשאלה 5?
<u>הערה:</u> בסעיף זה אם לא מימשתם את סעיף c השתמשו בערך $M = 50$.

תשובה- הדיוק שקיבלתי 97.35% . (השתמשתי בערך M=50) .
זהו שיפור ביחס להרצת התוכנית ללא הגיזום- אז התקבל דיוק של 94.69%.

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