תרגיל בית 1 – **KNN ,ANN**

**תאריך הגשה: 29.4.2021**

נתוני הקבצים:

נתונים ארבעה קבצי csv (שני זוגות) אשר כוללים נתונים מספריים:  
  
Hananya1.csv ו- Hananya2.csv,

Hashmal1.csv ו- Hashmal2.csv.

כל קובץ כולל בשורה הראשונה את מספר הנקודות בקובץ והמספר 128.

כל נקודה מורכבת מ 132 מספרים:

Y X Scale Angle v[1]… v[128].

ארבעת המספרים הראשונים הם "השם" של הנקודה ו 128 המספרים האחרים הם הנקודה במרחב 128 ממדים.

לכל זוג של קבצים יש לבנות את מבנה הנתונים מהקובץ הראשון ולהשתמש בנקודות של הקובץ השני בתור שאילתות.

משמעות נתוני הקבצים:

כפי שאולי שמתם לב לכל קובץ נלווית תמונה עם אותו שם.

האפליקציה שממנה נוצרו הקבצים היא התאמת תמונות. בהינתן זוג תמונות שבהם רואים בערך את אותה סצנה, אנו מעוניינים להתאים בין נקודה בתמונה הראשונה לנקודה מהתמונה השנייה שבה רואים את אותו מקום.

האלגוריתם סורק את התמונה, מוצא נקודות מיוחדות ומתאר סביבה מסביב לנקודה על ידי הווקטור באורך 128 מספרים. X ו Y הם כמובן מיקום הנקודה בתמונה.

על ידי מציאת הנקודה עם הווקטור הכי קרוב בעזרת אלגוריתם ה ANN נמצא מועמד לנקודה התואמת.

עבור כל נקודה ישנה נקודה שהכי קרובה אליה, אולם לא לכל הנקודות יש נקודה תואמת בתמונה (יכול להיות משום שאין חפיפה בין התמונות או שהתהליך שסרק וחיפש נקודות מיוחדות לא חשב שבן הזוג של הנקודה מיוחד מספיק).

לכן, יש להחליט האם הנקודה מתאימה או לא.

אופציה ראשונה לכך היא על ידי הגדרת נקודות כתואמות אם המרחק ביניהם במרחב 128 ממדי הוא קטן מסף כלשהו. אולם, רעיון זה לא עבד טוב.

לכן, הציעו את שיטת היחס הבאה:

נמצא את שני השכנים הכי קרובים לנקודה במבנה הנתונים ונחשב את היחס בין המרחקים (מרחק ראשון חלקי מרחק שני). אם המרחק לשכן הראשון הוא הרבה יותר קטן מהמרחק לשכן השני (יחס קטן מ 0.8 לדוגמא) אז ניקח את הזוג. אם לא, אז לא ניקח אותו.   
דוגמה לפסאדו קוד:

nearest\_n, second\_nearest = kneighbors(sample, k=2)

ratio = nearest\_n /second\_nearest

if ratio < 0.8 return sample, nearest\_n

עם מבנה הנתונים שתממשו ניתן בקלות לנסות את השיטה הזו. אפשר גם לראות עד כמה השיטה מצליחה לזהות התאמות נכונות על ידי סימון ההתאמות על התמונות והסתכלות בתמונות.

בקורס למדנו על שתי שיטות של ANN:

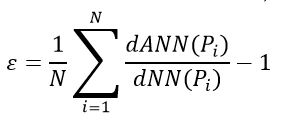
* Locality Sensitive Hashing (LSH)
* Randomized KD-Trees (RKDT).

פרמטרים:

לכל אחד משני האלגוריתמים יש שני פרמטרים שיש לתת לאלגוריתם בעת בניית מבנה הנתונים.   
  
ב- LSH ישנם K ו-L: K הוא מספר האי-שיוונים (חתכים) שיש לחשב כדי ליצור את מפתח ה- hash ואילו L הוא מספר טבלאות ה- hash שיש לבנות.  
  
ב- RKDT ישנם N0 ו- L0: N0הוא מספר הנקודות המקסימאלי שנמצא בכל עלה של העץ ו- L0 הוא מספר העצים שיש לבנות.

כל מבנה נתונים מסוג זה נבחן בשתי דרכים:

1. **מהירות ריצה:** השוואת מהירות הריצה של אלגוריתם ה-ANN אל מול האלגוריתם הלינארי KNN. נבדוק את זמן הריצה של כל אלגוריתם עבור הפעולה של kneighbors על הסט בזוג הסדור שלא הוכנס לתוך מבנה הנתונים (לדוגמא אם השתמשתם ב-hashmalX כסט אימון והכנסתם אותו למבנה הנתונים, נבדוק את זמן הריצה עבור hashmalY).
2. **דיוק:** כיוון שמדובר על ANN הוא לא תמיד מוצא את השכן הקרוב ביותר.

אם נגדיר את dNN(P) כמרחק לשכן הקרוב ביותר לנקודה P ואת dANN(P) כמרחק לנקודה שהחזיר האלגוריתם (היחס בין המרחקים תמיד גדול/שווה 1).   
  
נגדיר את השגיאה הממוצעת ε כך:

מטלות לביצוע (חובה):

בתרגיל זה אתם נדרשים לממש **אחד** ממבני הנתונים **(לבחירתכם)** של אלגוריתם ANNואלגוריתם לינארי KNN פשוט לצורך בדיקת נכונות והאצה.

בנו מחלקה עבור אלגוריתם ה-ANN עם מבנה הנתונים אותו בחרתם לממש.   
המחלקה תכיל את המתודות הבאות (שימו לב: אלו **לא בהכרח** כל המתודות שיש לבנות. ניתן להוסיף מתודות לפי הצורך, לבחירתכם):

* ANN() – מתודת הבנאי (כאן יוגדרו ה-hyper parameters)
* fit() – מתודה המכניסה את סט האימון למבנה הנתונים.
* kneighbors() – מתודה זו תקבל סט דגימות חדשות ותחזיר עבור כל דגימה את k השכנים הקרובים ביותר אליה ומרחקם ממנה.

**\*** באופן דומה בנו מחלקה עבור האלגוריתם הלינארי KNN.

* + - 1. מימוש אלגוריתם לינארי KNN. (15 נקודות)
      2. מימוש אלגוריתם ANN על ידי בחירה של אחד ממבני הנתונים. (15 נקודות)
      3. בהינתן k=2 – יש לבדוק את נכונות האלגוריתם ANN שמימשתם באמצעות [שיטת היחס](#שיטת_היחס) הנ"ל.
* הציגו את התוצאות שקיבלתם והסבירו אותן.
* השתמשו בספריית pillow והציגו עבור 10 התוצאות הטובות ביותר את השכנים הקרובים יותר שמצאתם על גבי התמונות, סמנו פיקסלים מקבילים בין הזוגות. כל זוג שכנים צבעו בצבע שונה. מהסתכלות בתוצאות ציינו אילו מההתאמות נכונות ואילו לא.ראו דוגמה מטה. (15 נקודות)
  + - 1. בהתאם למבנה הנתונים אותו בחרתם לממש עבור אלגוריתם ה-ANN (LSH או RKDT), מצאו את הפרמטרים אשר יביאו לתוצאה האופטימלית (השגיאה הקטנה ביותר אליה תצליחו להגיע).
* בחרו לכל פרמטר 10 ערכים ומצאו את הערכים האופטימליים באמצעות grid search.  
  (15 נקודות)
  + - 1. כתבו מתודה המקבלת ערך נמוך של ε (לדוגמא 0.1) ותחזיר את 5 זוגות הפרמטרים עבורם אלגוריתם ה-ANN רץ הכי מהר.
* הציגו את ערכי הפרמטרים וזמני הריצה של כל הזוגות בגרף. (15 נקודות)
  + - 1. עבור זוג ערכי הפרמטרים הכי מהיר (שהתקבל בסעיף 5), הריצו את אלגוריתם ה-ANN   
         10 פעמים והראו שמתקבלות תוצאות שונות (דיוק וזמן ריצה) בכל הרצה.
* הציגו את התוצאות בוויזואליזציה מתאימה. (10 נקודות)
  + - 1. הריצו את האלגוריתם המקביל ב-sklearn ([NearestNeighbors](https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.NearestNeighbors.html)). מודל לזה משתמש ב-KD Tree כאלגוריתם המסייע, אך הפונקציונליות שלו תואמת לזו של האלגוריתם שלכם. בצעו למודל זה אופטימיזציה לפרמטרים באמצעות grid search והשוו את התוצאות (דיוק וזמן ריצה) עם התוצאה הטובה ביותר שקיבלתם בסעיף 6. (15 נקודות) (15 נקודות)

**בונוסים (לא חובה):**

* + - 1. ממשו את מבנה הנתונים אותו לא בחרתם לממש בסעיף החובה, כולל סעיפים 2-4. (20 נקודות)
      2. ממשו מתודה נוספת עבור אלגוריתם ANN אשר מקבל רדיוס מרחק מסוים ומחזירה את השכנים הקרובים ביותר בתחום רדיוס זה. נגדיר את הרדיוס להיות 1.5\*d(NN).   
         השוו את התוצאות [למתודה radius\_neighbors הקיימת ב-sklearn](https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.NearestNeighbors.html#sklearn.neighbors.NearestNeighbors.radius_neighbors)  . (10 נקודות)

שימו לב: **עבור כל סעיף מסעיפי החובה ו/או הבונוס –** עליכם לתעד את שלבי המימוש שלכם באמצעות הערות בקוד + markdown cells לתוצאות ומסקנות.

דוגמה עבור סעיף 3:

**קווים מנחים:**

נא לקרוא היטב את החלק הבא, טיב העבודה והציונים יהיו תלויים בסעיפים הבאים:

הגשה:

* מחברת Jupyter notebook בפורמט html, ipynb.
* ההגשה היא בזוגות (פורום למציאת שותפים קיים במודל).
* תאריך הגשה: 29.4.2021.   
  הגשה באיחור (ללא סיבה מוצדקת שתאושר מראש) תגבה 3- נקודות עבור כל יום איחור.   
  לאחר חמישה ימי איחור, העבודות לא תיבדקנה.

קוד:

* הקוד צריך לרוץ ללא אזהרות או שגיאות.
* תיעוד טוב הוא **קריטי**.
* השתמשו בחבילות מוכרות ובעלות הסברים מפורשים.
* במידה והתקנתם ספריות כלשהן מעבר לאלו שהוצגו בתרגולים, נא לפרט זאת בדו"ח.
* כאשר אתם מציירים גרפים חובה להוסיף את הדברים הבאים: שמות צירים ברורים, כותרת ומקרא אם יש צורך.
* נא לציין את סעיפי התרגיל גם בקוד.
* השתמשו בשמות איברים משמעותיים.
* אל תשמשו במילים שמורות.
* נדרש מכם לכתוב לפונקציות, לולאות או כל פתרון אחרון.
* השתמשו בקבועים היכן שניתן.

**ניקוד וציון:**

ניתן לקבל יותר מ 100 נקודות עבור התרגיל.

התרגיל ינוקד על פי הקריטריונים הבאים:

* נכונות
* בהירות
* יעילות המימוש
* אלגנטיות המימוש

לימוד עצמי:

* כפי שציינו בתחילת הקורס, לימוד עצמי הוא חלק חשוב מהקורס.
* התייחסו בזהירות ובצורה ביקורתית לכל מקור מידע.
* ניתן ומומלץ להתייעץ עם שאר הסטודנטים בקורס, אך כל זוג נדרש לכתוב את עבודותיו לבד.
* סביר להניח שלא כל התוצאות והאלגוריתמים יהיו זהים.

שאלות ושעות קבלה:

* נא לפרסם את השאלות בפורום במודל, לאחר שקראתם את ההודעות הקודמות (שאלות בנושא מקצועי שישלחו במייל לא יענו).
* במידה ומעוניינים לקבוע שעת קבלה עם אחד המתרגלים – יש לשלוח לפני כן את השאלות במייל.
* בכל מקרה אחר (שאלות אישיות, בקשה להארכה עם סיבה מוצדקת וכו') יש לשלוח מייל למתרגל/ת.

בהצלחה,

צוות הקורס.