חלק ב'

3.2) לצרף קבצי פלט!

חשוב לשמור על עקביות זו משום שהחלוקה אקראית. אמנם מספר הדוגמאות המסווגות כחיוביות ושליליות קבוע בכל קבוצה עבור כל הרצה של הפונקציה (עם אותו num\_folds) אבל הדוגמאות מחולקות בין הקבוצות באופן אקראי. לכן על מנת לשמור על אותו מסווג יש להשתמש באותה החלוקה, כלומר לקרוא לפונקציה פעם אחת בלבד.

5.1) להוסיף csv!

5.2) להוסיף גרף!

5.3 + 5.4) כפי שניתן לראות מהגרף, הביצועים הטובים ביותר התקבלו עבור k = 1 והביצועים הגרועים ביותר התקבלו עבור k = 7 עם ערכי דיוק ממוצע של 0.953 ו-0.937 בהתאמה.

ראשית, נציין שהבדלי הדיוק יחסית קטנים וכן החלוקה ל-folds אקראית, לכן לא ניתן להסיק מסקנות נחרצות מדי מהתוצאות.

אם בכל זאת מנתחים את התוצאות ניתן לשער שהביצועים היו מיטביים עבור k = 1 וגרועים ביותר עבור k=7 משום שהגיוני שככל שאדם דומה יותר לאדם אחר, מבחינת נתונים גופניים, כך סביר יותר שיהיו לו בעיות רפואיות דומות לאותו אדם. לכן, במקרה שלנו, סביר שלאדם יהיו בעיות לב אם לאדם הדומה לו ביותר מבחינת נתונים גופניים יש בעיות לב, והתחשבות באנשים דומים פחות יכולה אף לפגוע בדיוק המסווג כפי שניתן לראות בתוצאותינו.

בתוצאותינו ניתן להבחין במגמת ירידה בטיב הביצוע ככל שגדל ערכו של k, דבר שמחזק את הנימוק מעלה (הדיוקים הממוצעים עבור k = 3, 5 כמעט זהים וכנ"ל עבור k = 7, 13).

7.3) להוסיף csv! להחליף בקוד את DecisionTreeClassifier(criterion="entropy") ב-Id3Estimator?

<https://pypi.org/project/decision-tree-id3/>

7.4) הניסוי שבו התקבלו התוצאות הטובות ביותר הוא הניסוי משאלה 5.3 (knn עם k = 1).

חלק ג'

* לסווג תוך משקול תוצאות מספר מסווגים או לחלופין להכריע לפי רוב.
* לבחון תכונות מיותרות או אפילו שפוגעות בסיווג.
* לנסות למצוא דברים משותפים לסיווגים שנכשלים.
* להשתמש במסווגי ספרייה טובים (גם כאלו שלא למדנו):

<https://pdfs.semanticscholar.org/c92d/38a7a76c20a317de63fb9278bb10102c758b.pdf>

* להתחשב במספר הרצות של מסווגי עצי החלטה (בשל אקראיות בחירת סדר התכונות).
* מה לגבי גיזום עצי החלטה?
* לנסות למצוא k אידאלי עבור knn (לאחר שנצליח להבין מאילו תכונות להתעלם כך שהתחשבות במספר שכנים תשפר את דיוק המסווג).
* המלצות ורמזים + הכוונה של הסגל בעמודים 5-6.