חיזוי מחלות לב באמצעות Adaboost ו-KNN

הדו"ח מציג מחקר על חיזוי מחלות לב באמצעות שני אלגוריתמים שונים לסיווג (AdaBoost ו-KNN).

רקע:

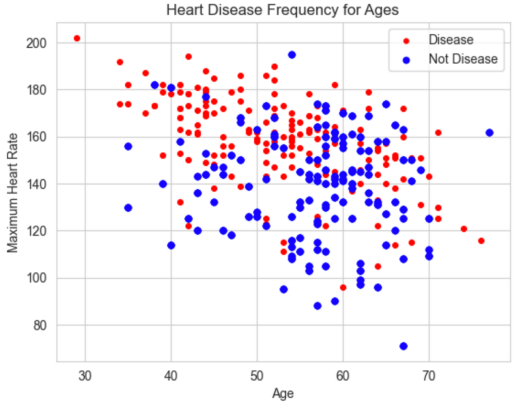
מחלות לב הן נושא בריאותי קריטי המשפיע על מיליוני אנשים ברחבי העולם. גילוי מוקדם וחיזוי מדויק של מחלות לב יכולים לשפר משמעותית את תוצאות המטופל. טכניקות למידת מכונה, כמו AdaBoost ו-KNN, הראו תוצאות מבטיחות באבחון רפואי.

המטרה העיקרית של מחקר זה היא לבנות ולהשוות שני מודלים לסיווג, AdaBoost ו-KNN, כדי לחזות נוכחות או היעדרות של מחלת לב על סמך מאפיינים קליניים ולא קליניים.

מערך נתונים:

מערך הנתונים המשמש למחקר זה מכיל מידע אנונימי על המטופלים השונים בהקשר למחלות לב. מערך הנתונים מורכב ממספר מאפיינים כאשר בין היתר כוללים: גיל, מין, לחץ דם, רמות כולסטרול ועוד. משתנה היעד (target) מצביע על נוכחות של מחלת לב (1) או בהיעדר מחלת לב (0). להלן דוגמא של חלק מהנתונים:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	62	0	0	140	268	0	0	160	0	3.6	0	2	2	0
1	58	1	2	132	224	0	0	173	0	3.2	2	2	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
4	62	0	0	150	244	0	1	154	1	1.4	1	0	2	0
5	58	0	0	170	225	1	0	146	1	2.8	1	2	1	0
6	61	1	0	120	260	0	1	140	1	3.6	1	1	3	0
7	66	0	3	150	226	0	1	114	0	2.6	0	0	2	1



התפלגות החולים לפי גיל ודופק מקסימלי.

חלוקת הנתונים:

- מטריצה X: מטריצת התכונת 'X' כוללת מספר משתני קלט המתארים את התכונות הקליניות והלא-קליניות של המטופלים. כל שורה ב-'X' מתאימה למטופל בודד, וכל עמודה מייצגת תכונה ספציפית.
- וקטור y: משתנה היעד 'y' מייצג את התוצאה הבינארית שאנו שואפים לחזות, ומצביע על נוכחות (1) או היעדר (0) של מחלת לב עבור כל חולה. כל רכיב ב-'y' מתאים לתווית היעד של המטופל המתאים במטריצת התכונה

'X'. משתנה היעד 'y' משמש במהלך שלב אימון המודל כדי ללמוד את הקשר בין תכונות הקלט ומשימת הסיווג הבינארי (חיזוי מחלות לב).

לאחר שהנתונים מעובדים מראש והמודל מאומן, 'X' ישמש כקלט למודל כדי ליצור תחזיות על נתונים חדשים, בלתי נראים, ו-'y' ישמש להערכת הדיוק והביצועים של המודל. על ידי השוואת הערכים החזויים עם תוויות היעד בפועל, אנו יכולים להעריך את יעילות המודל בחיזוי מדויק של נוכחות או היעדרות של מחלות לב בהתבסס על תכונות הקלט הנתונות.

<u>עיבוד הנתונים:</u>

הנתונים שקיבלנו עברו עיבוד מקדים של כדי להבטיח את המהימנות והאיכות של נתוני הקלט.

כחלק מהעיבוד המקדים ביצענו את השלבים הבאים:

- <u>סטנדריזציה של התכונות:</u> סטנדרטיזציה היא צעד מכריע כדי להביא את כל התכונות לקנה מידה משותף, ולמנוע מכל תכונה מסוימת לשלוט בתהליך אימון המודל בשל גודלה הגדול יותר. לשם כך, התכונות עוברות טרנספורמציה כך שבסופו של דבר הממוצע הוא 0 וסטיית התקן היא 1. התכונות הסטנדרטיות מבטיחות השוואה הוגנת בין תכונות שונות במהלך שלב המודלים.
- <u>טרנספורמציה של וקטור y:</u> על מנת להשתמש באלגורתימים בצורה נוחה יותר בחרנו להציג חולים במחלת לב עם הערך 1 ואילו אנשים בריאים עם הערך 1-.
- <u>פיצול הנתונים:</u> כדי להעריך את ביצועי ההכללה של המודל בצורה יעילה, מערך הנתונים המעובד מראש מחולק לתת-קבוצות אימון ובדיקה.

על ידי טיפול קפדני בעיבוד מקדים של נתונים, אנו שואפים לייעל את הביצועים והאמינות של המודלים לחיזוי מחלות לב. הנתונים המעובדים מראש מוכנים כעת לאימון מודלים, הערכה והשוואת ביצועים.

<u>AdaBoost:</u>

בפרוייקט זה נעשה שימוש ביישום מותאם אישית (את האלגוריתם כתבנו בעצמנו) של AdaBoost, תוך שימוש בעצי החלטה כמסווגים חלשים.

כל עץ החלטה מורכב מתכונה אחת וסף לקבלת החלטות בינאריות.

הרעיון המרכזי מאחורי AdaBoost הוא לשלב לומדים חלשים (בדרך כלל עם רמה אחת בלבד) בכדי ליצור מודל חיזוי חזק ומדויק. במהלך הריצה הוא ישלב את הלומדים החזקים הללו למסווג לומד חזק.

- <u>משקלים ראשוניים-</u> בהתחלה, לכל נקודת נתוני אימון מוקצה משקל שווה (N/1, כאשר N הוא המספר הכולל של נקודות הנתונים).
- <u>אימון לומדים חלשים</u>- הלומד החלש הראשון (מעריך בסיס) מאומן על נתוני האימון באמצעות משקלים ראשוניים אלו. הוא מנסה לסווג את הנתונים על ידי אופטימיזציה של הביצועים שלהם על פי קריטריון ספציפי.
- <u>שגיאה משוקללת-</u> שגיאה של הלומד החלש הראשון מחושבת, בהתחשב בנקודות הנתונים המשוקללות. לנקודות נתונים שסווגו בצורה שגויה יש משקל גבוה יותר בחישוב השגיאה, מה שמדגיש את החשיבות של הדגימות שסווגו בצורה שגויה.
- <u>משקל מסווג-</u> משקל (אלפא) מחושב עבור הלומד החלש המאומן על סמך השגיאה שלו. שגיאה נמוכה יותר מובילה לערך אלפא גבוה יותר, מה שמצביע על חשיבות גבוהה יותר של הלומד החלש בהרכב הסופי.
- <u>עדכון משקלים-</u> המשקלים של נקודות הנתונים שסווגו בצורה שגויה גדלים , בעוד שהמשקל של נקודות המסווגות כהלכה יורד.
- <u>חזרה-</u> התהליך חוזר על עצמו למספר מוגדר של איטרציות או עד שאין שיפור. כל איטרציה יוצרת לומד חלש חדש, ומשקולות נקודות הנתונים מתעדכנות בהתאם.

===החיזוי הסופי נעשה על ידי שילוב של התחזיות של כל הלומדים החלשים, בשקלול לפי ערכי האלפא שלהם===

H(x) = sign(alpha_1 * h_1(x) + alpha_2 * h_2(x) + … + alpha_t * h_t(x))

אחד היתרונות המרכזיים של AdaBoost הוא היכולת שלו להתמקד בדוגמאות שסווגו בצורה שגויה במהלך האימון, מה שמוביל לרוב להכללה ולחוסן משופרים במסווג שנוצר. עם זאת, AdaBoost יכול להיות רגיש לנתונים רועשים וחריגים.

KNN:

KNN הוא אלגוריתם למידת מכונה המשמש למשימות סיווג ורגרסייה. הוא חשוב בשביל "הגישה ההפוכה", למצוא מודל פשוט שמסביר את הכל. KNN הוא איננו מודל לינארי! הוא מסתכל לוקאלית על ערכים קרובים ומוציא מספר.

זהו אלגוריתם למידה פרימיטיבי ומבוסס מופעים , כלומר אינו מניח הנחות כלשהן לגבי התפלגות הנתונים הבסיסית ומשתמש ישירות במידע על מנת לבצע תחזיות.

האלגוריתם מחשב את המרחק בין נקודות נתונים במרחב התכונה, כדי לזהות את 'k' השכנים הקרובים ביותר.

'k' מייצג את מספר השכנים הקרובים ביותר שישפיעו על החיזוי הסופי. הערך האופטימלי של 'k' הוא קריטי מכיוון ש-'k' קטן עלול לגרום להתאמת יתר, בעוד ש-'k' גדול עלול להוביל לחוסר התאמה. עבור משימות סיווג, מחלקת הרוב מבין השכנים הקרובים ביותר 'k' נחשבת כתחזית עבור נקודת הנתונים הנתונה.

היתרונות של KNN

- קל להבנה ויישום.
- אינו מניח הנחות כלשהן לגבי התפלגות הנתונים הבסיסית, מה שהופך אותו למתאים לסוגים שונים של נתונים.

החסרונות של KNN

- יקר מבחינה חישובית. מכיוון ש-KNN צריך לחשב מרחקים עבור כל נקודת נתונים במהלך חיזוי, במיוחד עבור מערכי נתונים גדולים.
- מסתמך על מדדי מרחק, מה שהופך אותו לרגיש לתכונות לא רלוונטיות או רועשות, שעלולות להשפיע לרעה על התחזיות.

לסיכום, K-Nearest Neighbors הוא אלגוריתם פשוט ויעיל למשימות סיווג ורגרסיה שונות. הפשטות והאופי הלא פרמטרי שלו הופכים אותו לבחירה פופולרית עבור תרחישים שבהם פרשנות וקלות היישום חשובים. עם זאת, חיוני לשקול את המגבלות שלו ולבחור את הערך המתאים של 'k' ומדד מרחק כדי להשיג את הביצועים הטובים ביותר עבור בעיה ספציפית.

***בפרוייקט זה נעשה שימוש באלגוריתם KNN המיובא מספריית sklearn.

דו"ח מסכם:

תוצאות ההערכה הראו שאלגוריתם AdaBoost המותאם אישית שלנו השיג דיוק בדיקה גבוהה יותר בהשוואה לאלגוריתם KNN . ביצענו השוואות שונות באמצעות מדדים שונים כמו:

- confusion matrix - מטריצה, כלי להערכת ביצועים כדי להשוות ערכים חזויים לעומת ערכים אמיתיים.
- accuracy - רמת הדיוק של האלגוריתם. (מספר הבדיקות) /(מספר הפעמים שהאלגוריתם צדק).
- recall - מה הם אחוזי ההצלחה בעבור קבוצת החולים. (TP)/(FN + TP)
- precision - מקרים בהם האלגוריתם מכריע חולה, כמה פעמים הוא צדק. (TP) /(TP+ FP)

- F1-score - ממוצע הרמוני, מדד שמשלב את ה- recall ואת ה- precision.

השפעה של פרמטרים:

באלגוריתם ה AdaBoost ביצענו בדיקות על הפרמטרים. הורדנו פרמטרים, החסרנו עמודות ומחקנו נתונים על מנת לבדוק את רמת ההשפעה על התוצאה הסופית.

הערכות:

- הערכנו שהורדת עמודות מהטבלה (תכונות) עלולה לפגוע ביעילותו ודיוקו של האלגוריתם על התוצאות. ולכן, בחרנו להוריד עמודות שונות. סברנו כי הורדת עמודות שונות ישפיעו במידה רבה על רמת הדיוק של האלגוריתם מהסיבה שהורדת תכונות, משמעותן פחות נתונים וכך האלגוריתם יהיה פחות מדוייק.
- הערכנו ששינוי כמות הstump-ים תשפיע על רמת הדיוק של האלגוריתם, כל stump הוא עץ החלטה שמניע את האלגוריתם. הערכנו שיותר stump-ים כאלה יתנו דיוק גבוהה יותר תוך פגיעה בזמן הריצה.

תוצאות מ- AdaBoost:

תוצאות ביחס לדיוק האלגוריתם:

- תוצאת רמת הדיוק של האלגוריתם שלנו:

Test Accuracy of Adaboost Algorithm: 90.73%

- תוצאת רמת הדיוק של האלגוריתם לאחר הורדת 2 עמודות מהדאטה שקיבלנו: רמת כולסטרול בדם (chol) ו-כמות סוכר בדם (fbs) :

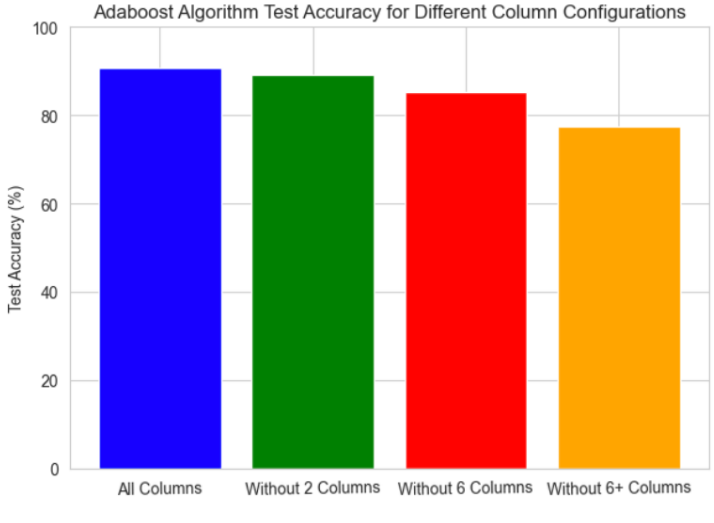
Test Accuracy of Adaboost Algorithm without fbs and chol parameters: 89.27%

- תוצאת רמת הדיוק של האלגוריתם לאחר הורדת 6 עמודות מהדאטה שקיבלנו:נוכחותם של כלי דם צבעוניים (ca), רמת כולסטרול בדם (chol), כמות סוכר בדם (fbs) , סוג הכאב בחזה (cp), דופק המרבי שהושג במהלך מבחן מאמץ (thalach) ו-לחץ דם במנוחה (לחץ הדם במנוחה) :

Test Accuracy of Adaboost Algorithm without 6 columns: 85.37%

- תוצאת רמת הדיוק של האלגוריתם עם מספר מצומצם של תכונות מהדאטה שקיבלנו, אלו הן עמודות שכן השארנו: גיל (age), מין (thai), (sex):

Test Accuracy of Adaboost Algorithm without 6 plus columns: 77.56%

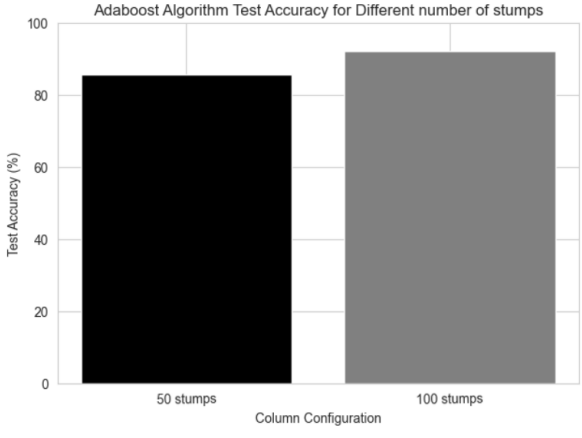


תוצאות ביחס לכמות ה- stump-ים:

- ולכן הרצנו את האלגוריתם על 50 stump-ים ועל 100 stump-ים.

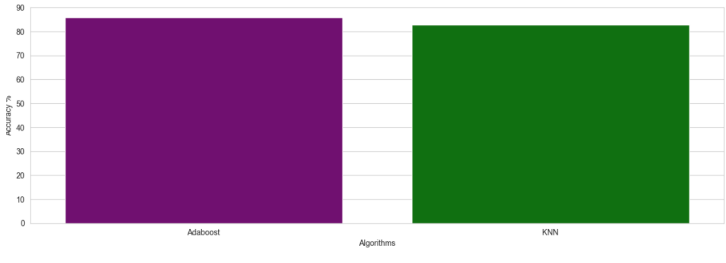
Test Accuracy of Adaboost Algorithm with 50 stumps: 85.85%

Test Accuracy of Adaboost Algorithm with 100 stumps: 92.20%

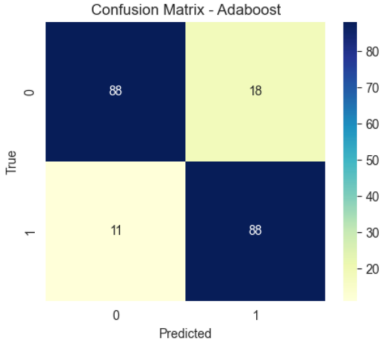


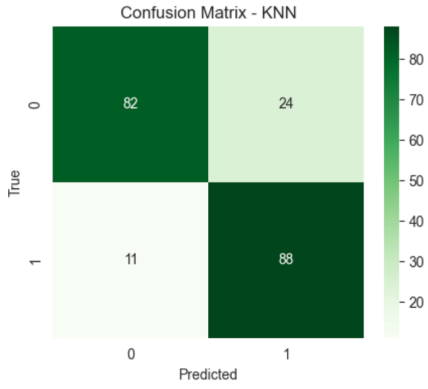
תוצאות ביחס לאלגוריתם אחר:

- השוואה לKNN: כדי לקבל תובנות נוספות, יישמנו והערכנו גם מודל KNN (K-Nearest Neighbors) לחיזוי מחלות לב. ביצענו בדיקה באמצעות המדדים השונים שהזכרנו לעיל. בנינו טבלה שמשווה בין האלגוריתמים שלנו. מדד ה- F1-score הראה תוצאות עדיפות באופן גורף בהשוואה לאלגוריתם ה- KNN.



Algorithm	Accuracy	FPR	Recall	F1 Score
Adaboost	85.85%	16.98%	88.89%	85.85%
KNN	82.93%	22.64%	88.89%	83.41%





מסקנות מ- AdaBoost:

צפינו כי הורדת עמודות עם תכונות משמעותיות תגרום להורדת רמת הדיוק של האלגוריתם. בפועל, מה שקיבלנו הראה על תוצאות שאינן חד משמעיות בחלק מהמקרים. כלומר, כאשר הורדנו שתי עמודות מהדאטה ראינו בהרצות שונות שרמת הדיוק של האלגוריתם גבוהה יותר בהשוואה לדאטה המקורית.

בחלק מההרצות נוכחנו לדעת שהורדת חלק מהתכונות לא השפיעה והסיבה לכך היא שעמודות אלו לא תורמות להחלטה האם בסוף המטופל חולה או לא חולה. יתכן ושתי העמודות שנשמטו אינן בעלות כוח חיזוי או מכילות מידע רועש או לא רלוונטי ולכן ההתעלנות מהן לא השפיעה באופן משמעותי על ביצועי המודל.

כאשר המשכנו בהורדת תכונות נוספות, 6 עמודות או יותר, האלגוריתם לא דייק בתוצאות וכפי שצפינו בהערכות רמת הדיוק ירדה. באופן כללי, ראינו שככל שאנחנו מחסירים יותר עמודות מהדאטה, רמת הדיוק הלכה כי יש פחות מידע.

בתייחס לכמות ה- stamp-ם, ראינו תוצאות שהן לא חד משמעיות. ברוב הבדיקות שעשינו, כאשר כמות ה- stump-ים הייתה גדולה יותר (100) התוצאות היו מדוייקות יותר, רמת הדיוק הייתה גבוהה יותר, אך יחד עם זאת במספר הרצות בודדות קיבלנו תוצאה הפוכה . ההנחה שלנו היא שנתקלנו בסיטואצייה של over-fittingוכתוצאה מכך המודל שלנו הפך למודל פחות יעיל עבור נתונים חדשים שלא נראו מה שהבוביל לירידה ברמת הדיוק.במקרה שלנו, בעבור 100 stump-ים יתכן ומודל ה- adaboost היה מורכב מדי ונוצר over-fit על קבוצת הלימוד. ולכן תפקד בצורה לא טובה במדד הדיוק עבור מספר קטן של הרצות לעומת ה- 50 stump-ים. כמובן שהיה כאן trade-of, בהוספת כמות ה- stump-ים זמן הריצה גדל ורמת הדיוק עלתה בהתאמה ברוב הבדיקות שעשינו.

בהתייחס להשוואה שנעשתה לאלגוריתם ה- KNN, אנו מאמינים שהסיבה לתוצאות שראינו נבעה מהחסרונות של אלגוריתם ה- KNN. כאמור, KNN מסתמך על מדדי מרחק, מה שהופך אותו לרגיש לתכונות לא רלוונטיות או רועשות, שעלולות להשפיע לרעה על התחזיות השונות. בכל אופן, AdaBoost הראה הראה ביצועים טובים יותר במדדים שביצענו. בהרצאה ראינו שמדד ה- accuracy יכול לצאת מאוד גבוהה ומדוייק, אבל לא בהכרח שהמדד הזה הוא מספיק טוב, ולכן ביצענו השוואה בין האלגוריתמים בעזרת עוד מדדים כמו: confusion matix, recall presicion ו- F1-score שבסופו של דבר הראו שאלגוריתם ה- AdaBoost היה עדיף בהתאם למקרה שלנו.

עוד בדיקה שעשינו הייתה להכניס נתונים של אדם בריא ולבדוק את תוצאות ההרצה של האלגוריתם על הנתונים שהוזנו. אלגוריתם ה- AdaBoost הכריע שהאדם היה בריא בעוד שאלגוריתם ה- KNN הכריע שהאדם חולה. כל זאת חיזק את ההשערות שלנו.