מעבדה בעיבוד אותות פזיולוגיים
הנדסה ביורפואית

מגישים:

דן טורצקי
סול אמארה

תאריך:

10.1.2023

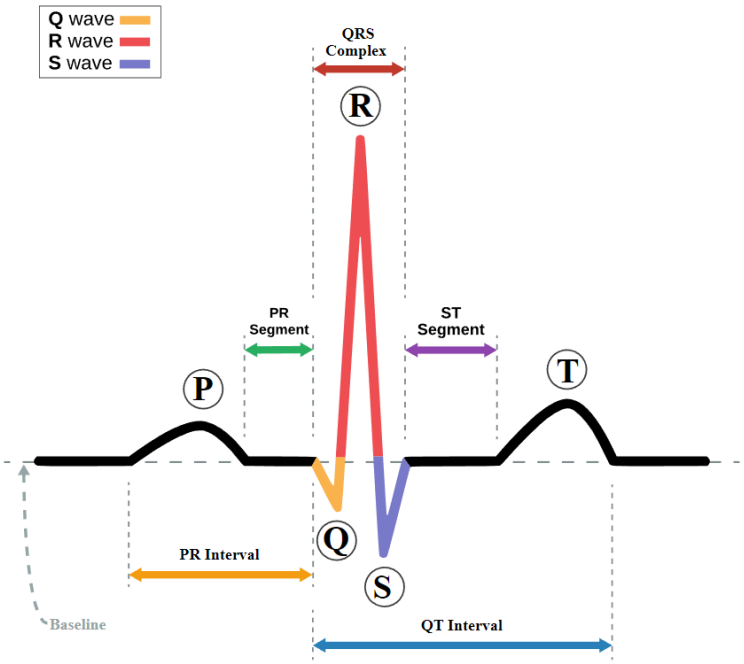
תוכן עניינים:

רקע תאורטי

1.1 חלק 1- אות הECG

תשובה לשאלה 1:

1. כיווץ והרפיית הלב מתרחשים בעקבות פעילות חשמלית של תאי עצב באמצעות סינפסות כימיות וחשמליות. ECG הינה הקלטה של פוטנציאלים חשמליים אלה במשך הזמן.
מניתוח של אות זה ניתן ללמוד על הפעילות החשמלית של הלב ולזהות דפוסים הנקשרים לפתולוגיות שונות.



איור 1: דופק סינוס תקין עם סימוני של הגלים והסגמנטים השונים [1]

באיור המוצג ישנו גרף תיאורטי של אק"ג תקין (פעימה בודדת). נתאר את המשמעות הפיזיולוגית של הגלים והמרווחים המסומנים באיור [2]

1. P wave: דה פולריזציה של העליות.
2. PR interval: מרווח הזמן בו מתפשט האות החשמלי ב AV node, Bundle of His וסיבי פורקיניה. במשך הזמן האק"ג בקו האיזו-אלקטרי (ה – DC של האות) התקין של סגמנט זה הינו 120~200ms, וחריגה ממשך זמן זה יכולה להעיד על פתולוגיות שונות (למשל כאלה הקשורות בהולכה ב AV node).
3. QRS complex: דה פולריזציה של החדרים.
4. ST segment: משך הזמן בין סוף הדה פולריזציה של החדרים עד תחילת הרה פולריזציה. סטייה של סגמנט זה מה – DC של האות יכולה להעיד על פתולוגיות שונות.
5. T wave: רה פולריזציה של החדרים.

תשובה לשאלה 2:

ישנם מס' סוגי רעשים היכולים להופיע באות ECG, ביניהם:

[3] [4] [5]

2.1. Base-line drift : רעש זה נובע מתנועה של הנבדק (כמו נשימה) או תזוזה של האלקטרודה על גבי העור, הגורמת להיסט של קו הבסיס (ה – DC) של האות. רעש זה מאופיין לרוב בתדרים נמוכים והאות הזמני יראה כאות הרוכב על סינוס (או קירוב כלשהו) בתדר נמוך. פונקציית האוטו-קורלציה במקרים רבים תאופיין ע״י דעיכה אקספוננצאלית איטית.

2.2. EMG : רעש זה נובע מפעילות חשמלית של השרירים. במישור הזמן האות יראה כתנודות אקראיות עם אמפ׳ גבוהה. כיוון שהשינויים הזמניים מהירים מרבית האנרגיה בתדר תהיה בתדרים גבוהים. פונקציית האוטו-קורלציה תאופיין לרוב ע״י דעיכה אקספוננציאלית מהירה.

2.3. רעש רשת : רעש זה נובע מקווי מתח המשרים שדה מגנטי שבתורו יוצר זרם דרך המטופל או במכשיר המדידה עצמו. רעש זה הינו בקירוב סינוסי טהור עם תדירות של $50\backslash 60Hz$ כתלות ברשת החשמל הפרוסה. בזמן רעש זה הוא סינוס באחת התדירויות הנ״ל, ובתדר יופיע כדלתא ב - $\pm 50\backslash 60\ Hz$. פונקציית האוטו-קורלציה של אות מחזורי הינה מחזורית גם כן.

2.4. Motion artifacts : רעש זה נובע מתנועה של האלקטרודות או של המטופל. מתיחה\הרפיה של העור גורמים לשינוי בהתנגדות הכניסה של האלקטרודה. רעש זה נראה בזמן כתנודות מהירות ובאמפ׳ גבוהה היכולות להיות דומות ל - QRS. מרבית האנרגיה בתדר תופיע בתדרים -1 10Hz דבר הגורם לחפיפה עם תדרי ה – QRS ויכול להקשות על סינונו. האוטו-קורלציה תדעך בקצב מהיר.

1.2 חלק 2- למידת מכונה ולמידה עמוקה

תשובה לשאלה 1:

[6]

Supervised learning – למידה מבוקרת, הינה סוג של למידת מכונה. בלמידה מסוג זה מאמנים את המודל לבצע תחזיות בהתבסס על בסיס נתונים מתויג – כלומר אנו נותנים למודל אינפוט, המודל אמור לפלוט תחזית בהינתן כניסה זו, כאשר אנו יודעים מה המוצא האמיתי שאמור להתקבל עבור דגימה זו. על כל שגיאה של המודל ניתן "עונש" שאמור לעודד את המודל לייצר תוצאות הדומות יותר לתיוג האמת שברשותנו. דוגמה לשיטה זו הינה משימת קלסיפיקציה – אנו רוצים ליצור מודל שיודע לקבל תמונה ולהחליט האם מופיעים בתמונה כלבים או חתולים.
האינפוט של המודל יהיה תמונות של כלבים או חתולים (לא שניהם באותה תמונה), והמוצא יהיה החלטה לאיזה קלאס התמונה שייכת – כלומר האם מופיע בתמונה חתול או כלב. נשתמש בתמונות מתויגות בהן ידוע מה מופיע בתמונה לשם אימון המודל. דוגמה נוספת למודל מסוג זה הינה מודל רגרסיה. במודל מסוג זה אנו רוצים לייצר פונקציה רציפה שתתאים לכל כניסה (מהתחום הרציף) מוצא (גם הוא בתחום הרציף). למשל, נרצה ליצור מודל בו בהינתן מידע כגון סוג החיה, ממדיה וגילה נשערך את משקלה. לשם כך נצטרך בסיס מידע המכיל סוג חיה, ממדיה, גיל ומשקל.

תשובה לשאלה 2:

Gradient descent – הינו אלגוריתם אופטימיזציה איטרטיבי, הנועד לפתור בעיית מינימום/מקסימום גלובלית (לרוב השימוש הוא לבעיית מינימום). באלגוריתם זה, בהינתן נקודת התחלה, בכל איטרציה ננוע בכיוון הגרדיאנט (הכיוון בו הפונקציה משתנה הכי הרבה) או בכיוון ההופכי, בהתאם להיות הבעיה מוגדרת כבעיית מקסימום או מינימום. גודל הצעד שאנו עושים הינו היפר-פרמטר חשוב של המודל. צעד קטן ידאג להתקדמות יציבה, אך ידרוש יותר חישובים כדי למצוא את המינימום. חולשה נוספת של צעד קטן הינה הסכנה להיתקע במינימום מקומי מבלי היכולת לצאת ממנו. צעד גדול מדי יכול לגרום למערכת להיות לא יציבה ולהתבדר או במקרה פחות קיצוני לדלג מעל המינימום כך שלא תהיה התכנסות. הייתרון של צעד גדול מספיק הינו זמן חישוב קטן יותר והפחתת הסיכון של היתקעות במינימום מקומי. שימוש באלגוריתם זה נפוץ מאוד בלמידת מכונה ולמידה עמוקה בלמידה מבוקרת. בלמידה מבוקרת אנו מאמנים את המודל ע"י תיקון השגיאות במוצא ביחס לתיוג הידוע שאנו רוצים שהמודל ייצא. באופן אידאלי היינו רוצים כי השגיאה בין מוצא המערכת למידע המתויג האמיתי תהיה אפס, כחומר אנו רוצים להביא את פונקציית השגיאה למינימום. פונקציית השגיאה תהיה מוגדרת כפונקציה שהאינפוט שלה הוא אינפוט המערכת וכן פרמטרים של המערכת, ואנו רוצים למצוא את הפרמטרים של המערכת שיובילו לשגיאה מינימלית בין מוצא המערכת למוצא האמת המתויג. אם כך ניתן לתאר בעיה זו כבעיית מינימיזציה של פונקציית השגיאה שהינה פונקציה של פרמטרי המערכת (משקולות). אנו מעדכנים את המשקולות כדי להתקרב למינימום.

תשובה לשאלה 3:

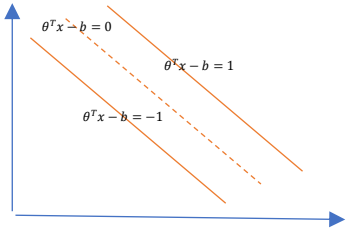
פונקציית loss – פונקציה זו משמשת בלמידה מבוקרת כמדד לשגיאה בין מוצא המערכת לערך האמת המתויג. ככל שהשגיאה יותר גדולה, כך ה"עונש" יותר גדול, כלומר מודל הלמידה מבצע שינויים יותר גדולים במודל כדי לפצות על השגיאה הגדולה על מנת להקטין את שגיאות המודל על דאטה הלמידה. ישנן סוגים רבים של פונקציות מחיר, אשר בכולן יש ביטוי המתייחס לשגיאה בין המוצא לתיוג האמת, כאשר המושג שגיאה יכול להיות מוגדר בדרכים שונות וכן הביטוי המתמטי המתאר את שגיאה זו יכול להיות מגוון רחב של פונקציות. במקרים מסוימים ישנו ביטוי נוסף בפונקציית המחיר שנועד לשרת מטרה נוספת שאנו רוצים לממש במודל, כמו למשל ביטויים הקשורים לאיזון בין הנתונים בבסיס המידע או ביטוי הקשור באיזון בין המשקולות השונות (הפרמטרים) של המודל.

תשובה לשאלה 4:

שלבי הלמידה של מסווג linear SVM:

בסוג מסווג זה נרצה לבצע הפרדה בין קבוצות עם קו לינארי עבור למידה מונחית. השלב הראשון הינו חלוקת המידע לדאטה למידה (בו נשתמש כדי לאמן את האלגוריתם ולמצוא את ההפרדה) ודאטה מבחן (עליו נבחן את ההפרדה שנקבל). לאחר מכן, נגדיר קו שיתאר את ההפרדה, $\theta^T x - b = 0$ כאשר נמצא את הקבועים, θ,באמצעות האלגוריתם. לקו זה נגדיר שוליים (margin), אותם נגדיר להיות אורתוגונליים לקו ההפרדה ובמרחק של $\frac{1}{||\theta||}$, שישמשו אותנו בתהליך הלמידה כאזור הפרדה מקסימלי. לעיתים נרצה שוליים רכים שיהיו מוכנים לקבל טעויות ולעיתים שוליים קשים. הא הינם היפר-פרמטרים אותם יש לבחור בהתאם לדאטה שלנו.

נדגים את ההפרדה עבור שני ממדים:



איור 2: הפרדה באמצעות linear SVM

כאשר קבוצה אחת נקבל עבור $\theta^T x - b > 1$ וקבוצה שניה עבור $\theta^T x - b < -1$.

בשל כך, נרצה למצוא את המינימום של $||\theta||$ כלומר שהביטוי $y_i(\theta^T x_i - b)$ יהיה גדול או שווה ל1. כלומר, את הקבועים θ נמצא על ידי מיקסום של הביטוי: $\max\left(0,1 - y_i(\theta^T x_i - b)\right)$.

לסיכום השלבים:

1. הפרדה לדאטה למידה ודאטה מבחן
2. בחירת היפר-פרמטרים
3. מציאת הθ שיביאו את הביטוי למקסימום
4. בדיקת המודל על דאטה המבחן
5. במידה ותוצאות המודל לא טובות ניתן לשנות את ההיפר-פרמטרים (או לעבור לקרנלים אחרים, אלגוריתמי SVM שאינם ליניאריים)

[7]

<u>תשובה לשאלה 5:</u>

שלבי הלמידה של רשת נוירונים:

גם כאן נתחיל מחלוקה של הדאטה לדאטה למידה ודאטה מבחן כאשר את שלבי האימון של המודל נבצע על דאטה הלמידה ולאחר מכן נבחן אותו על דאטה המבחן. לאחר מכן, נבחר את כמות השכבות וכמות הנוירונים בכל שכבה. נבצע אתחול של המשקלים ונתחיל באימון המודל- נכניס את הדאטה למידה ונחשב את המוצא עבורו. נחשב את ההפסד- השוני בין המוצא שקיבלנו לבין המוצא המתויג. לאחר מכן, נבצע עדכון של המשקלים בכיוון ההפוך לגרדינט של ההפסד. נחזור על התהליך עד שנגיע להתכנסות עבור המשקלים ונבצע בדיקה על דאטה המבחן.

לסיכום השלבים:

1. הפרדה לדאטה למידה ודאטה מבחן

2. בחירת היפר-פרמטרים, כמות השכבות וכמות הנוירונים בכל שכבה
3. אתחול המשקלים
4. חישוב ההפסד
5. חישוב גרדינט (שיפוע) ההפסד ביחס למשקלים
6. עדכון המשקלים בכיוון ההפוך לגרדינט
7. חזרה איטרטיבית על 4-6 עד הגעה להתכנסות של המשקלים
8. בדיקת המודל על דאטה המבחן
9. במידה ותוצאות המודל לא טובות ניתן לשנות את ההיפר-פרמטרים או את כמות השכבות

[8]

<u>תשובה לשאלה 6:</u>

המטרה של דאטה המבחן הינה אימון המודל ומציאת הנעלמים (המשקלים למשל), ואילו מטרת דאטה המבחן הינה למדוד את טיב המודל. במידה ונבצע את הבדיקה של טיב המודל על אותו דאטה עליו אימנו אותו, נקבל overfitting כלומר מודל שנותן תוצאות מאוד טובות לדאטה המסוים עליו אימנו אך אינו יכליל מקרים אחרים ולא ייתן תוצאות טובות למידע אחר. מטרת המודל הינה לתת הפרדה כללית בין הקבוצות, ונרצה שאם נקבל מידע חדש המודל ייתן תוצאה טובה גם עליו ולכן נרצה להימנע מoverfitting.

<u>תשובה לשאלה 7:</u>

דאטה ולידציה משמש להערכת ביצועי האלגוריתם. לאחר שימוש בדאטה הלמידה בעזרתו נמצא את הפרמטרים, נבדוק את ביצועי האלגוריתם על דאטה הוולידציה. במידה ונרצה לשפר את ביצועי האלגוריתם ניתן לעשות זאת ולחזור שוב על הבדיקה על דאטה זה. כלומר, כל עוד נרצה לשפר ולעדכן את האלגוריתם נשתמש בלמידה ובוולידציה וכאשר נסיים ונרצה לבדוק את ביצועיו נשתמש בדאטה המבחן. ההבדל בין הוולידציה למבחן הוא שלאחר שהשתמשנו במבחן הוא כבר יחשב כוולידציה. כלומר, דאטה מבחן הוא דאטה שלא שומש עדיין לעומת דאטה הוולידציה איתו ניתן לבדוק שוב ולעדכן.

<u>תשובה לשאלה 8:</u>

Overfitting משמעותו התאמת יתר של האלגוריתם לנתונים. מושג זה מתייחס למודל שמותאם מידי לנתונים שנתנו לו כך שאין שגיאות בזיהויו (או שהן מינימליות), אך כאשר נרצה לבחון אותו על מידע חדש הוא לא יעשה התאמה טובה כלומר הוא לא מתאים כמודל מכליל. מצב זה קורה כאשר משתמשים ביותר מידי פיצ׳רים ביחס לכמות המידע או שמשתמשים במודל מסובך מידי.
בנוסף, overfitting קורה במצבים בהם לא עושים הפרדה לדאטה למידה וולידציה כך שהאלגוריתם מאומן על נתונים מסוימים ואמינותו לא נבדקת ומתעדכנת על נתונים אחרים.

[9]

<u>תשובה לשאלה 9:</u>

במצבים של חוסר איזון בין הנתונים כלומר כאשר יש יותר דגימות של קבוצה אחת מאחרת, המודל יכול להיות מדויק יותר לקבוצה הגדולה כך שזיהויו אליה יהיו נכונים (בשל שונות גדולה יותר בין הנתונים עליו התאמן) אך לקבוצה השנייה לה יש פחות נתונים המודל יהיה פחות מדויק. בעיה זו עלולה לגרום לרגישות יתר עבור הקבוצה השנייה (כניסיון פיצוי), מה שיכול להוביל לירידה בדיוק המודל.

[10]

ניסוי 1 : רעשים באות הECG

1.1

סינון רעש רשת: רעש רשת הינו בתדר [Hz] 50 ולכן על מנת לסנן אותו נרצה לייצר מסנן שמוריד תדר זה בלבד. נסמן את תדר הדגימה בF_s ואז תדר הקטעון יהיה $\frac{50}{F_s}$' ניתן לבצע את הסינון באמצעות FIR בו נשים אפס בתדר זה ונרצה סדר מסנן גבוה לקבלת תחום מעבר קטן ככל האפשר. אמנם, מסנן זה יהיה בעל רוחב סרט גדול ולכן ניתן לשפר אותו באמצעות מסנן מסוג *IIR* בעל קטבים בתדר זה ברדיוס *r*. ככל שנקח רדיוס גדול יותר נוכל לקבל תגובה חדה יותר (אך יותר זמן חישוב) וזה המסנן שנבחר. תמסורת המסנן :

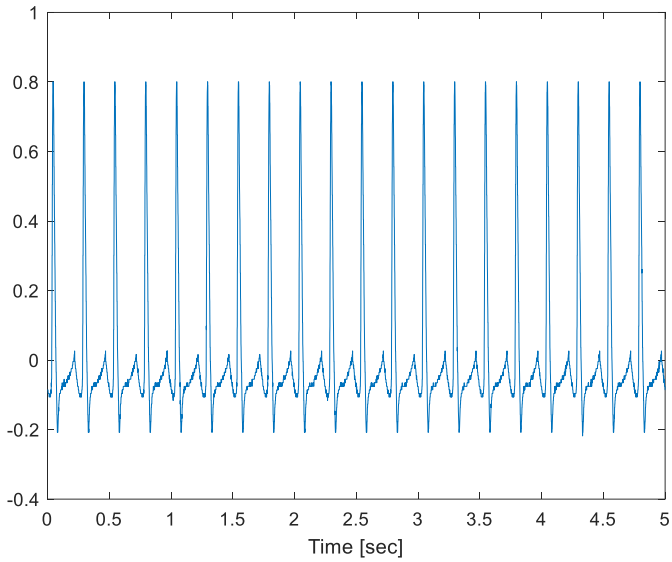
$$H(z) = \frac{(1 - z_1 z^{-1})(1 - z_2 z^{-1})}{(1 - r z_1 z^{-1})(1 - r z_2 z^{-1})} = \frac{1 - 2\cos(\omega_0)\, z^{-1} + z^{-2}}{1 - 2r\cos(\omega_0)\, z^{-1} + r^2 z^{-2}}\ , \omega_0 = 2\pi \cdot \frac{50}{F_s}$$

סינון *Base line*: אות ה*ECG* מתחיל מתדר של *0.67 [Hz]* ולכן ניתן לסנן את הבייס ליין באמצעות *HPF* בתדר של *0.5[Hz]* , כלומר תדר קטעון $\frac{0.5}{F_s}$. נשתמש במסנן *FIR* בעל סדר גבוה כדי לקבל סינון טוב (מעל 1150).

סינון EMG: רעש שרירים נמצא בתדר של 20-500 הרץ . במידה ויש לנו אותות רפרנס של רעש זה ניתן לסנן אותו באמצעות אלגוריתם הLMS שזהו אלגוריתם מתעדכן המחשב משקלים של אותות הרפרנס לסינון הרעש בצורה האידיאלית. במידה ולא נתונים לנו אותות הרפרנס ניתן לעשות LPF בתדר קטעון של $\frac{20}{F_s}$ כך שיסנן את תדרי רעש הEMG אך כיוון שאות הECG מכיל גם מידע בתדרים אלה נקבל פגיעה בסיגנל ולכן שיטת הLMS עדיפה עבור רעש זה. ממעבדות קודמות נסיק כי ניתן להשתמש במסנן בעל 500 מקדמים כיוון שהוא יתן תגובה לתדר חדה יחסית.

סינון motion artifacts: תזוזת אלקטרודות בתדרים של 1-10 הרץ. כיוון שרעש זה חופף רבות עם תדרים של QRS, לא ניתן להסרו עם מסנן תדרי של תדרים מסוימים. שיטה לסנן אות זה היא שימוש בסנסור תאוצה, למדידת תנועת הגוף ובכך לסנן, לפחות חלקית אות זה באמצעות שיטות כגון LMS. שימוש ב – HPF בתחום בתדר קטעון של $10Hz$ הינו אפשרי, אך ככל הנראה יפגע משמעותית באות ה – ECG.

1.2



איור 3: מקטע מתוך אות הECG

מאיור זה ניתן לראות מקטע באורך של 5 שניות מתוך האות שהקלטנו במעבדה 1. ניתן לראות כי אות זה מכיל את קומפלקס הQRS וגל P.

1.3

EMG הינו רעש אקראי בתדרים גבוהים ולכן נשתמש בהוספת רעש גאוסי לבן (פונקציית awgn במטלאב).

על מנת לקבל $SNR = 30[dB]$: $30 = 10 \cdot \log\left(\frac{signal\ power}{noise\ power}\right)$

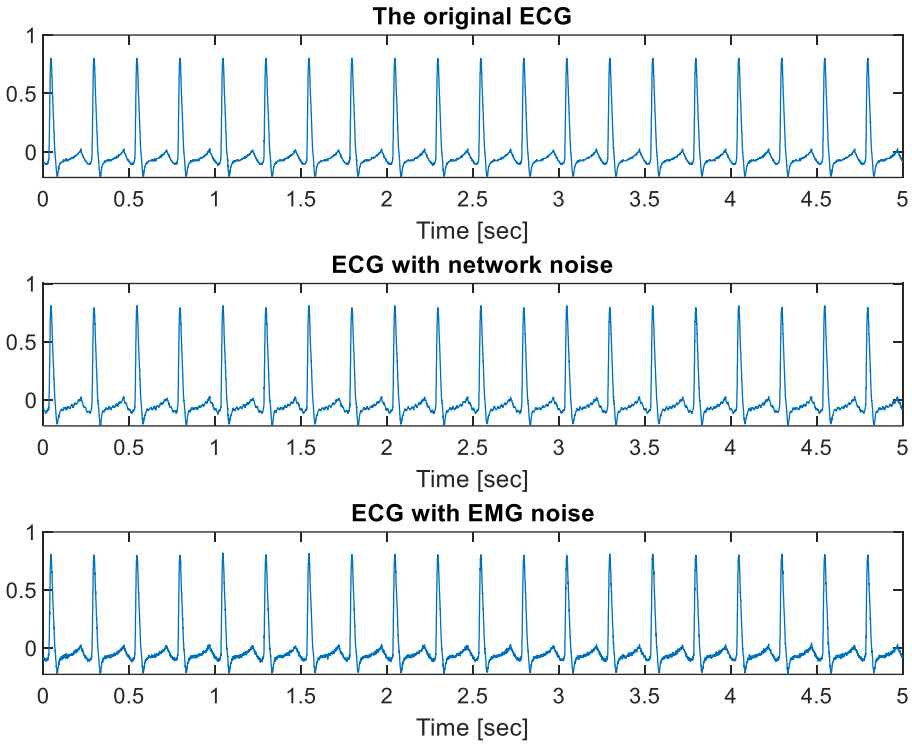
$$\frac{signal\ power}{noise\ power} = 10^3 \rightarrow noise_{power} = 10^{-3} signal_{power}$$

כאשר את העוצמה נחשב: $\sum_{-\infty}^{\infty} x_i^2$.

בשל כך, בהנחה ונסמן את הרעש: $noise = a \cdot \mathrm{f}$ כאשר a הוא ההגבר, נוכל להגיד:

$$noise_{power} = a^2 \cdot f_{power} \rightarrow a = \sqrt{\frac{noise_{power}}{f_{power}}} = \sqrt{\frac{10^{-3} signal_{power}}{f_{power}}}$$

כמו כן לאחר חישוב a והרעש ביצענו בדיקה באמצעות המטלב וקיבלנו שהSNR=30 כרצוי.

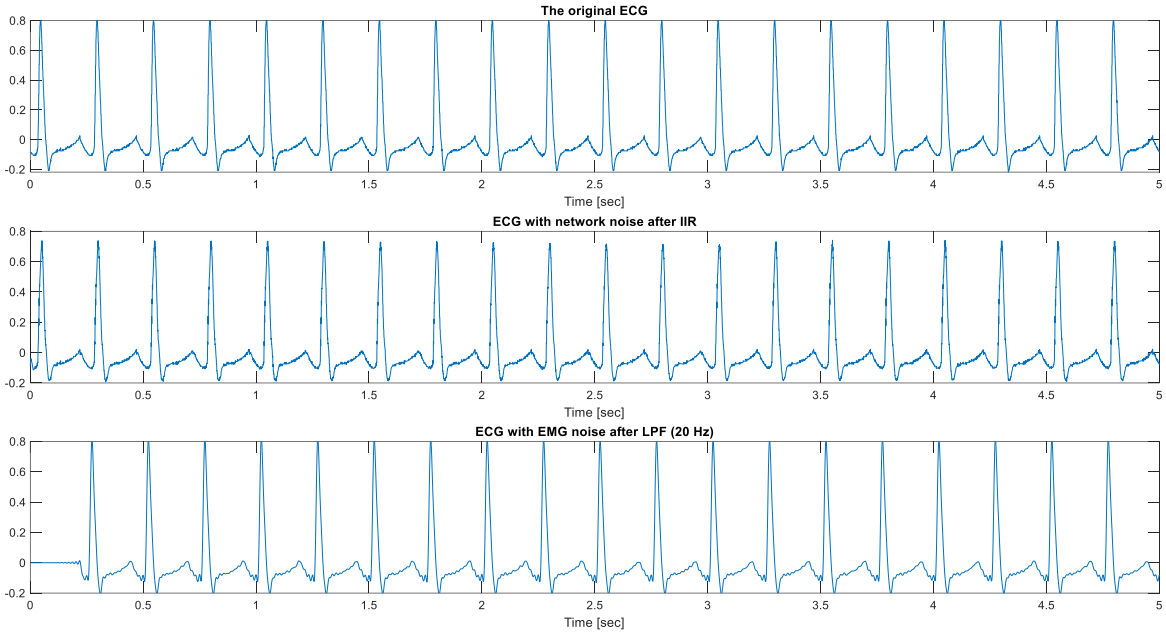


איור 4: אות ה ECG המקורי והאות בתוספת הרעשים

מאיור זה ניתן לראות את האות המקורי (העליון) ולאחר מכן את האות בתוספת רעש הרשת (אות סינוסי בתדר של 50 הרץ) ובתוספת רעש השרירים (אות אקראי בהתפלגות נורמלית בעל ממוצע 0 ושונות 1 עליו הפעלנו BPF בתדרים 20-500 הרץ שהם תדרים המאפיינים רעש מסוג זה).

1.4

לאחר הפעלת המסננים כפי שהסברנו בסעיף 1.1:

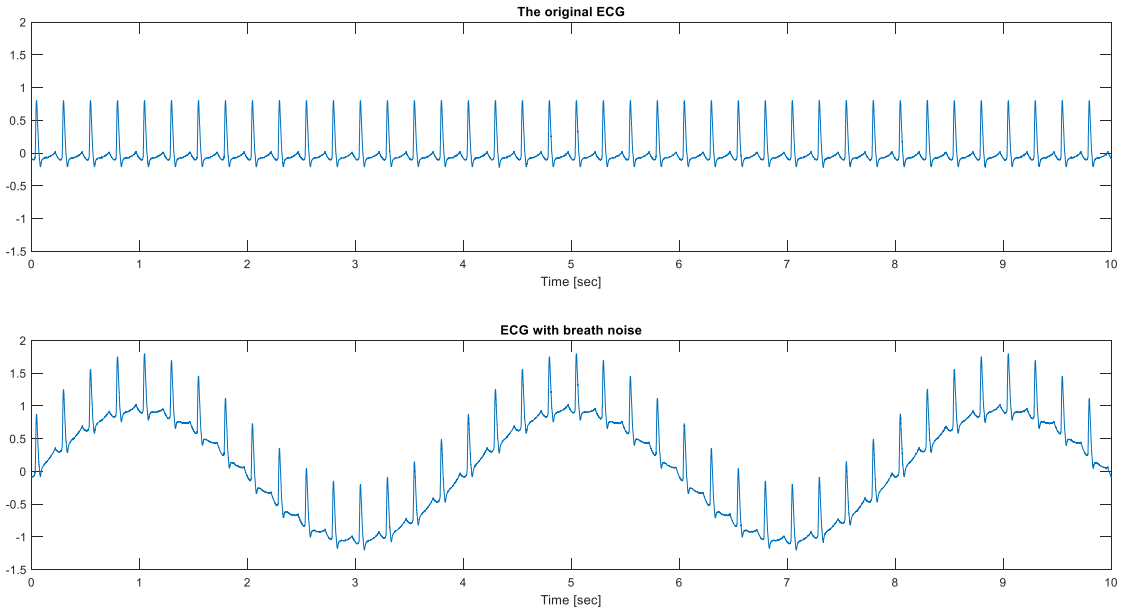


איור 5: האותות המסוננים

מאיור זה ניתן לראות כי קיבלנו תוצאת סינון טובה עבור סינון רעש הרשת באמצעות מסנן הIIR כאשר יש אפסים בתדר 50 הרץ וקטבים בתדר זה ברדיוס של 0.9. בהשוואה לאות המקורי בעין אנושית לא ניתן לראות הבדל (מלבד לתחילת הסגנל) בין האותות ולכן נסיק כי מסנן זה נתן את תוצאת הסינון הדרושה. עם זאת, עבור מסנן הEMG בו השתשמנו במסנן LPF מסוג FIR בתדר של 20 הרץ ובעל 500 מקדמים, קיבלנו בהחלט סינון של הרעש אך תוצאת הסינון איננה זהה לאות המקורי כלומר לא הצלחנו לבצע את הסינון המקסימלי. כפי שציינו, ניתן לקבל תוצאה טובה יותר באמצעות מדידות רפרנס ושימוש באלגוריתמים מורכבים יותר כמו LMS או שימוש במסנן בעל תגובה חדה יותר והעלאת כמות המקדמים.

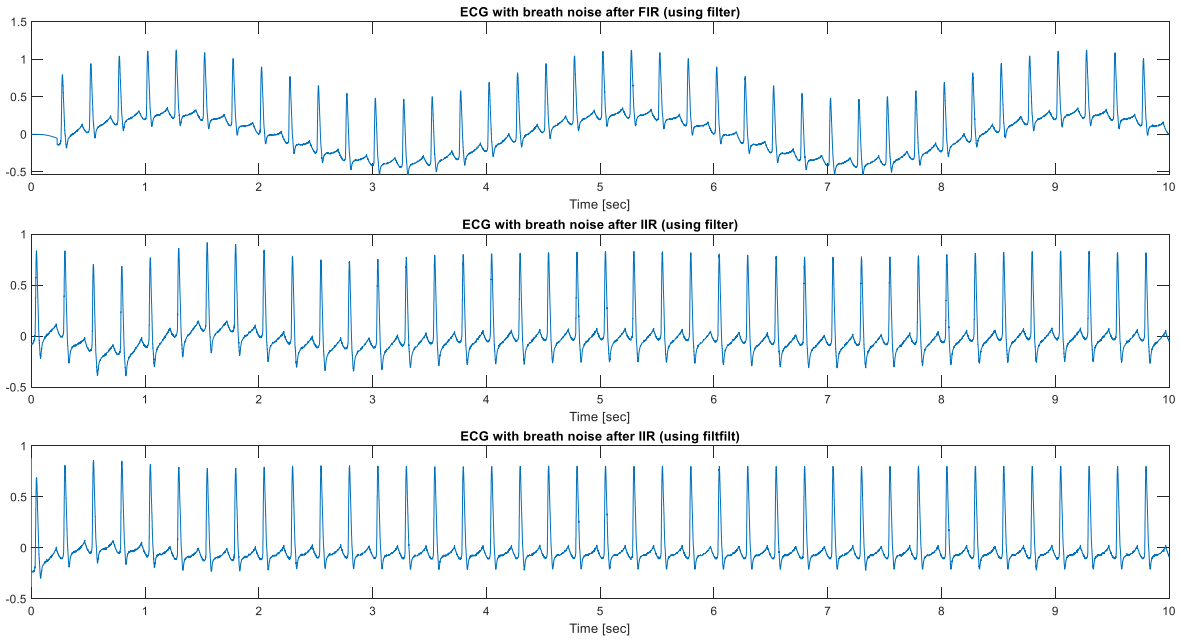
1.5

אות נשימה הוא בקירוב סינוסי ועבור אדם רגוע קצב הנשימה בקרוב 15 נשימות לדקה כלומר $\frac{15}{60} = 0.25\ [Hz]$.



איור 6: האות המקורי והאות בתוספת רעש נשימה

מאיור זה ניתן לראות את השפעת רעש הנשימה על אות הECG הנמדד. נציג כעת את התוצאות לאחר הסינונים השונים והורדת DC:



איור 7: האותות לאחר סינון רעש הנשימה

מאיור זה ניתן לראות את תוצאות הסינון הן באמצעות מסנן FIR והן סאמצעות מסנן IIR. ראשית באמצעות שימוש בפקודת filter ניתן לראות כי תוצאות הסינון של מסנן הIIR הניבו תוצאות טובות יותר של סינון רעשי הנשימה בהשוואה למסנני FIR. נסיק כי לסוג מסנן זה תגובה לתדר חדה יותר בשל שימוש בקטבים, מה שמוביל את המסנן להיות בעל תגובה להלם אינסופית. כמו

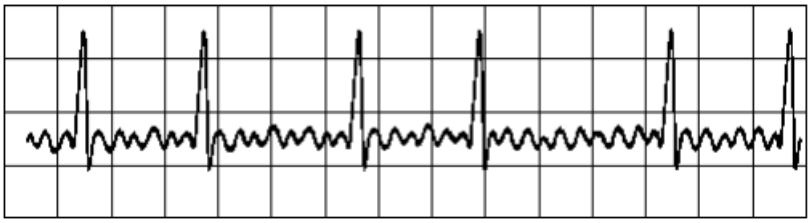
כן, בעת שימוש בפונקציית filtfilt עבור מסנן הIIR קיבלנו את תוצאת הסינון הטובה ביותר שכן הסינון בתחילת הסיגנל באמצעות פקודה זו יותר טוב בשל תכונות הפונקציה.

פקודת filtfilt מטרתה לבצע סינון פעם קדימה ופעם שניה אחרונה. כלומר, לאחר ביצוע הסינון קדימה היא הופכת את אות המוצא ומפעילה שוב את הסינון. לאחר סינון זה, לא נקבל עיוותי פאזה כלומר ללא הזזה בזמן. פעולה זו מסבירה את תוצאת הסינון הטובה יותר שראינו באיור 6.

ניסוי 2: Machine learning for AF classification

2.1

Artial fibrillation פרפור עליות הוא קצב מאוד מהיר של כיווץ העליות בקצב של 400-700 פעימות לדקה, ולכן במצב זה העליות אינן מתכווצות בצורה מתואמת אלא רועדות או מפרפרות. בשל כך הדם יזרום לאט יותר מעליות ולכן מצב זה יכול להוביל למצבים מסכני חיים כמו קרישי דם, שבץ וסיבוכים נוספים. בשל כך, בציר הזמן נקבל קו בסיס עם גלים במקום גלי P כפי שמופיע באיור 2, ואילו בציר התדר נקבל מרכיב תדרי גבוה יותר עבור גלים אלו בשל הקצב הגבוה. [5]



איור 8: אות ECG בעת פרפור עליות [11]

2.4

נשתמש בפונקציה שעושה עיבוד מקדים לסיגנל הכוללת:

- הורדת DC ע״י חיסור הממוצע מהאות
- הורדת baseline על ידי מסנן fir מסוג HPF בתדר של 0.5 הרץ
- הורדת רעש רשת על ידי מסנן נוטץ מסוג IIR בתדר של 50 הרץ
- הורדת רעש נשימה באמצעות מסנן (IIR) butterworth מסוג HPF בתדר של 0.5 הרץ

2.5

הפיצ׳רים בהם נשתמש לצורך זיהוי של AF:

מדדים בתחום הזמן-

- מספר חציות האפס – עבור סיגנל ללא DC, נצפה כי מתוך צורת הAF יהיו יותר חציות של אפס בין קומפלקסי הQRS ולכן מספר חציות האפס הכולל של הסיגנל יעלה.
- קצב הלב- כפי שהסברנו ב4.1, עבור AF קצב הלב גדל ולכן נצפה לקבל הפרדה באמצעותו בין המצבים. את קצב הלב נחשב באמצעות זיהוי אינטרוולי הRR על ידי אלגוריתם פן טומפיקנס כאשר $HR = \frac{60}{mean(RR)}$
- SDNN – מדד שמחשב את סטיית התקן בין מרווחי הRR. נצפה שבעת מצב של AF השונות בין מרווחי הRR תגדל ולכן נצפה שמדד זה יתן הפרדה טובה. הוא מחושב על ידי הנוסחה: $SDNN = \sqrt{\frac{1}{M-1}\sum_{k=1}^{M}(r_k - mean(r))^2}$
- rMSSD- זהו מדד שמנתח את ההפרשים בין מרווחי RR עוקבים ולכן נצפה שבעת מצב של AF המרווחים לא יהיו תקינים. מדד זה מחושב על ידי

$$rMSSD = \sqrt{\frac{1}{M-1}\sum_{k=2}^{M}(r_k - r_{k-1})^2}$$

מדדים בתחום התדר-

- HAPO – זהו ערך התדר המחלק את הספקטרום לשני שטחים השווים בגודלם. עבור מצבים של AF טווח התדרים גדל משמעותית ולכן נצפה שמדד זה יגדל גם ויתן הפרדה משמעותית בין מצב של ECG תקין לבין AF.
- Spectral centroid- מדד זה נותן אפיון ספקטרלי של האות, כאשר הוא מחשב את מרכז המסה של הספקטרום באמצעות ממוצע משוקלל של התדרים. גם כאן נצפה שיהיה הבדל משמעותי בין מצב תקין למצב של AF שכן טווח התדרים שונה משמעותית.
- MDF- מדד זה הינו התדירות החציונית. גם כאן, עבור טווח תדרים גבוה יותר נצפה לתדירות חציונית גבוהה יותר.

2.10

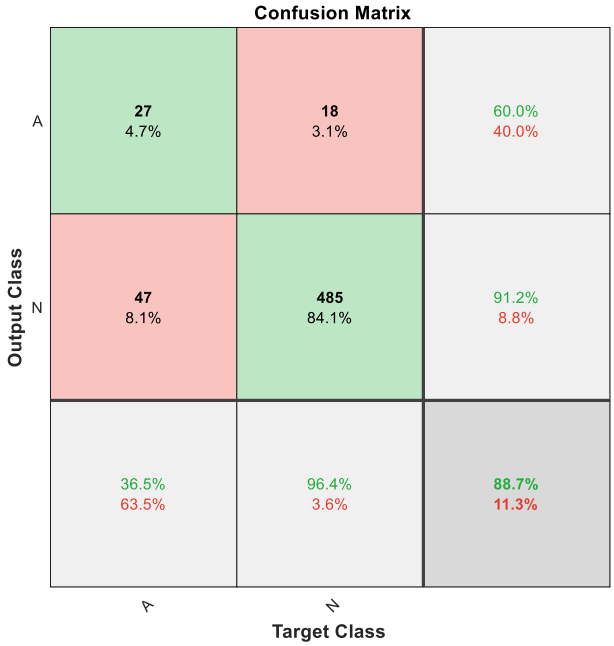
עצי החלטה- משתמש בפיצ׳רים ובונה עצי החלטה הכוללים סיווג לקבוצות הולכות וקטנות. כלומר, בהתחלה האלגוריתם יבדוק מאפיין מסוים ויחלק את התוצאות ל2, לאחר מכן כל פיצול כזה יעבור בדיקה של מאפיין אחר וכך הלאה עד להחלטה. הבחירה של המאפיינים מתבצעת על ידי בדיקה בכל שלב מהו המאפיין שייתן את ההפרדה המקסימלית.

Naïve bayes -אלגוריתם זה משתמש במשפט ההסתברות של בייס ובהנחה שאין תלות בין הפיצ׳רים. משפט ההסתברות של בייס: $P(A|B) = \frac{P(B|A)\cdot P(A)}{P(B)}$. האלגוריתם משתמש במשפט זה כדי לסווג ל*K* קטגוריות (במקרה שלנו *N* או *A*),כדי לבחור ביניהם הוא מחשב את ההסתברויות המותנות להיות בכל קטגוריה בהינתן הפיצ׳רים ובוחר את זו עם ההסתברות הגבוהה ביותר.

SVM- מוצא מישור בממד גבוה בו ניתן לעשות הפרדה בין הקטגוריות בצורה הטובה ביותר. בהינתן דאטה אותו נרצה לסווג, האלגוריתם מחשב את המרחק שלו מכל אחד מהמישורים (של כל אחד מהקטגוריות) ובוחר במישור עם המרחק הכי קטן להיות הסיווג של הדאטה הנבדק. אלגוריתם זה עושה שימוש בקרנלים שונים כדי לעשות את מעבר לממד הגבוה.

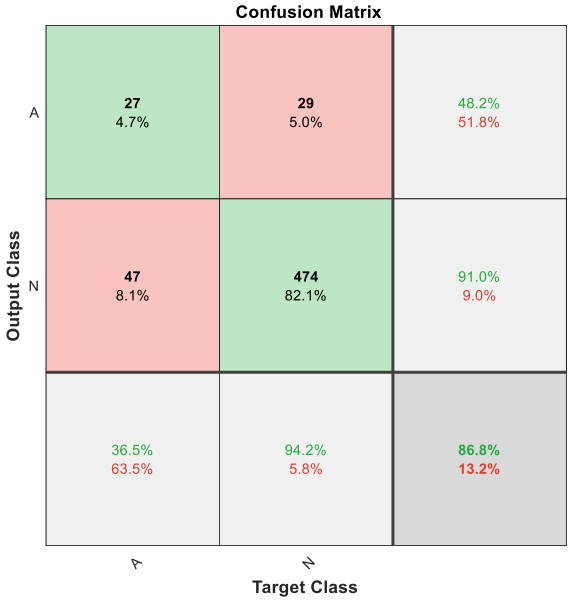
2.11

לאחר מכן, ייבאנו את שלושת המודלים הנ״ל וביצענו סיווג עבור דאטה המבחן. מטריצות המבוכה שקיבלנו עבור דאטה המבחן הן:



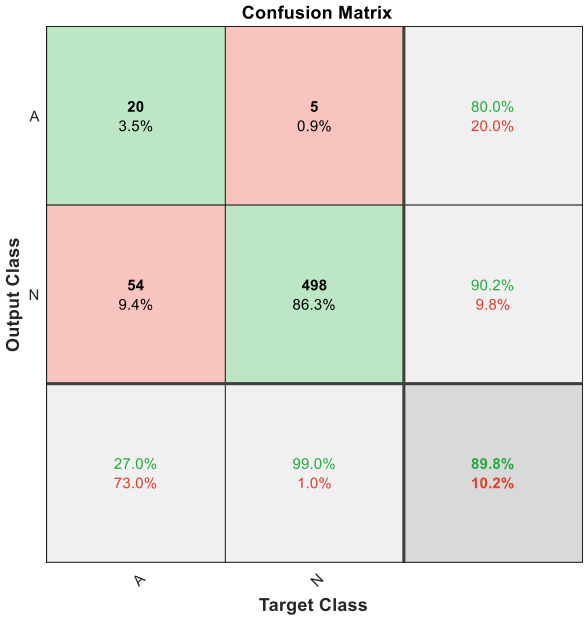
איור 9 : מטריצת בלבול עבור Fine Tree

מאיור זה ניתן לראות כי אחוז ההצלחה בזיהוי AF עבור עצי ההחלטה הינו 36.5% ועבור ECG תקין הינו 96.4%. דיוק האלגוריתם באופן כללי הינו 88.7%.



איור 10 : מטריצת בלבול עבור naive bayes

מאיור זה ניתן לראות כי אחוז ההצלחה בזיהוי AF עבור naïve bayes הינו 36.5% ועבור ECG תקין הינו 94.2%. דיוק האלגוריתם באופן כללי הינו 86.8%.



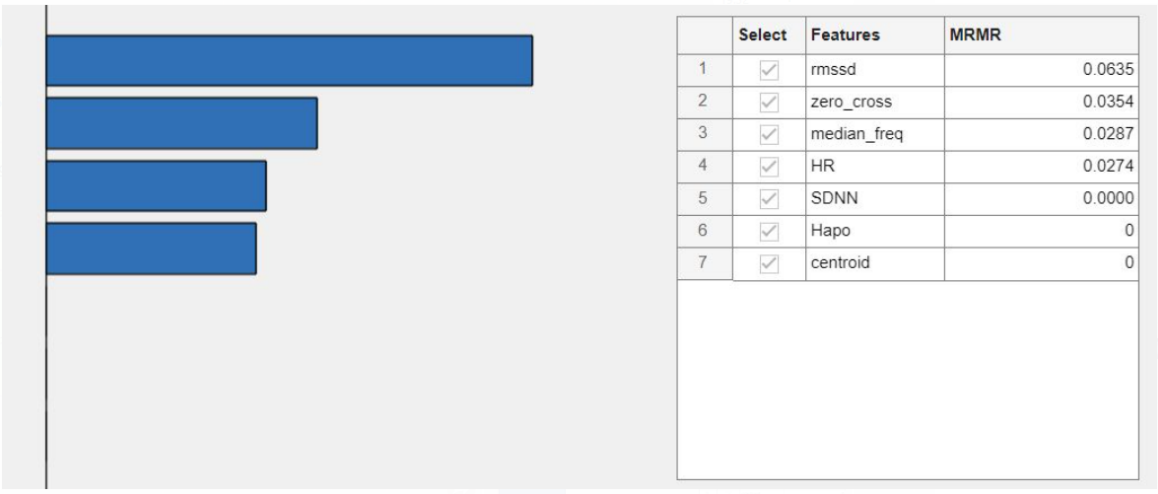
איור 11 : מטריצת בלבלול עבור SVM

מאיור זה ניתן לראות כי אחוז ההצלחה בזיהוי AF עבור SVM הינו 27% ועבור ECG תקין הינו 99%. דיוק האלגוריתם באופן כללי הינו 89.8%.

מאיורים אלו נסיק כי האלגוריתם שמבצע את הסיווג הטוב יותר הינו ה Fine tree בעזרתו קיבלנו אחוזי דיוק טובים הן עבור זיהוי הAF והן עבור זיהוי סיגנל תקין.

כמו כן, ניתן לראות כי כל שלושת המודלים הצליחו לזהות בצורה טובה יותר את הECG התקין וזאת משום שבאלגוריתם הלמידה אין איזון בדאטה ויש הרבה יותר סיגנלים תקינים. ניתן לשפר את יעילותם על ידי הוספת סיגנלים עם AF ללמידה וכך המודל יוכל לבחון מקרים נרחבים יותר ואולי להצליח לבצע זיהוי טוב יותר עבור מצבים אלה.

בניסוי זה עשינו שימוש בפיצ׳רים נוספים מלבד אלה שרשמנו ב2.5 כמו מדד *pnn50*, והפיצ׳רים המופיעים כאן הניבו את התוצאות הטובות ביותר. עם זאת, ניתן לראות שגם תוצאות אלה הינן מיטביות כיוון שעבור זיהוי ה*AF* אחוזי ההצלחה נמוכים (אחוזי דיוק גבוהים נובעים מכך שהצלחנו לזהות הרבה מקרים של סיגנל תקין). נסיק כי ניתן לבצע שיפור של האלגוריתם באמצעות דרכים נוספות כמו רשתות נוירונים ומודלים מורכבים יותר.



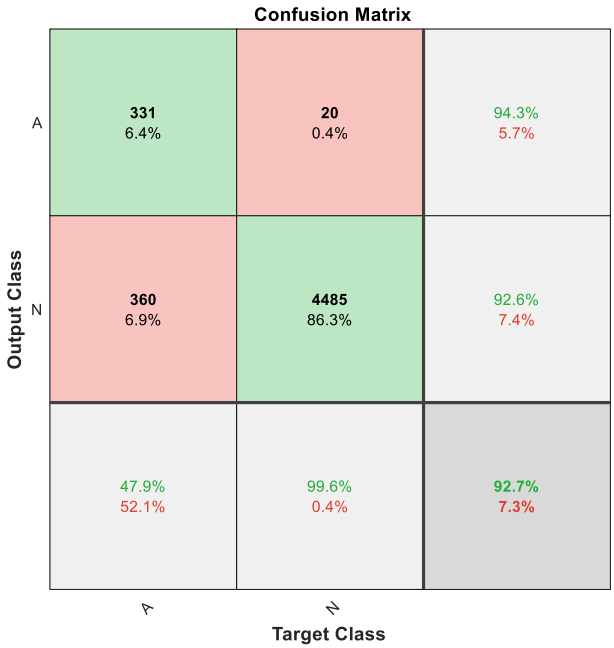
	Select	Features	MRMR
1	✓	rmssd	0.0635
2	✓	zero_cross	0.0354
3	✓	median_freq	0.0287
4	✓	HR	0.0274
5	✓	SDNN	0.0000
6	✓	Hapo	0
7	✓	centroid	0

איור 12: חשיבות הפיצ'רים השונים במודל

באמצעות אלגוריתם MRMR של ה – classification learner חילצנו את חשיבות כל פיצ'ר במודל. ניתן לראות כי המדדים המשמעותיים ביותר הינם – rmssd, ZCR, median frequency ודופק. שאר המדדים לא משמעותיים כלל למודל לפי האלגוריתם. בהתאם להסברים בסעיפים הקודמים ניתן להבין מדוע מדדים אלה התקבלו כמשמעותיים למודל.

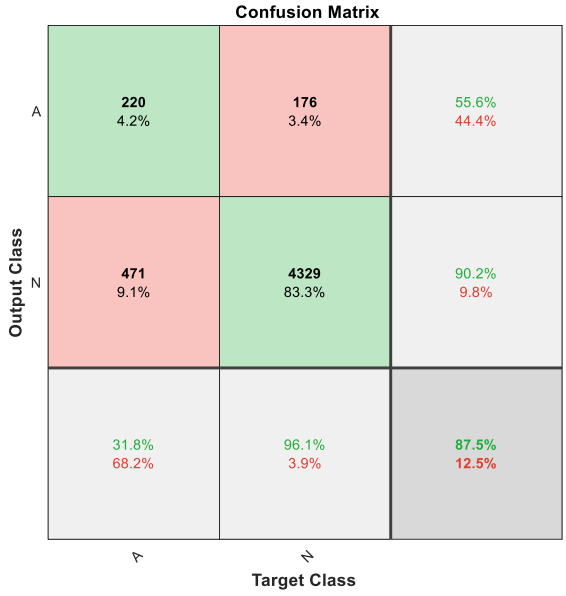
2.12

סיווג עבור נתוני האימון:



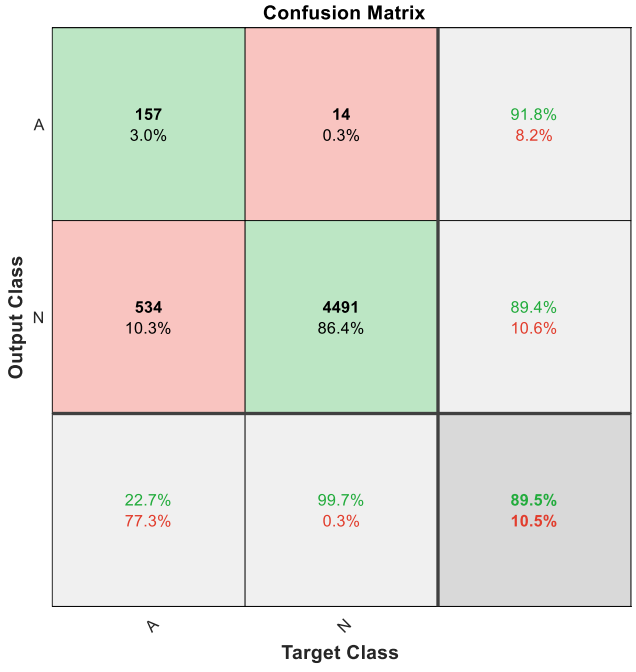
איור 13: מטריצת בלבלול עבור Fine Tree

מאיור זה ניתן לראות כי אחוז ההצלחה בזיהוי AF עבור עצי ההחלטה הינו 47.9% עבור ECG תקין הינו 99.6%. דיוק האלגוריתם באופן כללי הינו 92.7%



איור 14 : מטריצת בלבול עבור Naive bayes

מאיור זה ניתן לראות כי אחוז ההצלחה בזיהוי AF עבור naïve bayes הינו 31.8% עבור ECG תקין הינו 96.1%. דיוק האלגוריתם באופן כללי הינו 87.5%.



איור 15 : מטריצת בלבול עבור SVM

מאיור זה ניתן לראות כי אחוז ההצלחה בזיהוי AF עבור SVM הינו 22.7% ועבור ECG תקין הינו 99.7%. דיוק האלגוריתם באופן כללי הינו 89.5%.

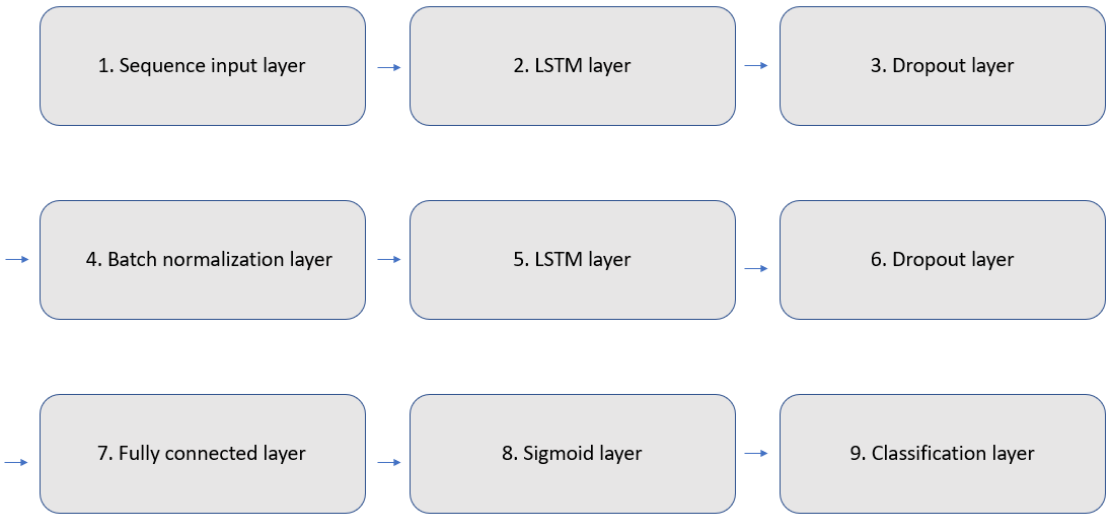
מאיורים 11-13 ניתן לראות כי תוצאת האלגוריתם על דאטה המבחן טובה יותר עבור זיהוי AF בכל שלושת המודלים. תוצאה זו הגיונית מכיוון שהאלגוריתם למד את תכונות הקטגוריות על דאטה זה ולכן הוא מכיר את השוניות והרעשים שקיימים בהם. לעומת זאת, כאשר האלגוריתם מקבל דאטה חדש הוא מבצע את הסיווג על פי המידע שכבר למד ולכן במקרה של רעשים למשל שלא הכיר יכולות להיות לו טעויות בזיהוי.

לסיכום נסיק כי אלגוריתם ה- Fine Tree הניב את התוצאה הטובה ביותר בהפרדה בין סיגנל ECG תקין לבין מצב של AF שכן אחוז הדיוק שלו על דאטה המבחן היה הגבוה ביותר בזיהוי הAF.

ניסוי 3: Deep learning for AF classification

3.1

דיאגרמת בלוקים של השכבות השונות:



איור 16: דיאגרמת בלוקים של השבכות

3.2

[12]

1. Sequence input layer:

שיכבה זו נועדה לבצע טרנספורמציה לאות הכניסה כך שיהיה מתאים לעיבוד ע"י שאר השכבות ברשת. הפקטור input size השווה ל – 1 משמעותו שיש מאפיין אחד באינפוט שאנו מכניסים, במקרה זה מדובר באמפליטודה של אות ה – ECG.

2. LSTM layer:

LSTM – Long short-term memory. שכבה זו מסוגלת לזהות דפוסים במידע וללמוד תלות ארוכת טווח בין חלקים שונים של אות הכניסה. שכבה כזו עובדת ע"י בחירה סלקטיבית של איזה מידע להחזיק ואיזה מידע לשחרר ומתי, בצורה שתאפשר את השגת הרשום מעל. ההיפר פרמטרים של שכבה זו הם:

- מס' הנוירונים בשכבה (lstm1) – 256. ככל שנבחר בכמות נוירונים גדולה יותר, כך נוכל לזהות מס' רב יותר של פיצ'רים ולמצוא קשרים יותר מורכבים במידע, אך נוסיף סיבוכיות חישוב ונסתכן באובר פיטינג למידע, עלכן חשוב לאזן בין שיקולים אלה בבחירת כמות הנוירונים של כל שכבה ברשת.
- פקטור נרמול למשקולות (reg) – 0.01. פקטור זה קובע כמה להתחשב בגודל המשקולות בפונקציית המחיר. חלק מפונקציית המחיר הוא סכום ריבועי המשקולות מוכפל בפקטור הנרמול. ככל שהפקטור יותר גדול כך המחיר של משקולות גדולות יגדל מה שיוביל להקטנתן. כלומר, ככל שפקטור זה גדול יותר, נמנע מהמשקולות לגדול בצורה יותר אגרסיבית, דבר הנועד למנוע אובר פיטינג של המשקולות לדאטה האימון ולאפשר הכללה טובה יותר של המודל לאינפוטים שונים.

- משקולות התחלתיות ('inputWeightInitializer','narrow-normal') – יצירת וקטור משקולות ראשוני לכל נוירון. הבחירה 'narrow-normal' הינה שהמשקולות הראשוניים בעלי התפלגות נורמלית עם ממוצע אפס ושונות בעלת יחס הפוך לכמות הנוירונים בשכבה (256). ההתפלגות נקראת צרה כיוון שאם לרשת יש יותר מנוירון אחד, השונות תהיה קטנה מאחת, השונות של התפלגות נורמלית סטנדרטית. בחירה זו נועדה להוות ריסון נוסף למשקולות כך שההכללה של המודל תהיה טובה יותר.
- מוצא הנוירונים הוא רצף או התוצאה האחרונה ('OutputMode', 'sequence') – קובע האם המוצא של כל נוירון הוא רצף המשכי של תוצאת פעולת השכבה על כל האינפוטים או רק עבור הכניסה האחרונה. נבחר ב – 'sequence' כאשר צריך להמשיך להשתמש במידע מזמנים שונים להמשך העיבוד, ונבחר ב – 'last' כאשר דרוש רק המצב הסופי של השכבה.

3. Dropout layer:

שכבה זו מסירה באופן רנדומלי חלק מהנוירונים בכל איטרצית למידה של הרשת (כלומר בכל איטרציה יוסרו נוירונים שונים). ההיפר פרמטר שלה הוא 'do' (drop out), והוא קובע את אחוז הנוירונים שיוסרו. מטרת שכבה זו היא למנוע אוברפיטינג למידע והכללה טובה יותר של המודל, ע״י כך שהוא לומד על ייצוגים שונים של מידע הלמידה (דומה במידה מסוימת לקרוס ולידציה).

4. Batch normalization layer:

שכבה זו מבצעת נרמול של שכבת האקטיבציה האחרונה (במקרה זה שכבת ה – LSTM). הפעולה נעשית ע״י הפחתת הממוצע וחלוקה בסטיית התקן של מקטע מסוים ממוצא השכבה המאקטבת האחרונה (עבור כל נוירון בנפרד). פעולה זו נועדה לייצב את ההתפלגויות הנוצרות מכל נוירון של השכבה המאקטבת, וכן להפחית את ה – covariate shift – מצב בו יש לדאטה הלמידה התפלגות שונה משל דאטה המבחן. דבר זה גורם גם הוא להפחתה של אוברפיטינג ומשפר את הכללת המודל. היפר פרמטר של שכבה זו הינו אפסילון (0.001), מספר המוסף לשונות של כל batch כדי להימנע מחלוקה באפס.

5. LTSM layer:

שכבת LTSM נוספת. הפעם עם 128 נוירונים ומצב Output של 'last'.

6. Dropout layer:

שכבה נוספת של DO עם אותו פרמטר לשם שיפור הכללת המודל.

7. Fully connected layer:

שכבה זו משתמשת בכל הפיצ׳רים שנוצרו עד כה מהשכבות הקודמות, ומסווגת את האות לקלאסים, שכמותם נקבעת לפי הפרמטר 'numClasses'. בקוד הנתון הוא מגדר ל – 2 קלאסים (ECG רגיל או כזה שמתרחש בו AF). לכל קלאס ניתן ציון.

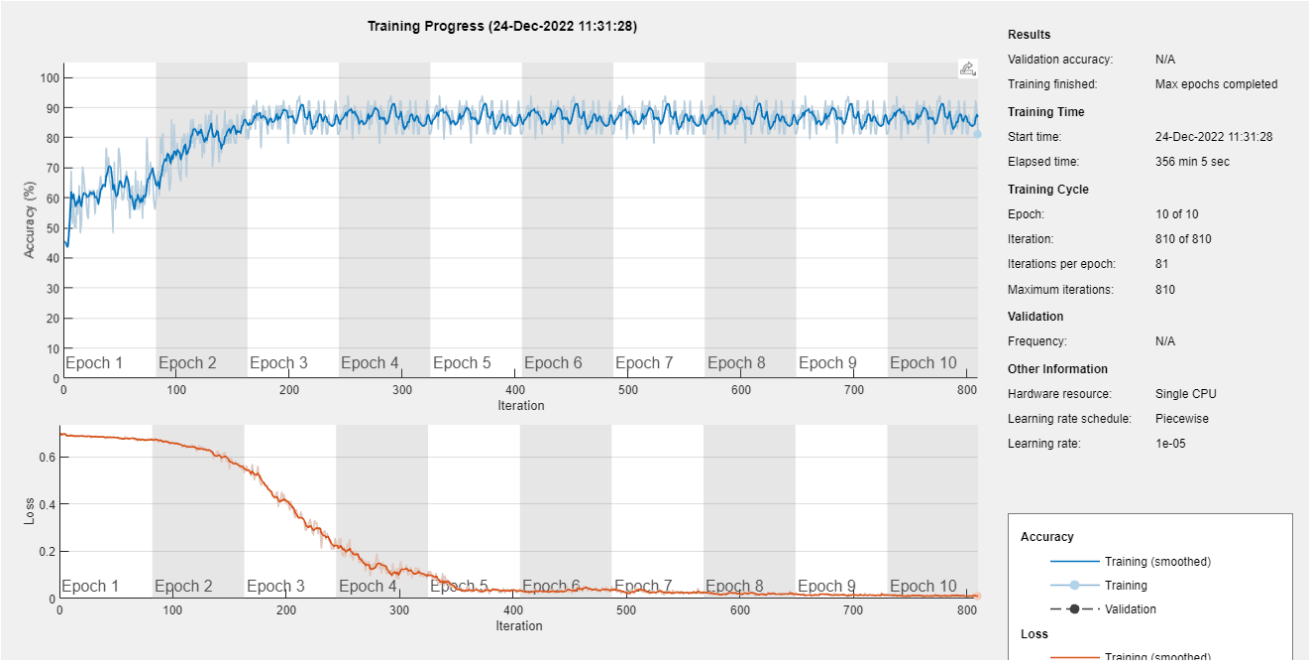
8. Sigmoid layer:

שכבה זו מיישמת פונקציה סיגמואידית על ציוני הקלסיפיקציה הממפה את כל המרחב הממשי לתחום [0,1], כך שהציון מקבל משמעות של הסתברות.

9. Classification layer:

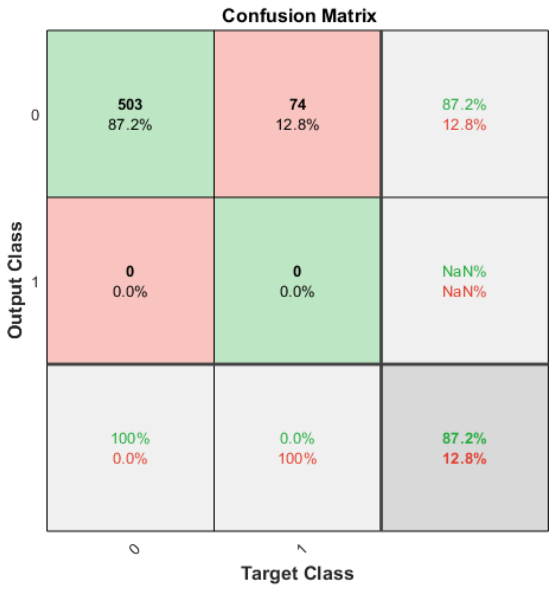
שכבה זו מגדירה את המוצא הסופי להיות הקלאס בעל ההסתברות המקסימלית מתוך ההסתברויות לקלאסים שחושבו בשכבה הקודמת.

.3.3+3.4



איור 17: עקומת הלמידה של הרשת הראשונה (ללא המשקולות)

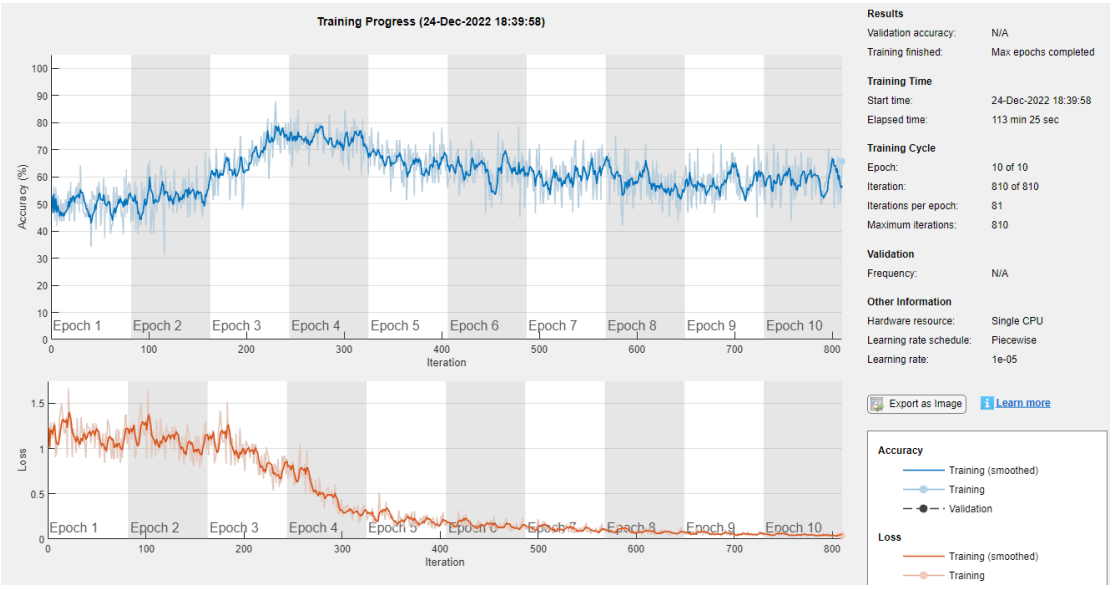
ניתן לראות כי בקירוב לאחר שני Epoch עקומת הדיוק מתכנסת לאזור ה – 80~90%, בעוד עקומת הלוס פנקשין ממשיכה לרדת ומתכנסת רק באזור 6 Epoch. המשך האימון לאחר ההתכנסות של פונקציית הדיוק יכול להעיד על כך שהמודל המשיך לבצע התאמות למידע האימון ובכך יוצר אוברפיטינג של המודל למידע האימון.



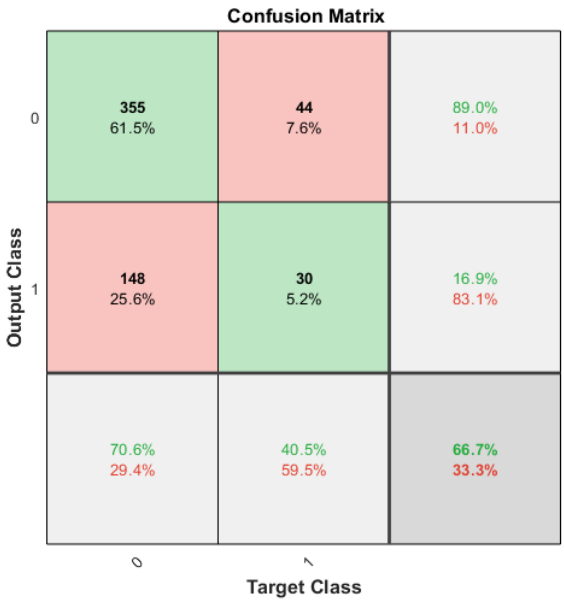
איור 18: מטריצת המבוכה של מידע המבחן עבור המודל הראשון

אחוז הדיוק של המודל על נתוני המבחן הינו 87.2%. על פניו זה אחוז דיוק גבוה, אך נשים לב כי 100% מהזיהויים הם של מצב בו אין AF, כלומר אין אף זיהוי (שגוי ולא שגוי) של AF, ואחוז הדיוק הגבוה יחסית נובע מכך ש – 87.2% ממידע המבחן, וכנראה אחוז דומה מדאטה הלמידה הוא של מצבים שאין בהם AF. יש חוסר איזון משמעותי במידע שברשותנו. במודל הלמידה לא מומש אף אמצעי המתייחס לחוסר האיזון במידע, פקטור משמעותי לסיווג בין קלאסים.

.3.6



איור 19: עקומת הלמידה של הרשת השנייה



איור 20: מטריצת המבוכה של מידע המבחן עבור המודל השני

ראשית, נשים לב כי הפעם בדיוק מתייצב בסביבת 60~%70, ואכן הדיוק הסופי של המודל הינו 66.7%. הפעם, עקומת הדיוק ועקומת ה – Loss מתכנסות בזמנים דומים, כך שייתכן שהמודל הנ"ל יותר מוכלל מהמודל הראשון. מהתבוננות במטריצת המבוכה, נראה כי הדיוק של המודל נמוך יותר, אך עבור מודל זה ישנם יותר זיהויים של AF (שגויים ונכונים). במודל זה, כיוון שבקלסיפיקציה הסופית בחרנו משקול לקלאסים (יחס המשקולות בערך ביחס של חוסר האיזון בדאטה), קיבלנו זיהויים של AF. אם זאת, אחוז הזיהויים הנכונים של מצב בו יש AF מסך הזיהויים של AF הינו 16.9%, ואחוז הזיהוי של AF מסך המקרים בהם התרחש AF הינו 40.5%. כלומר, הוספת המשקול, המתחשבת בחוסר האיזון בדאטה הובילה ליותר זיהויים של AF, אבל 83.1% מהם לא נכונים ורק 40.5% מסך ה – AF בדאטה המבחן זוהו.

3.7

כיוון שהמודל הינו קלסיפיקציה בינארית והשכבה לפני האחרונה היא שכבה סיגמואידית, ישנה סבירות טובה כי פונקציית המחיר הינה פונקציית מחיר מסוג Cross-Entropy אשר הגדרת:

$$L = -\frac{1}{N}\sum \log(p_t)$$

כאשר סוכמים על כל הדגימות ב - *batch* מסוים, p_t הינה ההסתברות השייכת לתגית האמת של הקלאס (למשל, אם תגית האמת של הדגימה היא 0, ניקח את ההסתברות שהתקבלה שהדגימה שייכת לקלאס המתויג ב - 0. נשים לב כי הסתברות זו היא לא בהכרח המקסימלית מבין השתיים שחושבו) ו – N זה מס' הדגימות ב - *batch*. ניתן לראות מהנוסחה כי ככל שההסתברות של תווית האמת נמוכה יותר, כך פונקציית המחיר תקבל ערך גבוה יותר, כלומר העונש על שגיאה בזיהוי יהיה חמור יותר. כדי לתקן מצב של חוסר איזון בנתונים, הוספת משקולות לפונקציית המחיר יכולה לשפר את הפרדיקציות מכך שניתן לתת משקל גדול יותר למידע המופיע פחות בנתונים, וכך העונש על שגיאה בחיזוי של קלאס ספציפי זה יהיה גדול יותר משגיאה בקלאס שמופיע יותר בנתונים. דבר זה אמור לגרום לשיפור בפרדיקציה הספציפית של הקלאס לו ניתן משקל גבוה יותר. מבחינה מתמטית הנוסחה תראה כך (עבור פונקציית המחיר הנידונה):

$$L = -\frac{1}{N}\sum w_t \log(p_t)$$

כאשר w_t הינה המשקולת המתאימה להסתברות p_t (למשל אם תגית האמת היא 0, ניקח את ההתסברות שחושבה לקלאס 0 ואת המשקולת המתאימה לקלאס 0). ניתן לראות כי בהתאם לערכי w שהינם המשקולות, חומרת העונש עבור שגיאה של כל קלאס תהיה בפרופורציה למשקל שהוא מקבל. גם אם זו לא פונקציית המחיר בה המודל משתמש, הקונספט הינו לתת עונש חמור יותר לקלאס שאנו רוצים לתת לו יותר חשיבות, ובמקרה זה בגלל חוסר האיזון בנתונים ניתן עונש חמור יותר לשגיאות בזיהוי *AF* מאשר לשגיאות בזיהוי מצב תקין. הוקטור $'classweight' =$ $[1,6]$ הוא וקטור משקולות זה, ואם פונקציית המחיר היא זו הנידונה למעלה, העונש על שגיאה בזיהוי *AF* גדול פי 6 משגיאה בזיהוי מצב תקין.

3.8

על פניו, הדיוק של המודל הראשון (87.2%), ללא המשקולות הינו גבוה יותר משל המודל השני (66.7%). כיוון שבמודל הראשון אין התייחסות לחוסר האיזון בנתונים, התקבל מצב בו כל התחזיות הן של מצב תקין, וכיוון שמצב זה רווח מאוד בנתונים שבידנו (~87%) הדיוק המתקבל גבוה. אם זאת, מטרתנו הייתה להצליח לזהות מצבי AF, ומודל זה נכשל באופן מוחלט בזיהוי AF. במודל השני, אחוז הדיוק נמוך יותר, אבל ישנם זיהויים של מצבי AF. הזיהויים התקבלו מכך שבמודל זה ישנו משקול לקלאסים, הגורמים לעונש גדול יותר עבור שגיאות בזיהוי AF, כלומר מעודדים את המערכת לבצע פחות שגיאות בזיהוי AF מאשר שגיאות בזיהוי מצב תקין. בחירה זו אומנם הובילה ליותר זיהויים של AF, אך גרמה להגדלה גם של זיהויים לא נכונים של מצב תקין כ – AF. יש פה טרייד אוף מסוים – מצד אחד לא רוצים לפספס מצבי AF ומצד שני לא רוצים להגדיר מצב תקין כ – AF. אחוז הזיהוי של AF מסך המקרים בהם התרחש AF הינו 40.5%, נתון שגם הוא אינו מזהיר, כלומר יש לנו הרבה (מאוד) זיהויי יתר של AF וגם יותר מ – 50% ממצבי ה – AF פוספסו. לדעתנו, זו תוצאה יותר טובה משל המודל הראשון אך גם תוצאה זו אינה טובה מספיק. ייתכן שדרוש מודל שמבצע משקול לקלאסים בדומה למודל השני, אך גם יותר מורכב לשם הפרדה טובה יותר בין שני הקלאסים. בבחירה זו נסתכן בהכללה פחות טובה של המודל, ולכן צריך לזכור לאזן בטרייד אוף בין מורכבות המודל והצלחתו על מידע הלמידה להכללה שלו.

3.9

ישנם מספר גישות להתמודדות עם מצב של חוסר איזון במידע, ביניהן:

1. דגימה מחודשת של המידע – אפשרות אחת הינה *under sampling* של הדגימה העודפת (מצב תקין), כלומר הורדה רנדומלית מתוך הקלאס שיש לנו עודף ממנו כך שכמות הדגימות תהיה שווה. לחילופין ניתן לבצע *over sampling* של הקלאס שיש לנו חוסר בו (מצב של *AF*), כלומר ליצור דגימות סינתטיות מתוך הדגימות של מצב *AF* כך שנקבל כמות זהה של דגימות.
2. שימוש בפונקציית מחיר שיש לה רגישות לחוסר איזון במידע – *focal loss function*. הגדרת המתמטית הינה:

$$L = -\frac{1}{N}\sum \alpha_t (1-p_t)^{\gamma} \cdot \log(p_t)$$

כאשר p_t כמו שהוסבר בסעיף 3.7, וכמו כן:

$$\alpha_t = \begin{cases} \alpha \ , & y_{act} = 1 \\ 1-\alpha \ , & y_{act} = 0 \end{cases}$$

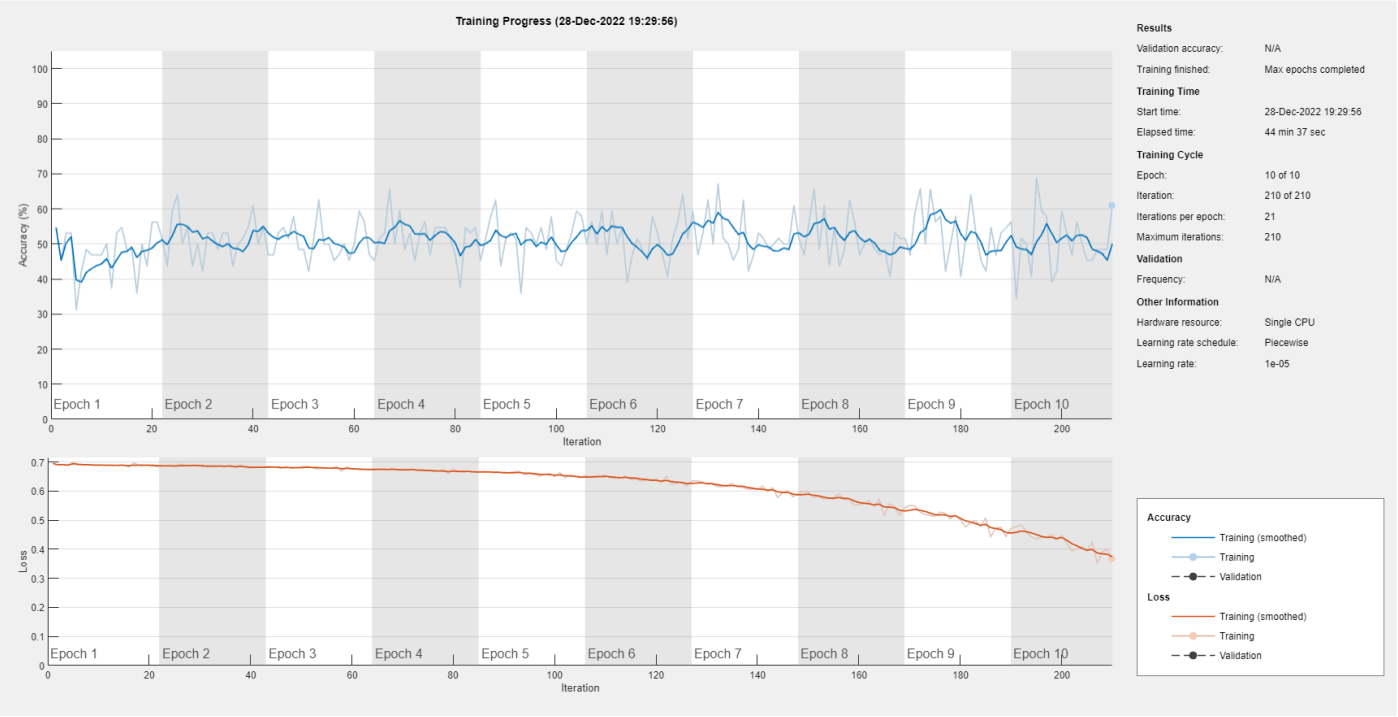
α הינו היפר פרמטר של פונקציית המחיר. נשים לב כי אם נבחר $\alpha = 1$ פונקציית המחיר תתייחס רק לשגיאות של זיהוי AF, אם נבחר $\alpha = 0$ נתייחס רק לשגיאות בזיהוי מצב תקין, ו- $\alpha = 0.5$ נותן משקל שווה לשניהם. משחק עם היפר פרמטר זה דומה למשקולות שהשתמשנו בהן במודל השני. מה ששונה בנוסחה זו משימוש במשקולות על הקלאסים הינו הפקטור $(1-p_t)^\gamma$. בעוד שהמשקולות וכן הפקטור α_t מתייחסים לחוסר האיזון המספרי בדאטה, הפקטור שהוצג כרגע מתייחס לקושי בזיהוי דגימות מסוימות. בפועל, הפקטור יכול להוריד את המשקל לדגימות שזוהו בהצלחה עם הסתברות גבוהה, כך שפונקציית המחיר תתמקד בדגימות שזוהו לא נכון. ככל שהפרמטר γ גדול יותר, כך הורדת המשקל תהיה משמעותית יותר. אפשר לראות כי כאשר p_t קטן, כלומר הקלאס הנכון זוהה בהסתברות נמוכה, לא תהיה הקטנה של מוצא פונקציית המחיר, אך כאשר - $p_t \to 1$, נקבל הנמכה משמעותית של ערך פונקציית המחיר בנקודה זו, דבר שיאפשר למודל להתחשב יותר בקלסיפקציות שגויות מכאלה שנחזו נכון.

[13]

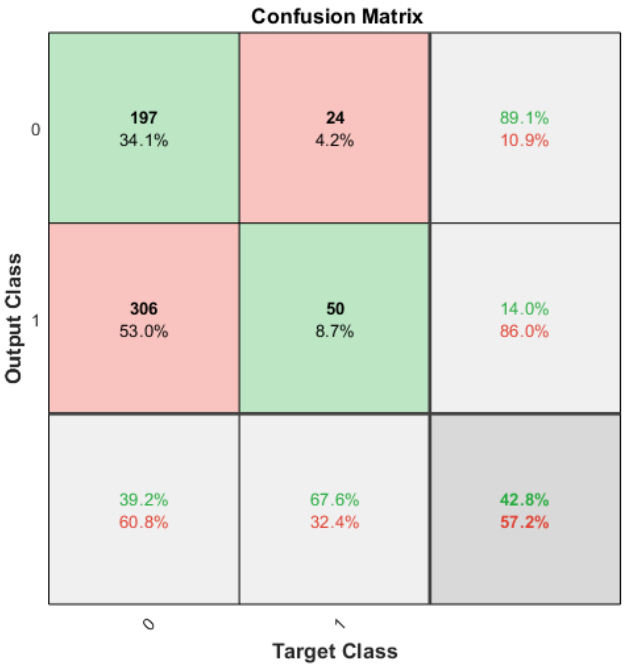
3. *Ensemble methods*:

בשיטה זו מייצרים מספר מודלים שונים ומשתמשים באיזשהי פונקציה של מוצאי המודלים השונים כדי לקבל חיזוי מדויק יותר. גישה מסוימת מתוך גישות רבות של טכניקה זו הינה להשתמש במודל על סגמנטים שונים של הדאטה למידה, כאשר בכל סגמנט קודם נעשה איזון של המידע ע״י *over\under sampling*, ולבסוף נשתמש בשילוב של המודלים השונים כדי לבצע החלטת חיזוי (למשל, העברת מידע הטסט בכל המודלים, ובחירת ההסתברות הגבוהה ביותר מבין התחזיות של כלל המודלים).

אנחנו בחרנו לממש את הפתרון הראשון שהצענו – ביצוע resampling על מנת לאזן את המידע. ניסינו מס׳ אפשרויות – oversampling של דגימות ה – AF ע״י שכפולי הדגימות, שילוב בין oversampling של דגימות ה – AF ו – under sampling של הדגימות הרגילות, כדי למנוע מצב של שיכפול של אותו מידע מס׳ פעמים וכן under sampling של הדגימות הרגילות בלבד. בשלוש השיטות יצרנו כמו דומה של שני סוגי הדגימות. שתי האפשרויות הראשונות הובילו לאותה תוצאה כמו המודל המקורי ללא המשקולות, בעוד האפשרות השלישית הובילה לתוצאות הבאות (קובץ המודל הינו 'resample_model.mlx'):



איור 21: עקומת הלמידה של המודל לאחר undersampling של דגימות ה – ECG הנורמליות



איור 22: מטריצת המבוכה של המודל לאחר undersampling של דאטה ה – ECG הנורמלי

נשים לב למספר דברים. ראשית, בגלל שכמות דגימות ה – AF קטנה יחסית למידע הנורמלי, לאחר Under sampling של המידע הנורמלי, נותרו משמעותית פחות דגימות מלפני כן. מצב זה הוביל לכך שפונקציית ה – Loss לא התכנסה לאפס. כמו כן, נשים לב כי לאורך הלמידה אין שיפור בדיוק וכי התוצאות הסופיות הן של דיוק הנמוך מהצ'אנס. השיפור היחיד שהתקבל במודל הינו כי המודל יותר מוכלל מהמודל המקורי, אך אמירה זו כמעט חסרת משמעות ביחס לעובדה שהתקבל מודל סיווג בינארי עם דיוק הנמוך מ – 50%. נציין כי ניסינו לעשות שימוש גם בפונקציית focal loss כפונקציית המחיר, אך גם משחק עם פרמטרי הפונקציה – אלפה וגאמה, ומשחק עם שאר פרמטרי המודל הוביל לתוצאות זהות למודל המקורי, או תוצאות גרועות אפילו מאלה שהוצגו באיור 20 (קובץ מודל זה מצורף ושמו 'focal_loss_model.mlx'). ייתכן כי כדי ליישם את המודל באחת מהשיטות שניסינו דרוש לבצע התאמה נוספת של פרמטרי המודל, כגון כמות ה – Epochs, קצב הלמידה של המודל, ועוד.

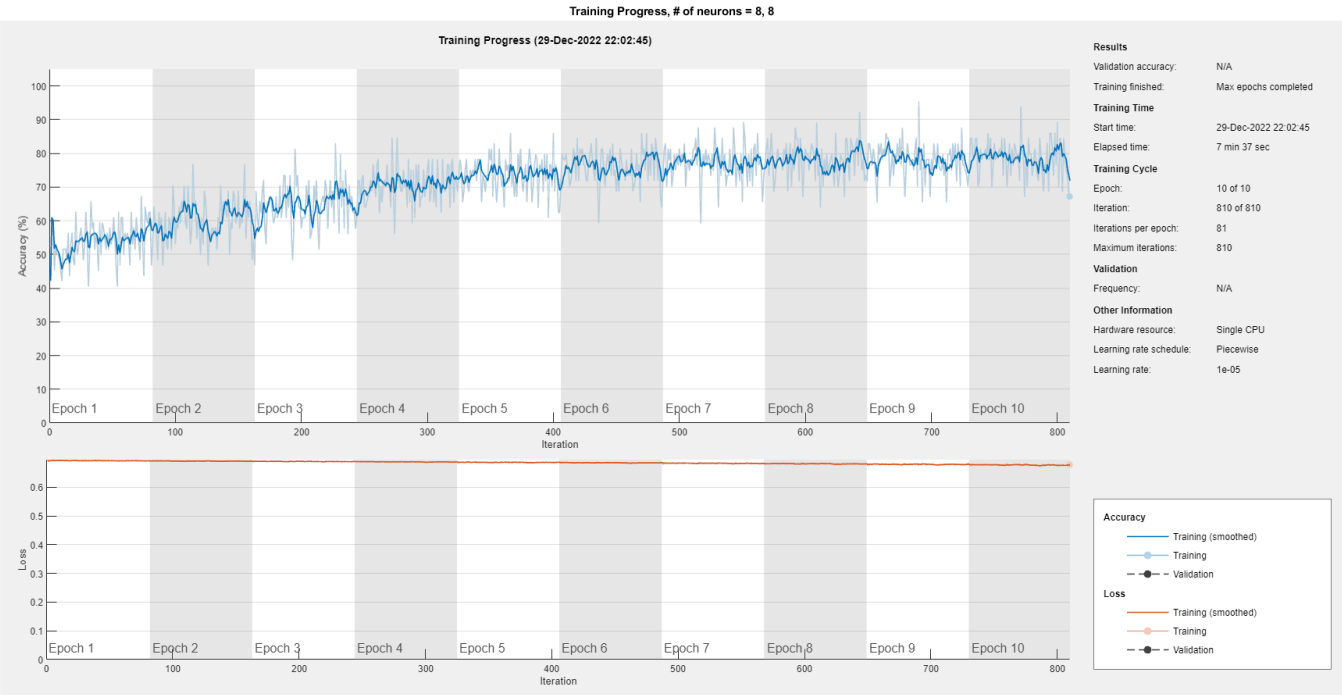
3.10

הפרמטרים השונים של הרשת מוגדרים בתחילת הקוד, ופרמטרי השכבות מוסברים בסעיף 3.2.
הפרמטר שאת השפעתו נבדוק הינו כמות הנוירונים ברשת ה – LSTM. כיוון שאין משמעות גדולה לשינוי כמות הנוירונים רק בשכבה אחת נבדוק את השפעת שינוי כמות הנוירונים בשתי השכבות. נצפה כי כמות נוירונים נמוכה תיצור מודל פשוט מידי שלא יתאים לדאטה הלימוד ולא לדאטה המבחן. כאשר נגדיל את כמות הנוירונים יהיה מורכב יותר, ועד נקודה מסוימת המודל יתאים גם לדאטה הלימוד וגם יבצע הכללה טובה. החל מנקודה מסוימת הגדלת כמות הנוירונים, שיוצרת מודל מורכב יותר תוביל להתאמת יתר של המודל לדאטה הלימוד והכללת המודל תפגע. הערכים שבחרנו עבור כמות הנוירונים במודל הינם:

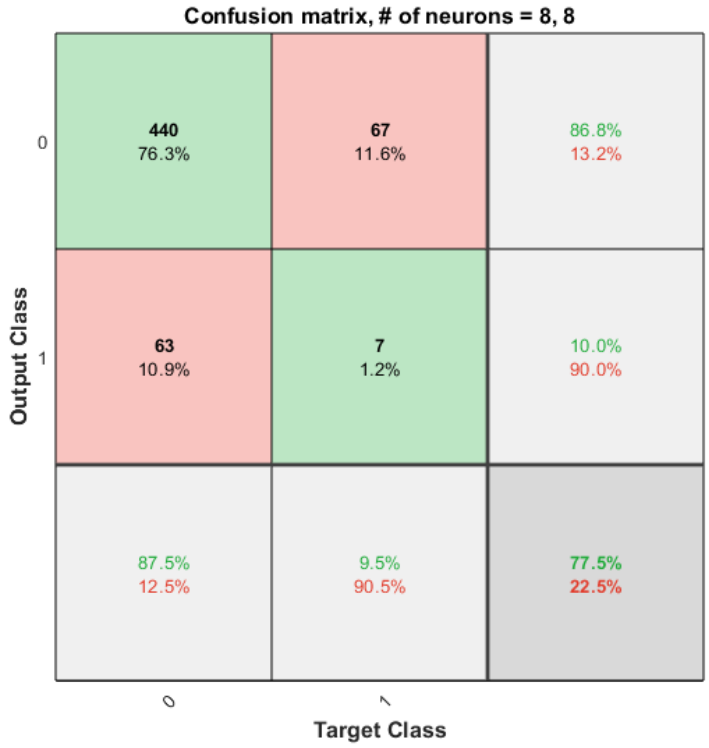
$$LSTM_1 = [8, 16, 32, 128]$$

$$LSTM_2 = [8, 16, 16, 64]$$

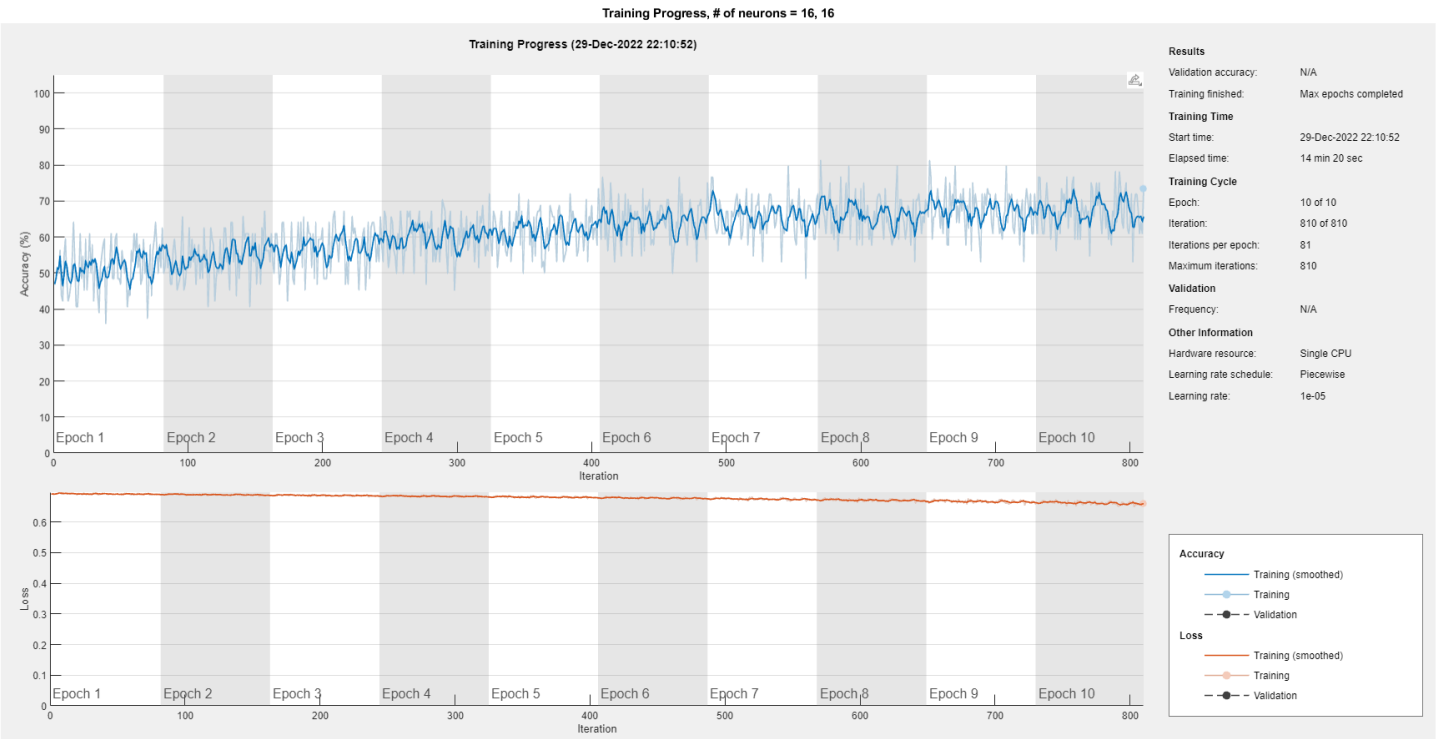
תוצאות המודלים הינן:



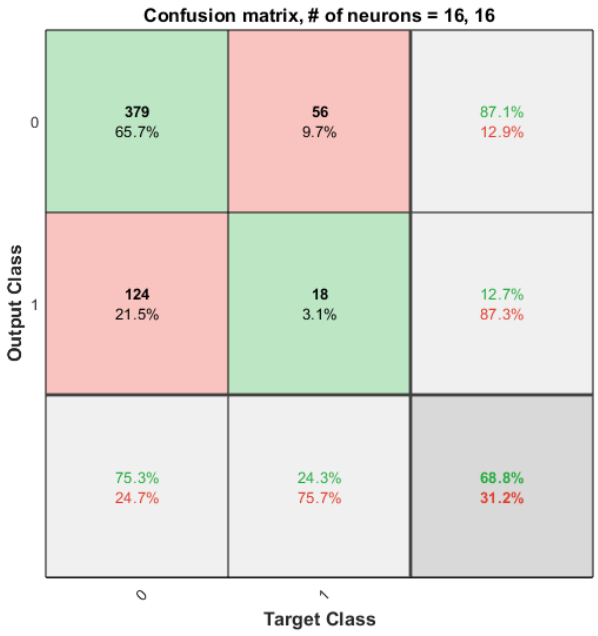
איור 23: עקומת הלמידה, מס' הנוירונים בשכבות ה – LSTM 8, 8 בהתאמה.



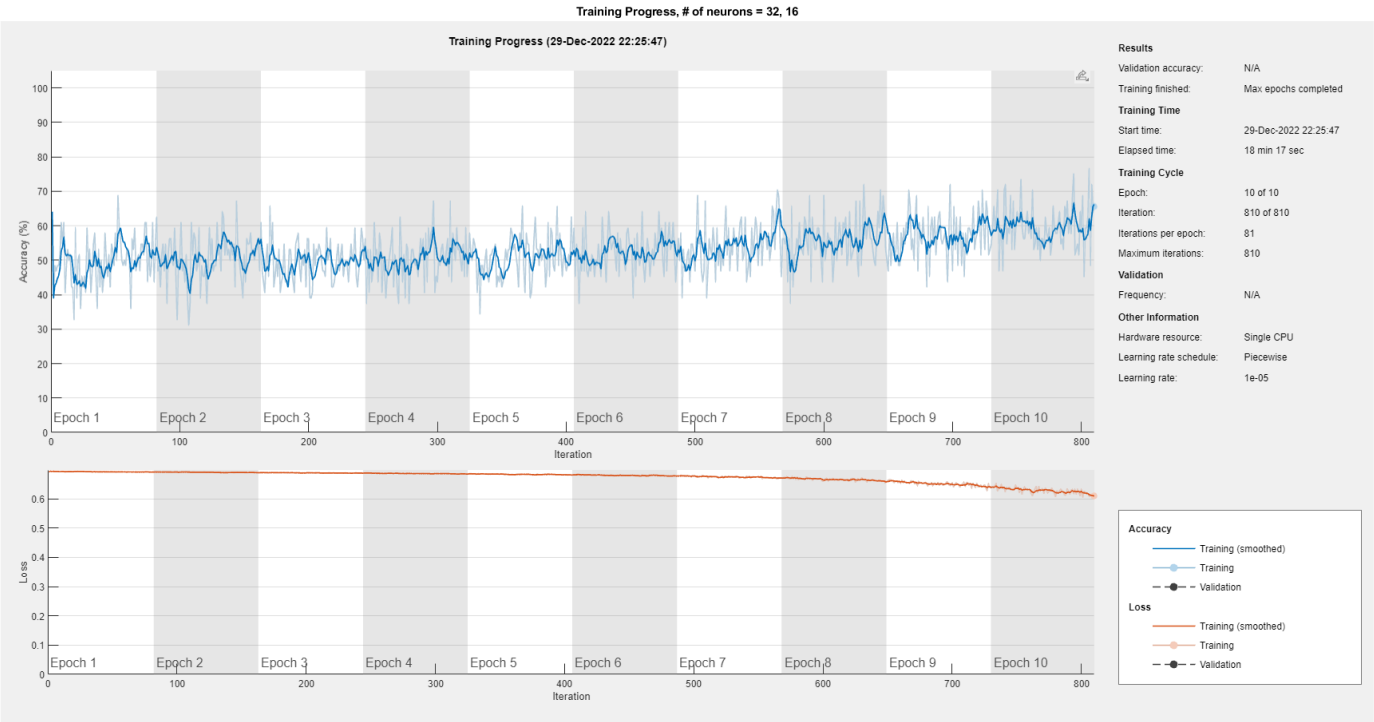
איור 24: מטריצת המבוכה, מס' הנוירונים בשכבות ה – LSTM 8, 8 בהתאמה.



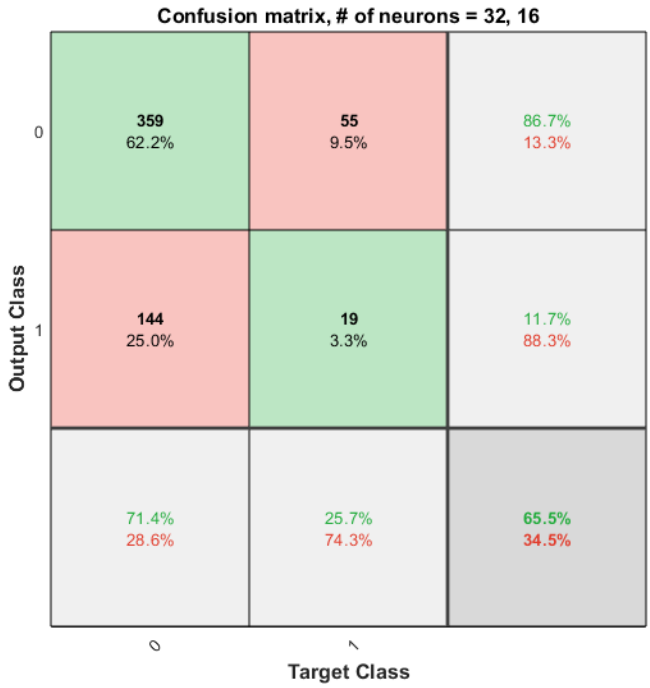
איור 25: עקומת הלמידה, מס' הנוירונים בשכבות ה – LSTM 16, 16 בהתאמה



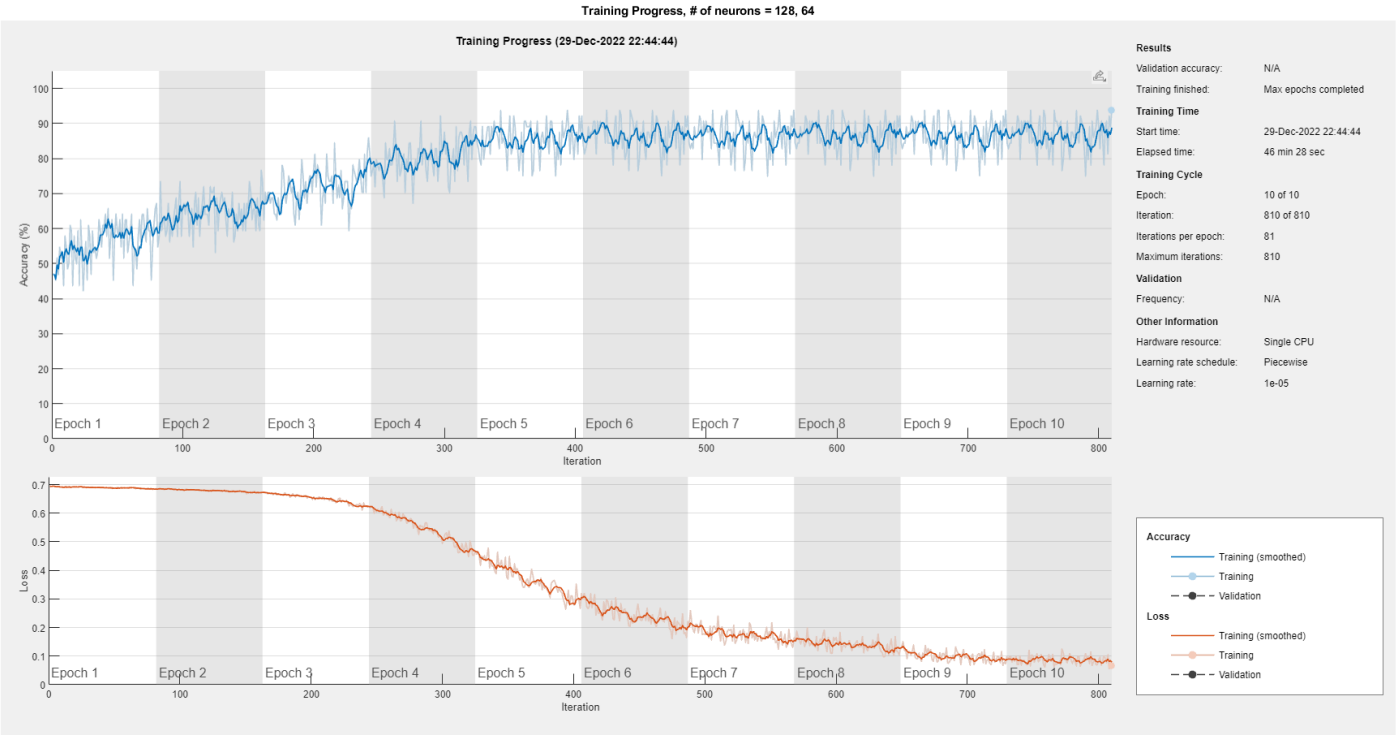
איור 26: מטריצת המבוכה, מס' הנוירונים בשכבות ה – LSTM 16, 16 בהתאמה



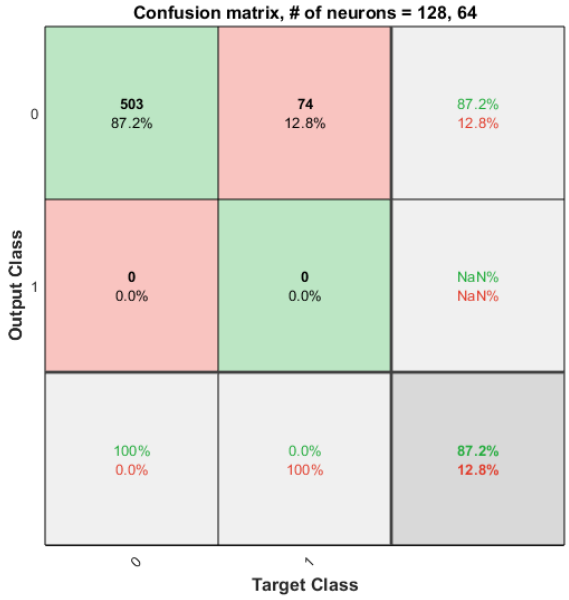
איור 27: עקומת הלמידה, מס' הנוירונים בשכבות ה – LSTM 32, 16 בהתאמה 25



איור 28 : מטריצת המבוכה, מס' הנוירונים בשכבות ה – LSTM 32, 16 בהתאמה



איור 29 : עקומת הלמידה, מס' הנוירונים בשכבות ה – LSTM 128, 64 בהתאמה



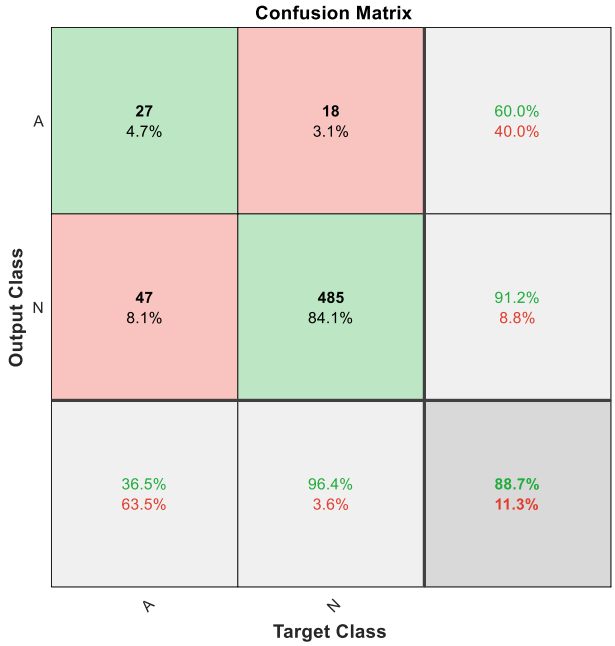
איור 30 : מטריצת המבוכה, מס' הנוירונים בשכבות ה – LSTM 128, 64 בהתאמה

ראשית, ננתח את שלושת המודלים הראשונים. נשים לב כי פונקציית הדיוק של שלושתם לא התכנסה לפני סיום הלמידה, אך קצב השיפור אכן הולך וקטן, כאשר פונקציית ה – loss של שלושתם לא התכנסה בסיום הלימוד. אם כך, ייתכן כי שימוש ב epochs נוספים היה מוביל למודלים טובים יותר. שנית, כפי שציפינו, עד נקודה מסוימת כאשר הגדלנו את כמות הנוירונים המודל נהיה מורכב יותר ובעל הכללה טובה יותר – הדיוק אומנם קטן עם הגדלת כמות הנוירונים, אך ההכללה השתפרה במובן שאחוז ה – AF שזוהו מכלל ה - AF גדל : 9.5% עבור המודל הראשון, 24.3% עבור המודל השני ו – 25.7% עבור המודל השלישי. המודל הרביעי הוביל לתוצאות זהות למודל המקורי, וזאת כיוון שהחל מכמות נוירונים מסוימת, תחת שאר פרמטרי המערכת, המודל יוצר overfitting גדול מדי לדאטה הלמידה ולכן המודל בעל הכללה גרועה. לדעתנו, כדי לייצר מודל יותר טוב דרוש שימוש במס' נוירונים קטן יותר מהמודל המקורי (בסביבה של 16-32), ודרוש להגדיל את כמות ה – epochs עד הנקודה בה פונקציית הדיוק מתכנסת.

3.11

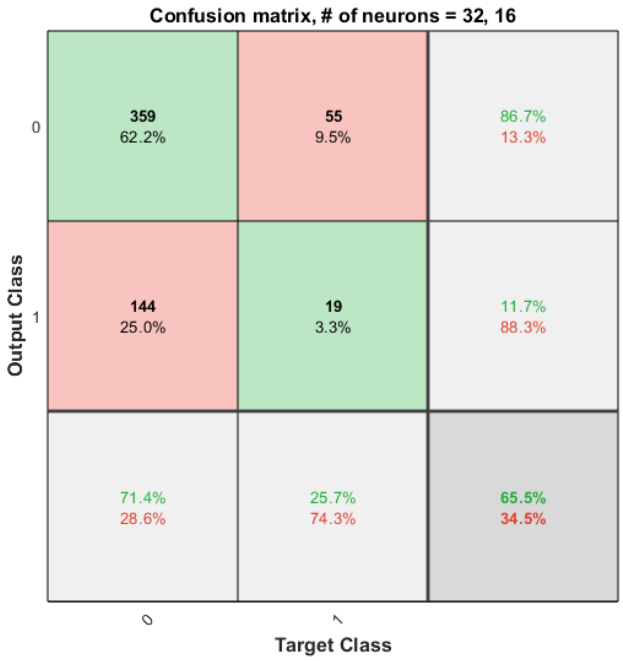
התוצאות המוצלחות ביותר של שתי שיטות הסיווג מבין המודלים שנבנו הינן :

עבור מסווג fine tree :



איור 31 : תוצאות הסיווג הטובות ביותר באמצעות מודל fine tree

תוצאות הסיווג הטובות ביותר באמצעות רשת הנוירונים :

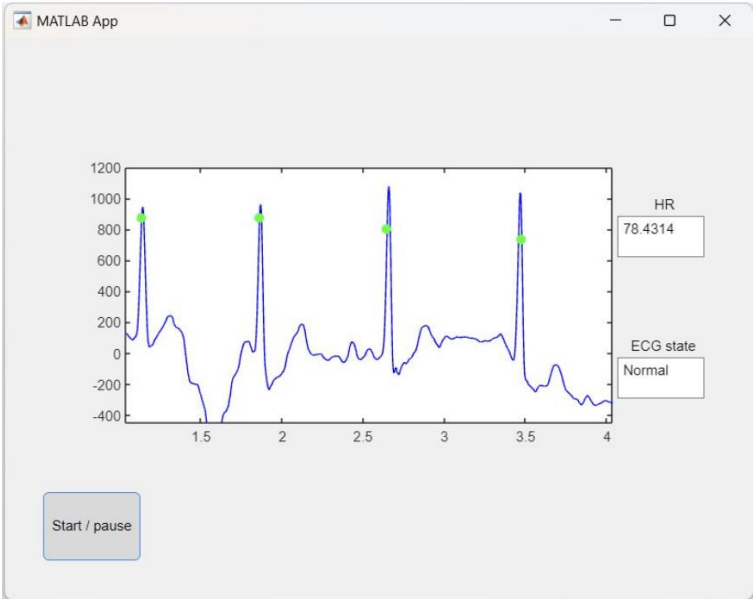


איור 32 : תוצאות הסיווג הטובות ביותר באמצעות רשת נוירונים

ניתן לראות כי תוצאות הסיווג באמצעות מודל ה – fine tree טובות משל רשת הנוירונים. מסווג רשת נוירונים מכיל מס׳ רב מאוד של פרמטרים, אשר יחסים שונים ביניהם משפיעים משמעותית על התוצאות. מצד אחד, ברשת נוירונים הרשת עצמה יוצרת פיצ׳רים מורכבים, דבר המפשט את התהליך, אך כדי להשתמש באופן אפקטיבי ברשת צריך הבנה טובה של השכבות השונות ואיך בחירת הפרמטרים השונים משפיעה על המערכת. במסווגים בהם השתמשנו בניסוי 2, אנו יצרנו את הפיצ׳רים בהם השתמשנו. מצד אחד יצירת הפיצ׳רים דרשה הבנה מסוימת של האות כדי לייצר אותם, אך מנגד הבחירה המושכלת של הפיצ׳רים היא זו שאפשרה יצירת מודל כה מוצלח. לדעתנו קשה להכריע חד משמעית איזה שיטה עדיפה היות ואין לנו היכרות עמוקה מספיק עם רשתות נוירונים, אך ביחס להבנה שלנו באותות ECG, היכולת להשתמש בידע זה אפשרה יצירת מודל סיווג מוצלח יותר משמעותית ולכן בהינתן הידע הרלוונטי שברשותנו לשם משימה זו, לדעתנו סיווג באמצעות מסווג מתאים יותר מרשת נוירונים.

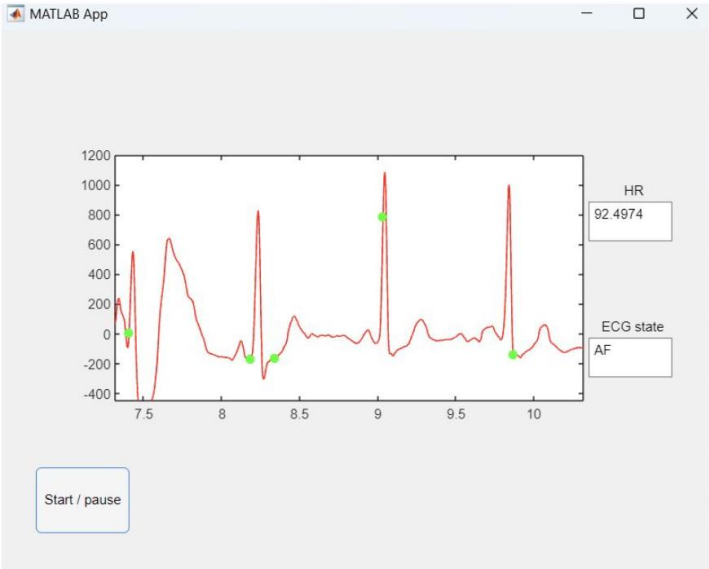
ניסוי 4: הצגה של אות ECG בעזרת ממשק GUI

בממשק ה – GUI לאחר לחיצת כפתור ה – start יופיעו אחד משני מצבים:



איור 33: ECG מצב תקין

כאשר ה – ECG תקין הגרף יופיע בכחול וב – ECG state יהיה כתוב – 'Normal'. זיהויי QRS מופיעים בעיגולים ירוקים.



איור 34: ECG במצב AF

כאשר יזוהה AF, הגרף יופיע באדום וב – ECG state יהיה כתוב – 'AF'.

מסקנות כלליות

במעבדה זו הכרנו אותות ECG, הרעשים המופיעים בהקלטת אותות אלה ואיך לסנן אותם. כמו כן, הכרנו שיטות שונות לסיווג בין אות ECG תקין לאות ECG עם AF. ראשית, ביצענו עיבוד מקדים לאות. לאחר מכן הוספנו רעש רשת מסונתז ו – EMG מסונתז וסיננו אותם באמצעות המסננים המתאימים. ראינו כי לשם סינון אות באופליין עדיף להשתמש בפקודת filtfilt המשתמשת בפונקציית התמסורת $H(w)$, בצורה $|H(w)|^2$ וכך נמנעים מעיוותי פאזה של התמסורת. בנוסף, במעבדה זו למדנו מהי פתולוגיה של פרפור עליות (AF) ובנינו מודל מכונת למידה, שלאחר ביצוע סינון מקדים יבצע קלסיפיקציה לפי פיצ׳רים אותם יצרנו. המודל הטוב ביותר שהצלחנו ליצור לא ביצע הרבה שגיאות כאשר זיהה *AF (80%)*, אך לא זיהה מכלל ה – *AF* אחוז גבוה של ה – *AF (36.5%)*. נשער כי תוצאות אלה נובעות מחוסר איזון של הדאטה כיוון שבדאטה הלמידה רק 13.3% מהדגימות הינן של *AF*. לאחר קלסיפיקציה באמצעות למידת מכונה, ביצענו קלסיפיקציה באמצעות למידה עמוקה. במודל זה, לעומת מודלים של למידת מכונה, המידע מוזן אל המודל, והמודל עצמו מייצר פיצ׳רים מהמידע, ומשפר את המשקולות של הפיצ׳רים ע״י למידת הדגימות. במודלים של למידת מכונה בהם מייצרים פיצ׳רים בצורה ידנית, דבר זה יכול לתת יתרון במידה וישנה הבנה עמוקה מספיק של המצבים ביניהם אנו מנסים להבדיל בקלסיפיקציה – כלומר במקרה זה הבנה של אותות ה - *ECG* והשפעת מצב *AF* על מאפייני ה – *ECG* אפשרה יצור מודל מוצלח יותר מאשר המודל שהתקבל באמצעות רשת הנוירונים. מצד שני, אין לנו הכרה עמוקה של רשתות נוירונים, בהם אין צורך בהבנה עמוקה של המצבים ביניהם מנסים להפריד, אך דרושה הבנה של השכבות השונות בהם משתמשים, הסדר והכמות של השכבות ואיך בחירה של ההיפר-פרמטרים שלהם משפיעה על המודל. עוד נאמר כי ראינו כי חוסר איזון בדאטה משפיע בצורה משמעותית על יכולות הזיהוי של שני המודלים. גם לאחר ניסיון להתמודד עם חיסור איזון זה באופן אקטיבי בניסוי 3, לא מצאנו שיפור משמעותי בתוצאות. לבסוף יצרנו *GUI* המציג ניתוח ״אונליין״ של אות אק״ג. בממשק השתמשנו במודל הקלסיפיקציה של *fine-tree* לסיווג האק״ג. ראינו כי התוצאות אינן טובות במיוחד, וייתכן שזאת גם כיוון שהמודל המקורי לא מזהה *AF* בצורה טובה מספיק וגם כי הוא מאומן על אותות ארוכים בהרבה מאלה שמסופקים באונליין.

מקורות

[1] "Electrocardiography," *Wikipedia*. Dec. 23, 2022. Accessed: Jan. 09, 2023. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Electrocardiography&oldid=1128987191

[2] F. Morris, W. Brady, and A. J. Camm, *ABC of clinical electrocardiography*, 2nd ed. Malden, Mass. ; Blackwell Pub., 2008.

[3] J. G. Webster, "Reducing Motion Artifacts and Interference in Biopotential Recording," *IEEE Trans. Biomed. Eng.*, vol. BME-31, no. 12, pp. 823–826, 1984, doi: 10.1109/TBME.1984.325244.

[4] K. Rahul, *Signal Processing Techniques for Removing Noise from ECG Signals. J Biomed Eng 1: 1-9*. 2019.

[5] L. Sörnmo and P. Laguna, Eds., "Copyright," in *Bioelectrical Signal Processing in Cardiac and Neurological Applications*, Burlington: Academic Press, 2005, p. iv. doi: 10.1016/B978-0-12-437552-9.50013-1.

[6] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*, First edition. Sebastopol, California: O'Reilly Media, 2017.

[7] Chih-Wei Hsu and Chih-Jen Lin, "A comparison of methods for multiclass support vector machines," *IEEE Trans. Neural Netw.*, vol. 13, no. 2, pp. 415–425, Mar. 2002, doi: 10.1109/72.991427.

[8] "Deep learning | Nature." https://www.nature.com/articles/nature14539 (accessed Jan. 09, 2023).

[9] EliteDataScience, "Overfitting in Machine Learning: What It Is and How to Prevent It," *EliteDataScience*, Sep. 07, 2017. https://elitedatascience.com/overfitting-in-machine-learning (accessed Jan. 09, 2023).

[10] J. Brownlee, "8 Tactics to Combat Imbalanced Classes in Your Machine Learning Dataset," *MachineLearningMastery.com*, Aug. 18, 2015. https://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/ (accessed Jan. 09, 2023).

[11] "Bioelectromagnetism." https://www.kenrico.com/media/bembook/ (accessed Jan. 09, 2023).

[12] K. Saitoh, *Deep Learning from the Basics: Python and Deep Learning: Theory and Implementation*. Packt Publishing Ltd, 2021.

[13] "Papers with Code - Focal Loss for Dense Object Detection." https://paperswithcode.com/paper/focal-loss-for-dense-object-detection (accessed Jan. 09, 2023).

נספחים

```
%% 1:
ECG = readtable('ECG_scope.csv');
volt = ECG.Voltage;
time = ECG.Time;
volt=volt-mean(volt); %dc=0
%1.2:
index=min(find(time>=5));
figure; plot(time(1:index),volt(1:index));xlabel('Time [sec]'); xlim([0
5]);

%1.3:
signalPower = sum((volt).^2);

%network noise:
f=sin(2*pi*50.*time);
fPower = sum((f).^2);
a=sqrt(signalPower*10^-3/fPower);
network_noise=a.*f;
%check:
S=snr(volt,network_noise)

ECG_noise1=volt+network_noise;

% EMG noise:
L=length(volt);
EMG_noise =normrnd(0,1,[L,1]); %EMG is from a normal distribution with mu=0
and sigma=1, random in the length of the signal
Fs=L/time(end);
fc1=20; %[Hz]
fc2=500; %[Hz]
[b,a] = butter(10,[fc1/(Fs/2) fc2/(Fs/2)],'bandpass'); %EMG frequency is
between 20-500 Hz so we do a BPF to the random signal
EMG_noise = filter(b,a,EMG_noise);
fPower = sum((EMG_noise).^2);
a=sqrt(signalPower*10^-3/fPower);
EMG_noise=a.*EMG_noise;
%check:
S=snr(volt,network_noise)

ECG_noise2=volt+EMG_noise;

figure; subplot(3,1,1);plot(time(1:index),volt(1:index));xlabel('Time
[sec]'); xlim([0 5]);title('The original ECG');
subplot(3,1,2);plot(time(1:index),ECG_noise1(1:index));xlabel('Time
[sec]'); xlim([0 5]);title('ECG with network noise');
subplot(3,1,3);plot(time(1:index),ECG_noise2(1:index));xlabel('Time
[sec]'); xlim([0 5]);title('ECG with EMG noise');

%1.4
%network noise
w0=2*pi*50/Fs; r=0.9;
b=[1 -2*cos(w0) 1];
a=[1 -2*r*cos(w0) r^2];
ECG1_withoutnoise=filter(b,a,ECG_noise1);
%EMG
w1=2*pi*20/Fs;
b=fir1(500,w1,'low');
```

```
ECG2_withoutnoise=filter(b,1,ECG_noise2);

figure; subplot(3,1,1);plot(time(1:index),volt(1:index));xlabel('Time
[sec]'); xlim([0 5]);title('The original ECG');
subplot(3,1,2);plot(time(1:index),ECG1_withoutnoise(1:index));xlabel('Time
[sec]'); xlim([0 5]);ylim([-0.2,0.8]);title('ECG with network noise after
IIR');
subplot(3,1,3);plot(time(1:index),ECG2_withoutnoise(1:index));xlabel('Time
[sec]'); xlim([0 5]); ylim([-0.2,0.8]);title('ECG with EMG noise after LPF
(20 Hz)');

%1.5
breathnoise=sin(2*pi*0.25.*time);
ECG_breathnoise=volt+breathnoise;

%fir:
w1=2*pi*0.5/Fs;
b=fir1(500,w1,'high');
freqz(b,1)

ECGfir=filter(b,1,ECG_breathnoise);
ECGfir=ECGfir-mean(ECGfir);
%iir:
[b,a] = butter(5,2*0.5/Fs,'high');
freqz(b,a)

ECGfir_filter=filter(b,a,ECG_breathnoise);
ECGfir_filter=ECGfir_filter-mean(ECGfir_filter);

ECGfir_filtfilt=filtfilt(b,a,ECG_breathnoise);
ECGfir_filtfilt=ECGfir_filtfilt-mean(ECGfir_filtfilt);

index=min(find(time>=10));
figure; subplot(2,1,1);plot(time(1:index),volt(1:index));xlabel('Time
[sec]'); xlim([0 10]);ylim([-1.5 2]);title('The original ECG');
subplot(2,1,2);plot(time(1:index),ECG_breathnoise(1:index));xlabel('Time
[sec]'); xlim([0 10]);ylim([-1.5 2]);title('ECG with breath noise');

figure;subplot(3,1,1);plot(time(1:index),ECGfir(1:index));xlabel('Time
[sec]');  xlim([0 10]); title('ECG with breath noise after FIR (using
filter)');
subplot(3,1,2);plot(time(1:index),ECGfir_filter(1:index));xlabel('Time
[sec]');  xlim([0 10]);title('ECG with breath noise after IIR (using
filter)');
subplot(3,1,3);plot(time(1:index),ECGfir_filtfilt(1:index));xlabel('Time
[sec]');  xlim([0 10]);title('ECG with breath noise after IIR (using
filtfilt)');

%% 2:

%2.3:
TrainLabels = readtable('Train_labels.xlsx');
Fs=300; %[Hz]


folder_path ='C:\Users\solam\OneDrive\Desktop\ECG data\Train';
files = dir(fullfile(folder_path,'*.mat'));
for i = 1:length(files)
   signal=load(fullfile(folder_path,files(i).name)).val;
```

```
   [signal,delay,t]=Pre_processing(signal,Fs); %Pre processing including
DC, network, breath, etc.
   L=length(signal);
   %number of zero crossing:
   zero_cross(i,1)=ZeroCrossing(signal,Fs)/L;
   %heart rate:
   [~,~,~,~,ind_peak]=PanThomp_QRS(signal,Fs);
   timeRR=t(ind_peak);
   HR(i,1)=60/mean(diff(timeRR));
   %SDNN:
   r=diff(timeRR);
   SDNN(i,1)=sqrt(sum((r-mean(r)).^2)/(length(r)-1));
   %rMSSD:
   rmssd(i,1)=rMSSD(timeRR);
   %HAPO:
   Hapo(i,1)=HaPo(signal,Fs);
   %spectral centroid:
   centroid(i,1)=Spectral_Centroid(signal,Fs);
   %MDF:
   median_freq(i,1)=MDF(signal,Fs);
end
Labels=TrainLabels.Label;
Train_features=table(zero_cross,HR,SDNN,rmssd,Hapo,centroid,median_freq,Lab
els);
writetable(Train_features,'Train_features.xlsx');

clear all;
TestLabels = readtable('Test_labels.xlsx');
Fs=300; %[Hz]

folder_path ='C:\Users\solam\OneDrive\Desktop\ECG data\Test';
files = dir(fullfile(folder_path,'*.mat'));
for i = 1:length(files)
   signal=load(fullfile(folder_path,files(i).name)).val;
   [signal,delay,t]=Pre_processing(signal,Fs); %Pre processing including
DC, network, breath, etc.
   L=length(signal);
   %number of zero crossing:
   zero_cross(i,1)=ZeroCrossing(signal,Fs)/L;
   %heart rate:
   [~,Newsignal,~,~,ind_peak]=PanThomp_QRS(signal,Fs);
   Newt=(0:1:length(Newsignal)-1)/Fs; %[sec]
   timeRR=Newt(ind_peak);
   HR(i,1)=60/mean(diff(timeRR));
   %SDNN:
   r=diff(timeRR);
   SDNN(i,1)=sqrt(sum((r-mean(r)).^2)/(length(r)-1));
   %rMSSD:
   rmssd(i,1)=rMSSD(timeRR);
   %HAPO:
   Hapo(i,1)=HaPo(signal,Fs);
   %spectral centroid:
   centroid(i,1)=Spectral_Centroid(signal,Fs);
   %MDF:
   median_freq(i,1)=MDF(signal,Fs);
end
Labels=TestLabels.Label;
Test_features=table(zero_cross,HR,SDNN,rmssd,Hapo,centroid,median_freq);
writetable(Test_features,'Test_features.xlsx');
```

```
%
Train_features=readtable('Train_features.xlsx');
Test_features=readtable('Test_features.xlsx');

%2.11:
test_pred_finetree =FineTree.predictFcn(Test_features);
test_pred_finetree=categorical(test_pred_finetree);
test_pred_naivebayes=
NaiveBayes.predictFcn(Test_features);test_pred_naivebayes
=categorical(test_pred_naivebayes);
test_pred_svm
=SVM.predictFcn(Test_features);test_pred_svm=categorical(test_pred_svm);
true_labels=categorical(Labels);


plotconfusion(true_labels,test_pred_finetree)
plotconfusion(true_labels,test_pred_naivebayes)
plotconfusion(true_labels,test_pred_svm)

%2.12
Train_features=Train_features(:,1:7);
train_pred_finetree =FineTree.predictFcn(Train_features);
train_pred_finetree=categorical(train_pred_finetree);
train_pred_naivebayes=
NaiveBayes.predictFcn(Train_features);train_pred_naivebayes
=categorical(train_pred_naivebayes);
train_pred_svm
=SVM.predictFcn(Train_features);train_pred_svm=categorical(train_pred_svm);
TrainLabels = readtable('Train_labels.xlsx');
true_labels=categorical(TrainLabels.Label);


plotconfusion(true_labels,train_pred_finetree)
plotconfusion(true_labels,train_pred_naivebayes)
plotconfusion(true_labels,train_pred_svm)

%% 3.
clear, clc

% 3.4
% run the first network
run('DLModel.mlx')

% exported the training progress as a figure neamed
% 'Training_prog_no_weights.png'
figure
imshow('Training_prog_no_weights.png')

% 3.6
% running the second model
clear, clc
run('ModelWithWeights.mlx')

% 3.9
% running the model with over sampled AF data to balanace the data:
clear, clc
run('re_sample_AF_model.mlx')
figure, imshow('Training_prog_resample.png')
```

```
%3.10
% we will observe how changing the amount of neurons in the LSTM layers
% affect the model:
clear, clc
run('neuron_size_model.mlx')

%plotting the results:
lstm1 = [8, 16, 32, 128]; % hidden units- first LSTM layer
lstm2 = [8, 16, 16, 64];% hidden units- second LSTM layer

for i=1:4
    figure, imshow(['LSTM_prog', num2str(i), '.png']) %training progress
plot
    title(['Training Progress, # of neurons = ', num2str(lstm1(i)), ', ',
num2str(lstm2(i))])
    YPred = classify(nets{i},Data_test,'MiniBatchSize',BatchSize);%
Classify test data for confusion matrix
    figure, plotconfusion(Test_labels,YPred);% Confusion matrix plot
    title(['Confusion matrix, # of neurons = ', num2str(lstm1(i)), ', ',
num2str(lstm2(i))])
end


%% 4.
% first change the directory to the Test file, assuming the Test file is
% inside the current directory
current = cd;
cd Test;

% get the list of names of the mat files
file_names = dir;

ECG = []; %initialization of ECG data vector
% iterate on the files concatinating them one after another
for i=3:length(file_names)
    test_data = load(file_names(i).name); %loading the data of specific
sample
    ECG = [ECG, test_data.val]; % concatinating the samples
end

% return to the former directory
cd(current)

%loading the classification model to workspace:
importdata('finetree.mat');
% now run the GUI
```