דו"ח עבודה 2 -למידה עמוקה

שאלה 1 –

סעיף a –

סט הנתונים שנשתמש בו בעבודה זו נקרא PAMAP2 שלמעשה מאגד את הנתונים הפיזיים של 9 נבדקים(אישה אחת ו-8 גברים) אשר ביצעו פעילויות (למשל הליכה, שכיבה, ריצה, אופניים וכו') תוך כדי שמודדים להם קצב פעימות לב, טמפרטורה ומדדי אינרציה ביד, חזה וקרסול.

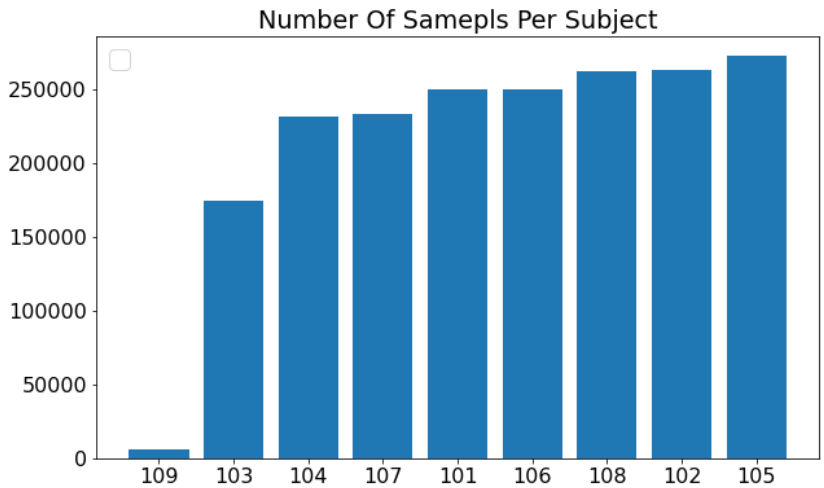
Each of the data-files contains 54 columns per row, the columns contain the following data:

- 1 timestamp (s)
- 2 activityID (see II.2. for the mapping to the activities)
- 3 heart rate (bpm)
- 4-20 IMU hand
- 21-37 IMU chest
- 38-54 IMU ankle

– Benchmark results on dataset

	Standard 9-fold cross-validation				LOSO 9-fold cross-validation			
Classifier	Precision	Recall	F-measure	Accuracy	Precision	Recall	F-measure	Accuracy
C4.5 decision tree	0.9968	0.9968	0.9968	0.9970	0.9349	0.9454	0.9401	0.9447
Boosted C4.5	0.9997	0.9994	0.9995	0.9995	0.9764	0.9825	0.9794	0.9785
Bagging C4.5	0.9971	0.9968	0.9970	0.9971	0.9346	0.9439	0.9392	0.9433
Naive Bayes	0.9899	0.9943	0.9921	0.9923	0.9670	0.9737	0.9703	0.9705
kNN	1.0000	1.0000	1.0000	1.0000	0.9955	0.9922	0.9938	0.9932

נלקח מ- https://www.researchgate.net/figure/7-Benchmark-on-the-PAMAP2-dataset-performance-measures-on-the-Basic-activity_tbl6_260291508



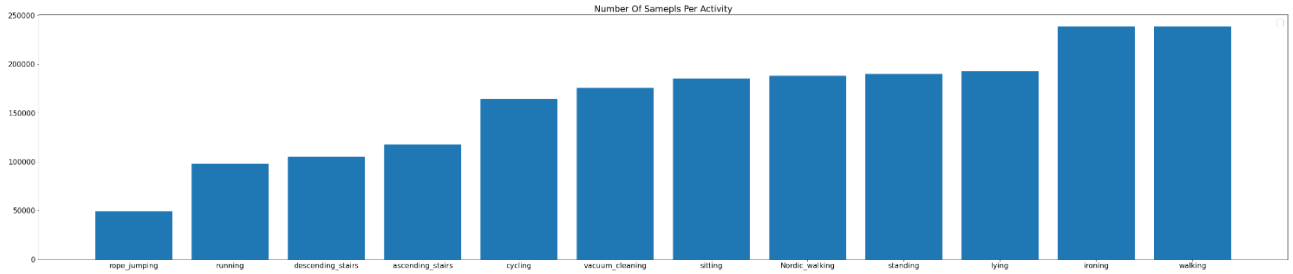
הנתונים דיי מאוזנים, רק נבדק 109 עם מס' נמוך משמעותית של דגימות

בשונה ממאגרי נתונים קודמים שעבדנו איתם, במאגר זה יש נתונים אשר מסתמכים על זמן וסנסורים, לכן כפי שנראה יהיו המון נתונים לא ולידיים ונצטרך לנקות אותם ע"י מחיקת רשומות שהפעילות בהן היא '0' כיוון שתיוג זה מייצג פעילות שלא תורמת לנו לבניית המודל. כמו כן נצטרך להשלים נתונים ע"י מיצוע של מדדים מסוימים כדי להחליף ערכי Nan – נעשה זאת ע"י אינטרפולציה של הנתונים הקיימים.

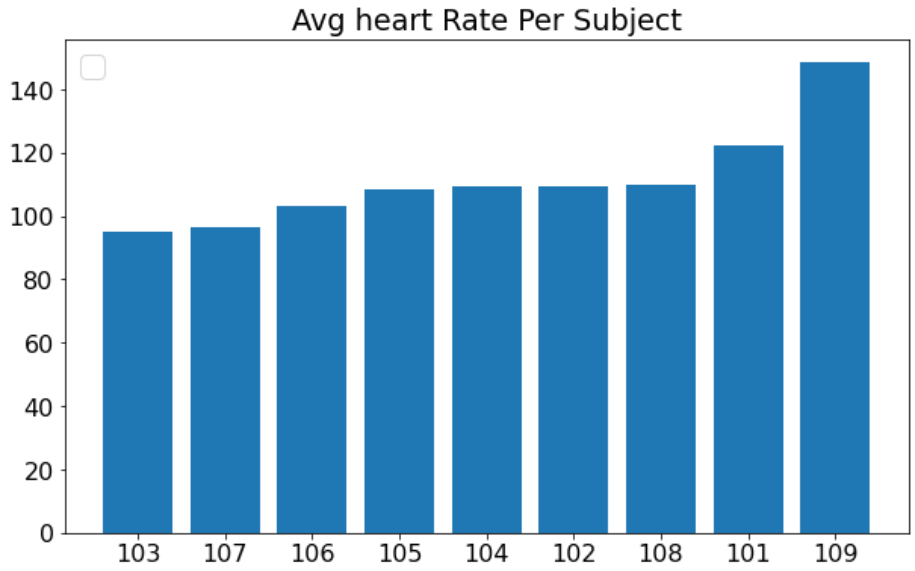
המודל יקבל סט נקודות בזמן אשר לכל אחת מהן יש את כל המדדים שנמדדו בזמן מסוים, כאשר כל הנקודות מתויגות עם פעילות מסוימת שקורית בזמן הזה.

סעיף b –

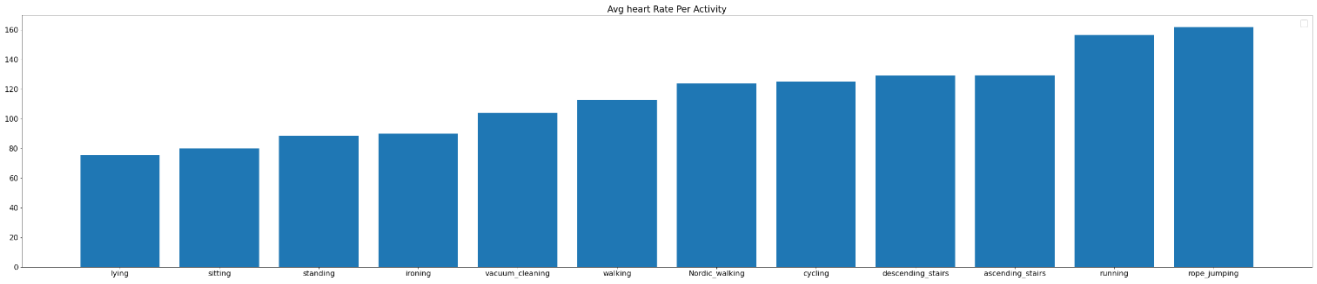
הבעיה שלנו היא בעיית קלסיפיקציה - המשימה שלנו היא לזהות עבור נקודה בזמן עם סט מדדים מסוימים איזו פעילות מתבצעת בנקודת זמן זו בהתבסס על סט נקודות זמן קודמות אשר מתארות פעילויות שונות ומדדים שונים. אנו למעשה חוזים עבור מדדי נבדק מסוים את הפעולה שהוא נוקט בנקודת זמן הבאה.



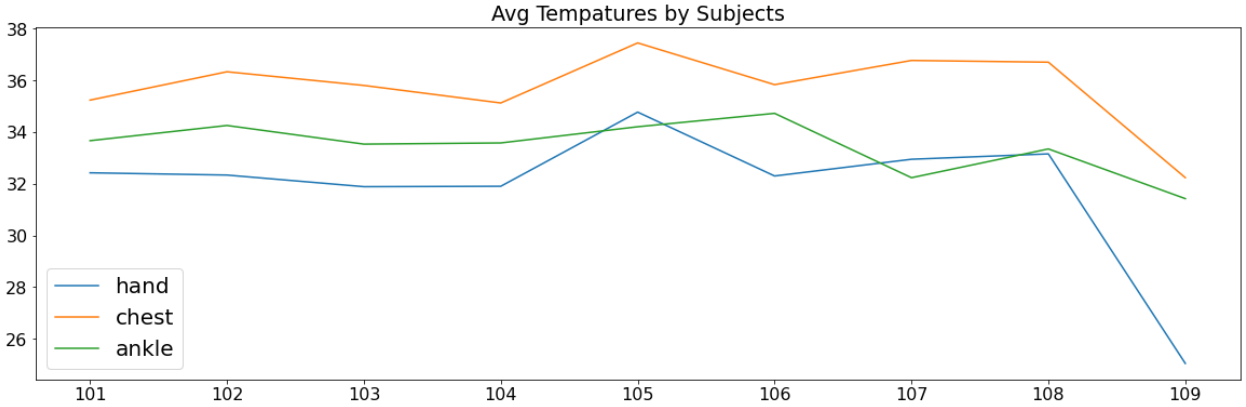
ישנה כמות נמוכה יחסית של דגימות של פעילות "קפיצה בחבל" וכמויות גבוהות של דגימות עבור "הליכה" ו-"גיהוץ". שאר הדאטה יחסית מאוזן מבחינה זו.



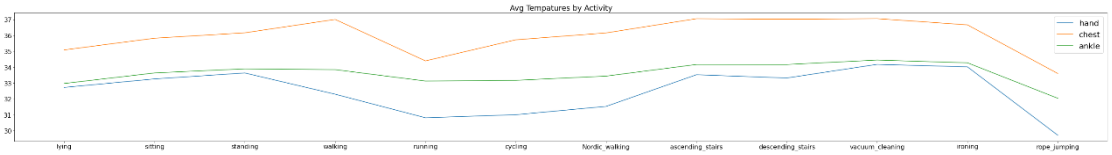
ממוצע פעימות לב לנבדק בפעילויות השונות שהוא מבצע – ניתן לראות שהממוצע דיי דומה אצל כל הנבדקים, אפשר לחשוב שנבדק 109 עם ממוצע גבוה בשל כמות דגימות נמוכה של פעילות מסוימת עם דופק גבוה.



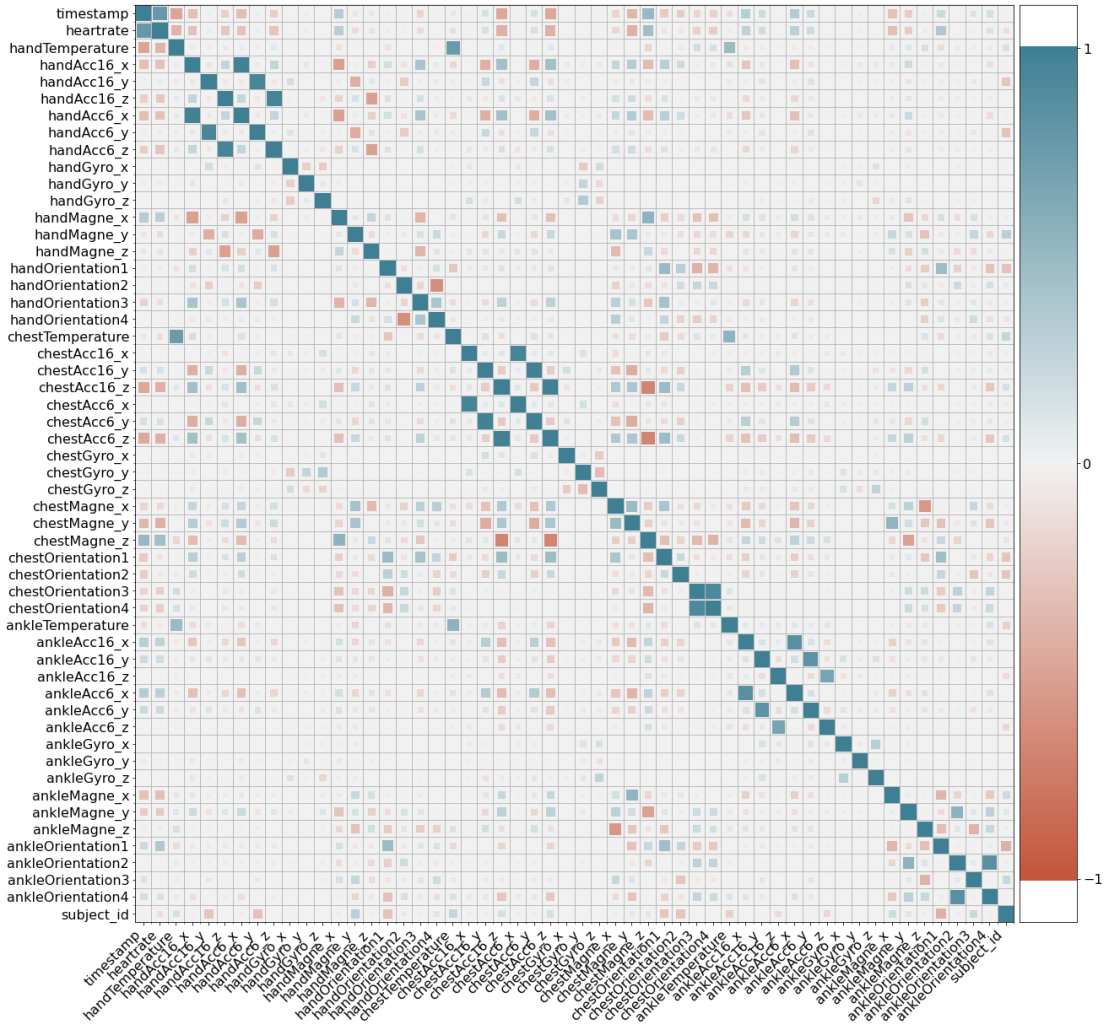
ממוצע פעימות לב לפי סוג הפעילות – פעולות נייחות יותר כמו שכיבה, עבודה משרדית וישיבה עם ממוצע נמוך יותר לעומת ריצה, קפיצה בחבל אשר דורשות יותר מאמץ ועל כן הממוצע בהן גבוה יותר.



טמפ׳ ממוצעת בכל אחד מחלקי הגוף הנמדדים עבור כל נבדק, נבדק 109 עם ממוצעים נמוכים יותר מהשאר.



קפיצה בחבל באופן מפתיע בעלת טמפ׳ הנמוכות מאוד באברי הגוף באופן מפתיע ע״פ הגרף, אך דבר זה יכול לנבוע מהעובדה שיש מעט מאוד רשומות המתארות פעולה זו.



ע״פ טבלת הקורלציה שיצרנו עבור העמודות בטבלה שלנו ניתן להבחין כי העמודות של חותמת הזמן וקצב הלב קשורות אחת לשנייה, כמו כן עבור כל סנסור יש קורלציה גבוהה בין צירי הx,y,z של אותו חלק בגוף עבור הסנסור. ישנם מדדים אשר לא תורמים במיוחד כמו מדד הג׳יירוסקופ אשר לא יוצר קורלציה גבוהה עם מדדים אחרים כך שאולי עדיף שלא להשתמש בו בבניית המודל. מדד הטמפ׳ בחלקי הגוף שנמדדים בעל קורלציה יחסית גבוהה גם כן – הגוף כולו מתחמם בפעילות ספורטיבית, חלק מהאיברים יותר וחלק פחות אך עדיין קיים הקשר הזה.

סעיף c –

Self-supervised – שימוש בנתונים **קיימים** (ולא בתיוג חיצוני כלשהו) בתור תיוג לאימון של משימה שהיא לאו דווקא המשימה שאנחנו רוצים לפתור, שיטה זו מצריכה הרבה דוגמאות כדי לבצע למידה של הנתונים ברצף. למעשה נרצה להשתמש בחלונות בגודל מסוים של נתונים לפני ואחרי כדי לחזות את הערך הנוכחי, ככל שהחלונות שבעזרתם נתאמן יהיו גדולים יותר כך יהיה לנו סיכוי טוב יותר לדייק (למידה ממושכת יותר קרוב לערך שאנחנו מחפשים), כמו כן כאשר החלון הנוכחי גדול יותר כך יהיה לנו קשה יותר לחזות (ולהפך – ככל שהוא קטן יותר כך יהיה לנו יותר קל לחזות).

נרצה ללמוד מאפיינים מסוימים של הנתונים באמצעות הנתונים עצמם ולא בעזרת תיוג חיצוני. מבצעים הסתרה של חלק מהמידע הזמין לנו(ומשתמשים בו בתור לייבל) ע״מ ללמוד קשרים בין הנתונים שלנו.

ניתן לבצע self-supervised גם מנתונים שאין להם תיוג למשימת הקלסיפיקציה, למשל עבור נבדקים 107,108 אפשר לבצע self-supervised כי אנחנו נשתמש בנתונים הקיימים שלהם ולא בנתוני התיוג החסרים.

בעצם כאשר אנו מבצעים משימת self-supervised אנו מאמנים את הרשת שלנו להגיע למשקולות מתאימות ויכולים להשתמש בהן למשימת הקלסיפיקציה.

משימות אפשריות:

- חיזוי קצב פעימות לב בנקודת זמן בהינתן חלונות נקודות זמן של לפני ואחרי הנקודה שאנחנו רוצים לחזות.
- חיזוי ערך מדד הסנסור בציר ה-X של היד בנקודת זמן בהינתן חלונות נקודות זמן של לפני ואחרי הנקודה שאנחנו רוצים לחזות.
- חיזוי קצב פעימות לב בנקודת זמן בהינתן חלונות נקודות זמן של לפני ואחרי של מדדים אחרים ללא מדד קצב פעימות הלב לפני ואחרי.
-

שאלה 2 –

סעיף a – שיטת הולידציה שנבצע למודל זה תהיה כך שניקח את נבדקים 101-105 כולל ונבדק 109 עבור סט האימון(נבדק 109 עם מעט דגימות לכן לא נוכל לשים אותו בסט הולידציה) ובסט הולידיציה נשים את נבדק 106 כאשר בסט האימון נשתמש בנתונים של נבדקים 107-108 .

סעיף b – בסעיף זה השתמשנו בבייסליין נאיבי אשר הולך לפי שני מדדים שחשבנו שברור שעבורם ישתנו התוצאות וגם ע״ב הגרפים בשאלה 1 ניתן לראות שמדד פעימות לב משתנה בין פעילויות ומד הטמפ׳ ביד משתנה יחסית בין פעילויות שונות, יצרנו את הטבלה הבאה כדי לקבל את הערכים שלפיהם נבצע את ההחלטה :

act_index	activity	heartrate	handTemp
1	lying	75.540579	32.726154
2	sitting	80.012614	33.262088
3	standing	88.554771	33.637791
17	ironing	90.062321	34.022834
16	vacuum_cleaning	104.194956	34.178360
4	walking	112.786174	32.300379
7	Nordic_walking	123.829853	31.534647
6	cycling	124.884246	31.008824
13	descending_stairs	129.156243	33.322055
12	ascending_stairs	129.525475	33.527258
5	running	156.590944	30.818058
24	rope_jumping	161.985048	29.720052

יצרנו פונקציה אשר מקבל רשומה ומחליטה ע״ב טווחים של פעימות לב וטמפ׳ יד לאיזה פעילות הרשומה הזו שייכת.

פעימות לב			95				118				140				
טמפ' יד		33		34		33		32.3		30.2					
פעילות	1	2,3		17		4		16	6,7		12,13		24		5

למשל רשומה שבה הדופק הוא מתחת ל-95 וטמפ׳ יד בין 33-34 תקבל סיווג רנדומלי בין פעילות 2 או 3.

עבור סט האימון קיבלנו דיוק של 0.319

עבור סט הולידציה קיבלנו דיוק של 0.366

עבור סט המבחן קיבלנו דיוק של 0.327

סעיף c -

עבור סעיף זה התבקשנו ליצור solid benchmark בעזרת מודלי ML ידועים.

בחרנו להשתמש בפיצ'רים הבאים שנראו לנו הרלוונטיים ביותר בגלל הקורלציה הגבוהה ביניהם:

```
'timestamp', 'activityID','heartrate','subject_id','handTemperatu
re', 'handAcc16_x', 'handAcc16_y', 'handAcc16_z', 'handAcc6_x', '
handAcc6_y', 'handAcc6_z', 'chestTemperature',   'chestAcc16_x',
'chestAcc16_y', 'chestAcc16_z', 'chestAcc6_x', 'chestAcc6_y', 'ch
estAcc6_z', 'ankleTemperature', 'ankleAcc16_x', 'ankleAcc16_y', '
ankleAcc16_z', 'ankleAcc6_x', 'ankleAcc6_y', 'ankleAcc6_z'
```

יצרנו דאטה חדש המורכב מהפיצ'רים הנ"ל אשר בנוי מחלונות זמן, השתמשנו ב-2 שניות חלון זמן אחורה (למעשה 200 רשומות) ביצענו להם ממוצע וניסינו לחזות את הרשומה חצי שנייה לאחר מכן(50 רשומות אחרי). כמו כן, במהלך ניקוי הנתונים הסרנו רשומות עם פעילות '0' אשר לא תורמת לביצוע הסיווג והשלמנו תאים בטבלה עם ערך Nan בעזרת פונקציית אינטרפולציה.

בחרנו לעבוד עם מודל Logistic Regression ומודל Random Forest.

עבור סט ולידציה קיבלנו:

0.575 – RF , 0.585 – LR

עבור סט המבחן קיבלנו:

0.565 – RF , 0.585 – LR

סעיף d –

יצרנו דאטה המתבסס על הדרך בסעיף הקודם רק שכאן לא ביצענו ממוצע של החלון זמן שעליו הסתכלנו, אלא שמנו את הרשומות כמו שהן. בנוסף, עבור רשומות האימון הסרנו את עמודת החיזוי activityID והמרנו את הערכים לערכים בין 0-1 בעזרת MinMaxScaler ע"מ שהמודל ירוץ מהר יותר ועל ערכים קטנים. עבור רשומות החיזוי רצינו לתת קטגוריה לכל פעילות אז הפכנו את המספר קטגוריה לערך וקטורי המיוצג ע"י one hot vector .

בנינו את המודל הבא:

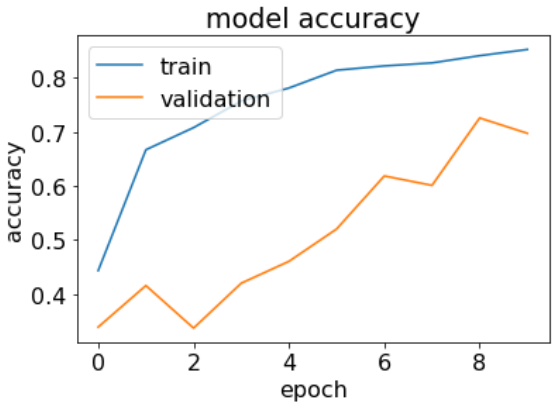
Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 200, 24)	4704
dense (Dense)	(None, 200, 24)	600
lstm_1 (LSTM)	(None, 12)	1776
dense_1 (Dense)	(None, 12)	156
dense_2 (Dense)	(None, 12)	156

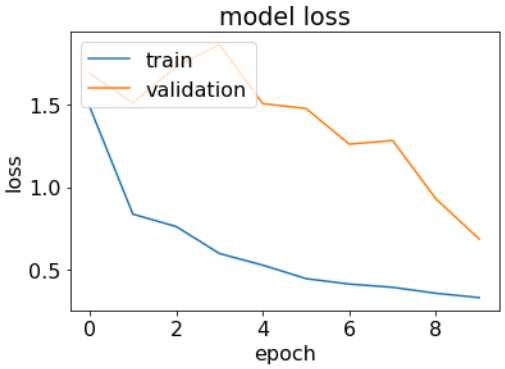
```
Total params: 7,392
Trainable params: 7,392
Non-trainable params: 0
```

בשכבה האחרונה של המודל השתמשנו בפונק' אקטיבציה של softmax כיוון שזוהי משימת סיווג.

פונקציית ההפסד של המודל היא categorial_crossentropy המתאימה למשימת סיווג – נרצה לסווג את הפעילות המתבצעת – הפעילויות הן קטגוריות.

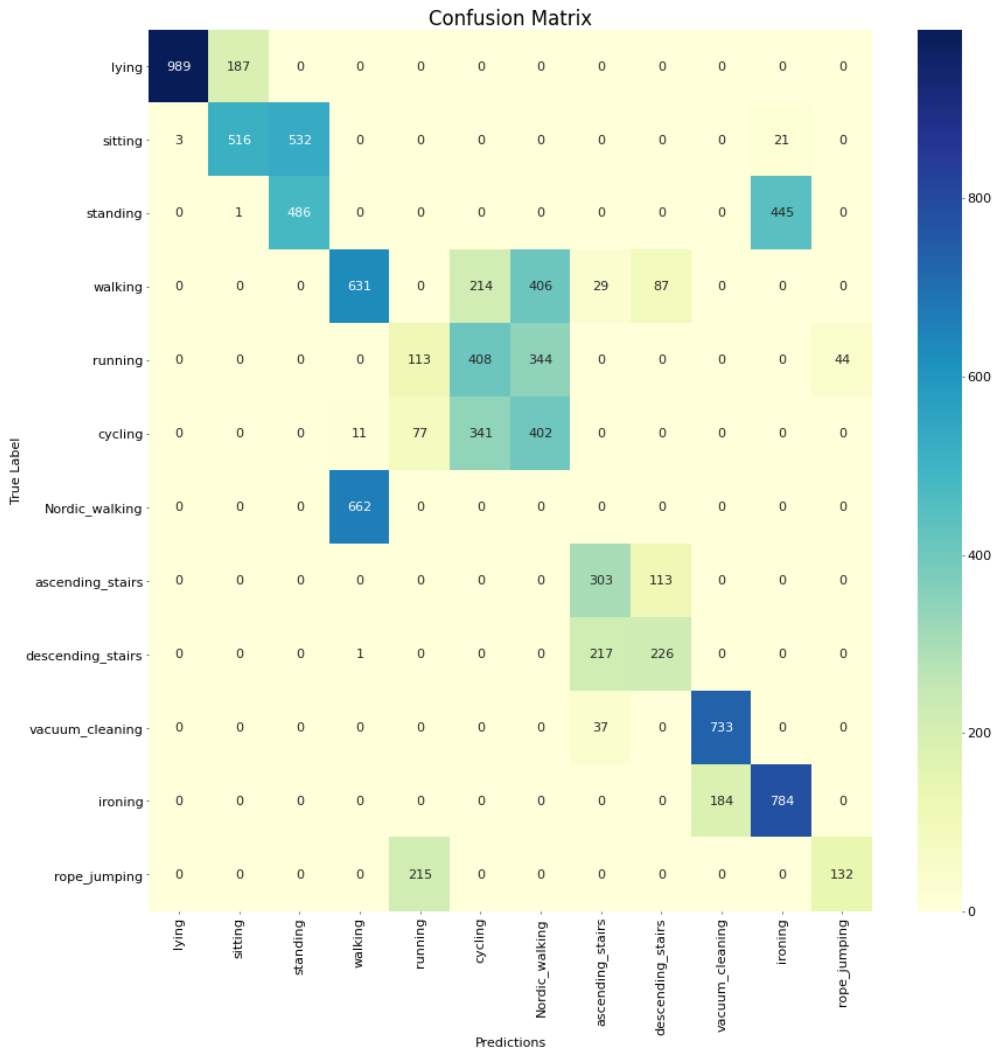
תוצאות :






```
model_activity.evaluate(x_test, y_test_ohe)
```

```
310/310 [==============================] - 4s 9ms/step - loss: 1.6452 - accuracy: 0.4944
[1.3603885173797607, 0.5310289263725281]
```



שגיאות בסיווג אפשריות במקרה שבו המודל לא מורכב כ״כ או לא מספיק מאומן(עשינו רק 10 אפוקים). ניתן לראות במטריצה את הפעילויות שזוהו כדומות כמו למשל ריצה ואופניים.

אפשר לראות גם שהאימון שלנו לא מצליח כך על הולידיציה אבל כן מצליח לתת תוצאות טובות יותר על סט המבחן אולי בגלל כמות הדגימות וסוג הפעילויות, נזכיר שסט הולידציה מורכב מנבדק 106 בלבד.

סעיף e –

נרצה לאמן את המודל שלנו בשיטת self-supervised למשימה הבאה:

- חיזוי קצב פעימות לב בנקודת זמן בהינתן חלונות נקודות זמן של לפני ואחרי הנקודה שאנחנו רוצים לחזות.

ביצענו הכנה של הנתונים כך שיתאימו למשימה החדשה, כאשר מבנה target מייצג את ה-heartrate שנמדד.

נשתמש למשימה זו במבנה מודל זהה המותאם לערכי הקלט והפלט החדשים. משימה זו הינה משימת רגרסיה שכן אנו חוזים ערך מספרי ולכן נשתמש כאן בפונקציית אקטיבציה relu וכן בפונקצית הפסד mse. נשתמש בלמידה של מודל זה על מנת לבצע feature extraction שיהוו קלט למודל שלנו מהסעיף הקודם.

המודל:

```
model_heartrate.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2 (LSTM)                (None, 64)                23040
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 33
=================================================================
Total params: 25,153
Trainable params: 25,153
Non-trainable params: 0
```

```
model_heartrate.evaluate(x_test_hr, y_test_hr)
```

```
310/310 [==============================] - 2s 5ms/step - loss: 10.8015 - mae: 3.1472 - root_mean_squa
red_error: 3.2866

[10.801493644714355, 3.147228479385376, 3.28656268119812]
```

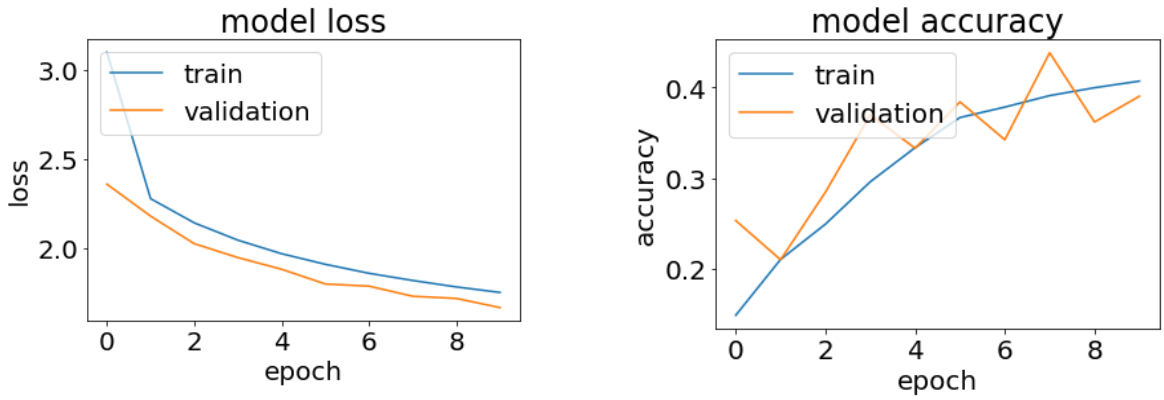
קיבלנו LOSS של MSE השווה ל-10 .

Transfer model – המודל שניצור לאחר שניקח את המשקולות של המודל לעיל:

```
transfer_model.summary()
```

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_2_input (InputLayer)    [(None, 200, 25)]         0
_________________________________________________________________
lstm_2 (LSTM)                (None, 64)                23040
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_5 (Dense)              (None, 12)                396
=================================================================
Total params: 25,516
Trainable params: 396
Non-trainable params: 25,120
_________________________________________________________________
```

תוצאות:




```
transfer_model.evaluate(x_test_hr, y_test_ohe)
```

```
310/310 [==============================] - 2s 7ms/step - loss: 1.7372 - accuracy: 0.3644 - auc: 0.861
3

[1.7372111082077026, 0.3643622398376465, 0.8613112568855286]
```

המשקולות של מודל חיזוי קצב פעימות הלב לא תרמו לנו לדיוק גבוה יותר במודל סיווג הפעילויות, קצב הלב הוא מדד רציף וכן קשה יותר לחזות אותו, אולי זה הדבר שהשפיע על המשקולות ועל האופן שבו הן מתקשרות לפעולת הסיווג במודל המקורי. בנוסף לכך, יתכן כי אימון על מספר epochs גדול יותר ישפר במעט את המשקלים שהמודל לומד וכך גם את תוצאת החיזוי.

סעיף f –

רעיונות לשיפורים למודל המקורי:

1. הוספת פונק' אקטיבציה לשכבות המודל.
2. בניית מודל מורכב יותר מבחינת השכבות.
3. הוספת רגולציה ע"מ למנוע התאמת יתר.

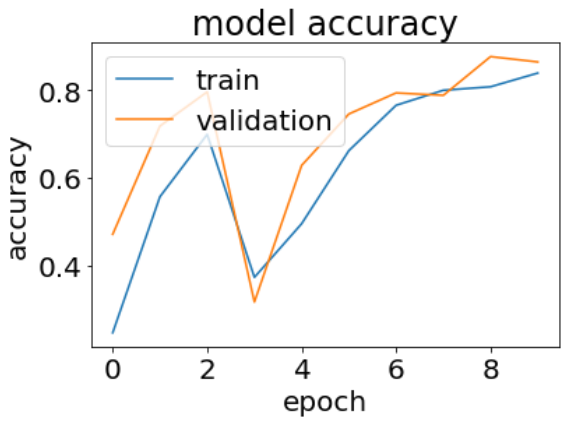
סעיף g –

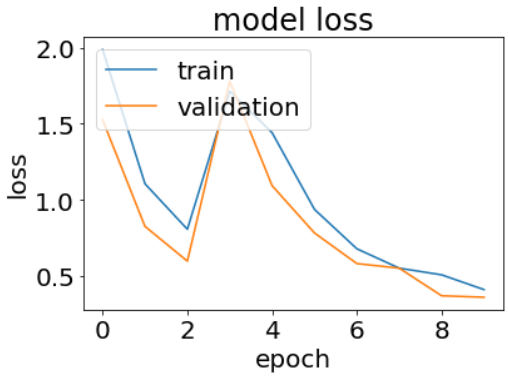
נממש את השיפור הראשון שהצענו :

```
improve1 = Sequential()
improve1.add(LSTM(24, input_shape=(x_train.shape[1],x_train.shape[2]), return_sequences=True, activation='tanh'))
improve1.add(Dense(24, activation='relu'))
improve1.add(LSTM(12, return_sequences=False, activation='tanh'))
improve1.add(Dense(12, activation='relu'))
improve1.add(Dense(y_train_ohe.shape[1],activation='softmax'))
improve1.compile( optimizer= 'adam',loss='categorical_crossentropy', metrics=['accuracy'])
improve1.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_3 (LSTM)                (None, 200, 24)           4704
_________________________________________________________________
dense_6 (Dense)              (None, 200, 24)           600
_________________________________________________________________
lstm_4 (LSTM)                (None, 12)                1776
_________________________________________________________________
dense_7 (Dense)              (None, 12)                156
_________________________________________________________________
dense_8 (Dense)              (None, 12)                156
=================================================================
Total params: 7,392
Trainable params: 7,392
Non-trainable params: 0
```

תוצאות :

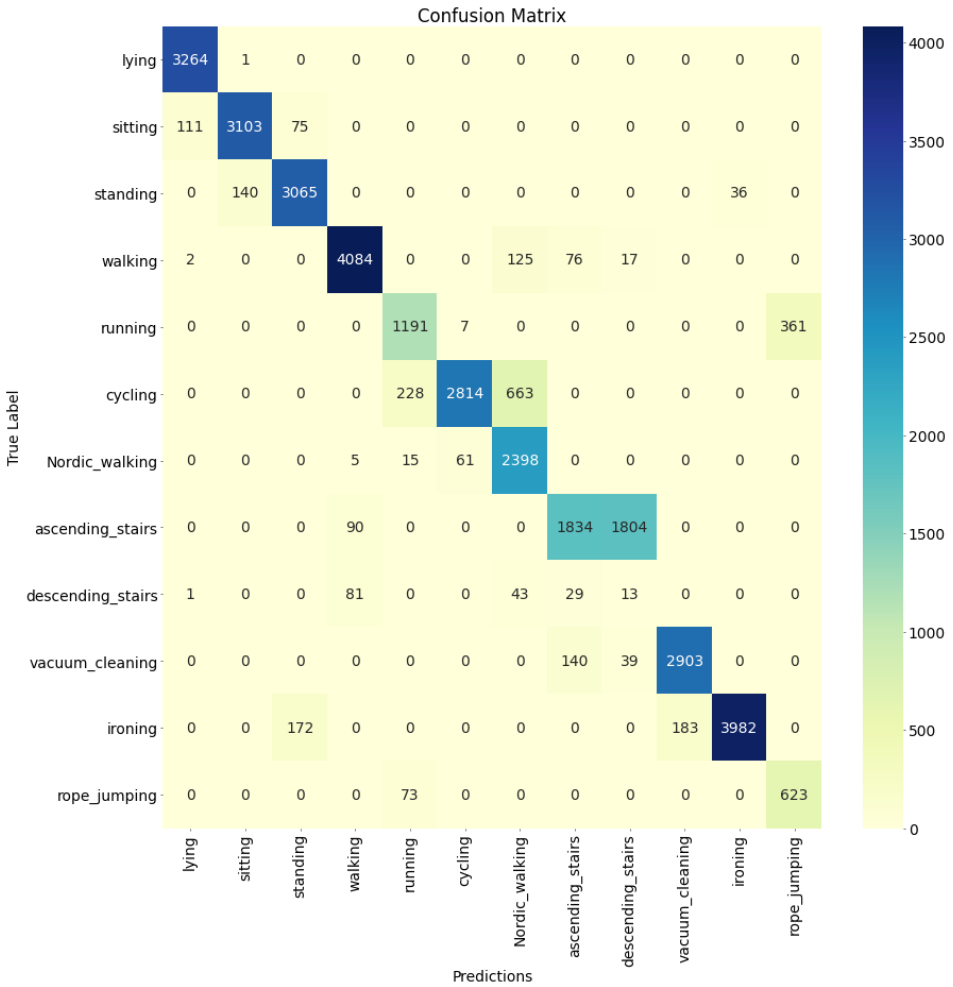





```
improve1.evaluate(x_test, y_test_ohe)
```

```
1058/1058 [==============================] - 9s 9ms/step - loss: 0.3623 - accuracy: 0.8648

[0.3622876703739166, 0.8647642731666565]
```



ישנו שיפור ניכר בתוצאות – דיוק של 86% לעומת 53% במודל המקורי

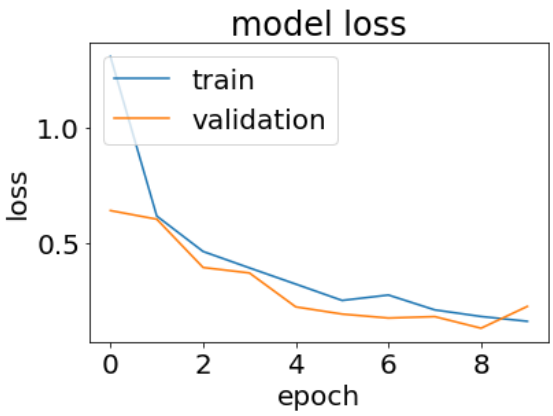
אפשר לראות כי עדיין המודל מתקשה בין פעולות של ירידה במדרגות לעומת עלייה במדרגות שהן פעולות מאוד דומות מבחינת הביצוע שלהן.

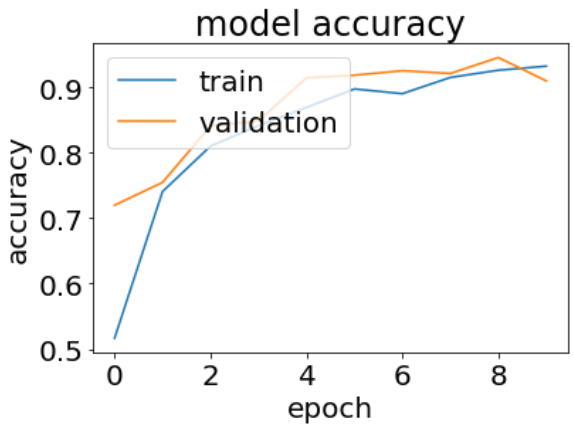
מימוש השיפור השני:

```
improve2 = Sequential()
improve2.add(LSTM(64, input_shape=(x_train.shape[1],x_train.shape[2]), return_sequences=True,
improve2.add(Dense(64, activation='relu'))
improve2.add(Dense(64, activation='relu'))
improve2.add(Dropout(0.2))
improve2.add(LSTM(32, return_sequences=False, activation='tanh'))
improve2.add(Dense(32, activation='relu'))
improve2.add(Dense(32, activation='relu'))
improve2.add(Dense(y_train_ohe.shape[1],activation='softmax'))
improve2.compile( optimizer= 'adam',loss='categorical_crossentropy', metrics=['accuracy'])
improve2.summary()
```

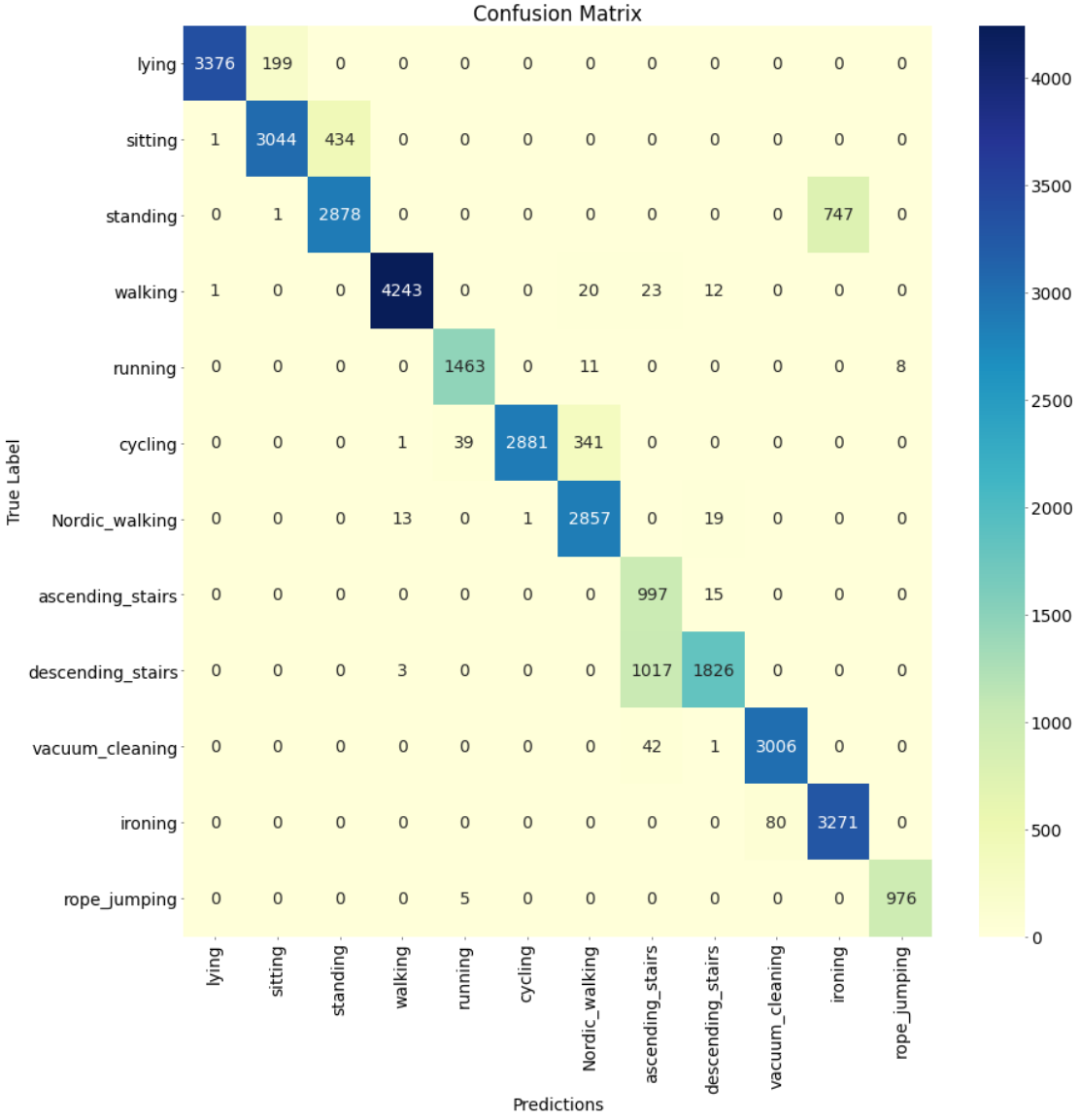
```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_5 (LSTM)                (None, 200, 64)           22784
_________________________________________________________________
dense_9 (Dense)              (None, 200, 64)           4160
_________________________________________________________________
dense_10 (Dense)             (None, 200, 64)           4160
_________________________________________________________________
dropout (Dropout)            (None, 200, 64)           0
_________________________________________________________________
lstm_6 (LSTM)                (None, 32)                12416
_________________________________________________________________
dense_11 (Dense)             (None, 32)                1056
_________________________________________________________________
dense_12 (Dense)             (None, 32)                1056
_________________________________________________________________
dense_13 (Dense)             (None, 12)                396
=================================================================
Total params: 46,028
Trainable params: 46,028
Non-trainable params: 0
```

תוצאות:





```
: improve2.evaluate(x_test, y_test_ohe)
1058/1058 [==============================] - 10s 10ms/step - loss: 0.2253 - accuracy: 0.9104
: [0.22531314194202423, 0.9103745818138123]
```



שיפור נוסף בתוצאות והמודל מגיע לדיוק של 91% - מבחינת סיווגים לא נכונים ניתן לראות שהוא מתבלבל בעיקר בין עמידה לגיהוץ שגם הן פעולות דומות מאוד (בדר״כ מגהצים בעמידה).

בנוסף עדיין יש בלבול בין ירידה במדרגות ועלייה במדרגות שזוהי גם אותה פעולה מבחינת ביצוע אך בכיוון שונה.

התוצאה שקיבלנו גבוהה יותר מה-benchmark של מודלי ה-ML שראינו אבל עדיין לא מגיע לתוצאות שמצאנו בשאלה 1 של מודלים שהם מורכבים הרבה יותר מהמודל שלנו ושנוסו ע״י מומחים בתחום.