**סדנא בלמידה עמוקה – עבודת הגשה #2**

עילי פרידמן – 305709248

סער גוטמן - 201543808

אורן שור – 204076053

**חלק 1:**

1. PAMAP2 הינו dataset המכיל מידע של ניתור כ- 18 פעולות פיזיות שונות שנמדד אצל 9 נבדקים שונים (8 גברים ואישה אחת). הנתונים נאספו ע"י 3 יחידות מדידה אינרציות ומדידת הדופק של המשתתפים ברגע מסוים המיוצג ע"י timestamp.
   * כל שורה במאגר הנתונים מכילה את הנתונים הבאים:

1. timestamp (s)

2. activityID

3. heart rate (bpm)

4-20. IMU hand

21-37. IMU chest

38-54. IMU ankle

כל אחד מחיישני ה- IMU מכילים את הנתונים הבאים:

The IMU sensory data contains the following columns:

1. temperature (Â°C)

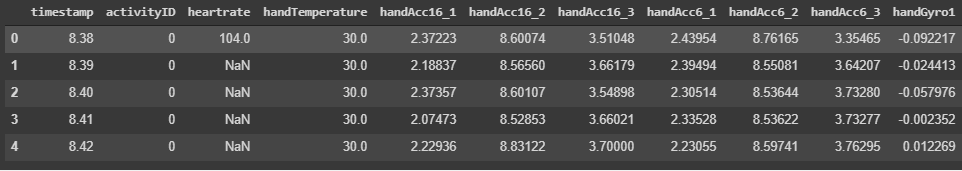
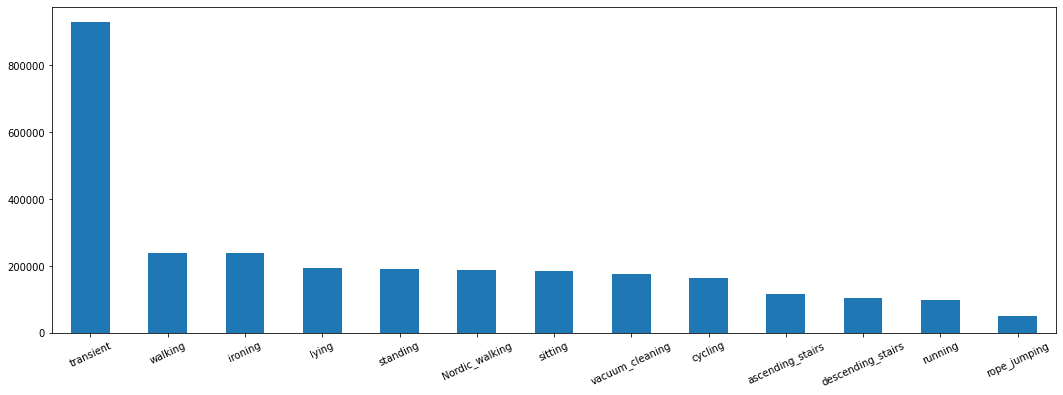
2-4. 3D-acceleration data (ms-2), scale: Â±16g, resolution: 13-bit

5-7. 3D-acceleration data (ms-2), scale: Â±6g, resolution: 13-bit

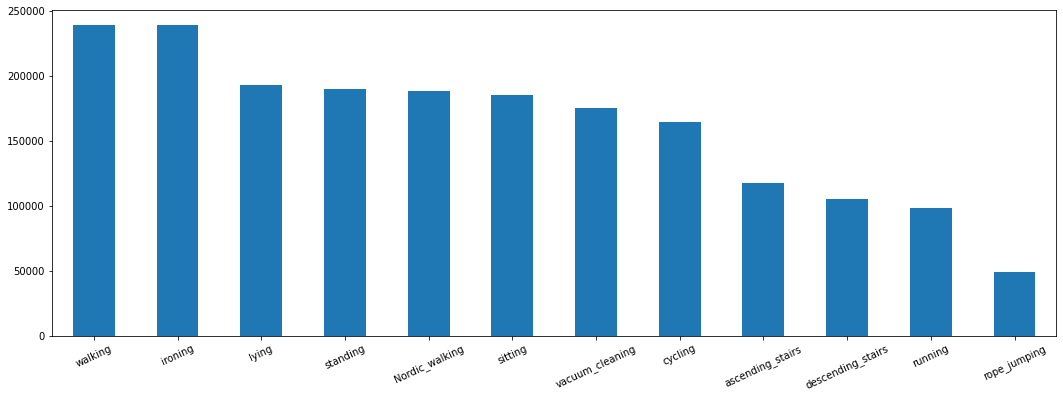
8-10. 3D-gyroscope data (rad/s)

11-13. 3D-magnetometer data (Î¼T)

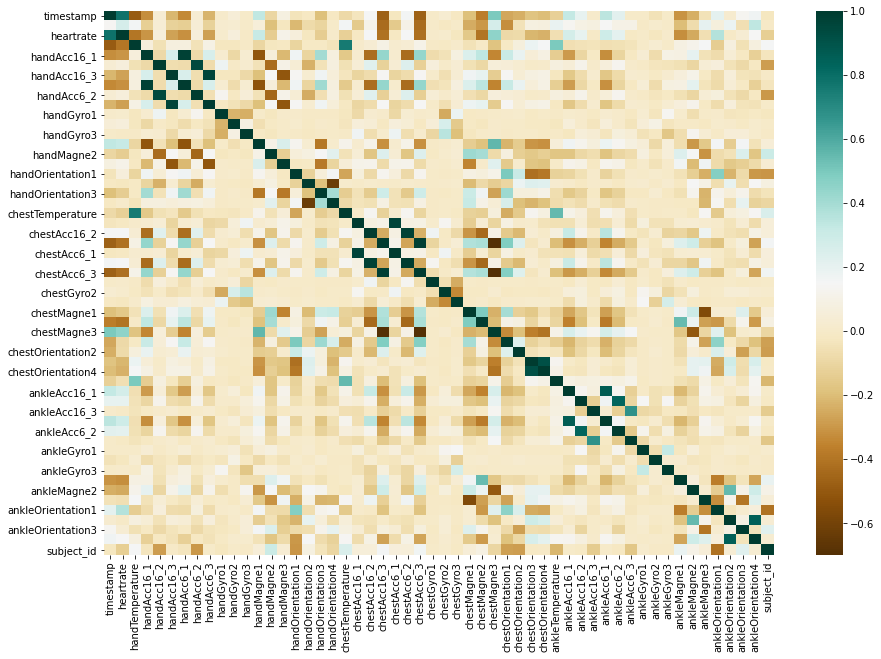
14-17. orientation (invalid in this data collection)

* + דוגמא לכמה שורות ב- dataset (שורות חלקיות):
  + ה- shape של הdataset: (2872533, 55)
  + מספר הפעולות הנתון ב- dataset: 13
  + פיזור המידע לפעולות השונות:

בגרף הנ"ל ניתן לראות כי הפעולה "transient" המתארת מצב ביניים שבו הנבדק לא עשה שום פעולה ייחודית מוסיפה "רעש" רב לנתונים ולא מתארת שום מצב "חשוב". לכן בחרנו "לנקות" את הנתונים ולהסיר את התצפיות המסווגות כפעילות זו.

* + פיזור המידע לפעולות השונות – כאשר ניקינו את התצפיות המסווגות כפעילות "transient":

מהגרף הנ"ל ניתן לראות כי אכן מאגר הנתונים "מאוזן".

* + מפת חום המתארת את ההתאמה בין הפיצ'רים (המדדים) ב- dataset:

ניתן לראות כי ישנה התאמה טובה בין המדדים של הטמפרטורות בחלקי הגוף השונים (כפי המצופה שחום גופנו יהיה דומה בחלקים השונים של הגוף). בנוסף ניתן לראות שהמידע מחיישני ה- gyroscope תואם במידה נמוכה מאוד לשאר המידע, דבר המעיד על חוסר חשיבותם באימון המודל.

1. המשימה הנדרשת במצב זה הינה משימה סיווג. בהינתן תצפית המוצגת כסט תצפיות חדש (ע"י כל אמצעי המדידה) של אדם ברגע מסוים, המשימה הינה לסווג את התצפית לפעולה המתבעת ע"י אותו אדם – לפי מאגר הפעולות הנתון.

כדי לפתור משימה זו, נאמן את מודל על סט אימון שבו סיווג התצפיות ידוע (supervised).

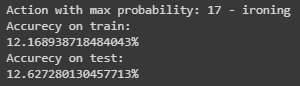
1. במשימות self-supervised שניתן לבצע לפני אימון המודל נרצה לנצל את המצאות מידע נוסף שנמצא ב- dataset. לדוגמה, נוכל להשתמש בפיצ'רים הנמצאים ב- dataset לבניית מסווג שאינו קשור ישירות למשימה שאנו מנסים לבצע, ולאחר מכן להשתמש במשקולות שנלמדו ע"י הרשת ובניית מסווג שכן מבצע את המשימה המקורית. כך נוכל ליצור מסווג לפעולות השונות שלא בהסתמך על התצפיות המסווגות הקיימות (שמספרן עשוי להיות לא מספיק ע"מ ליצור מודל מדויק).

דוגמאות לשימוש במידע הנוסף ע"מ לאמן מודל ללא הגבלת כמות תצפיות:

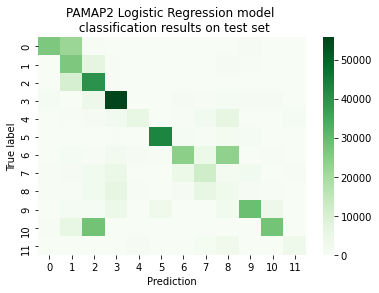
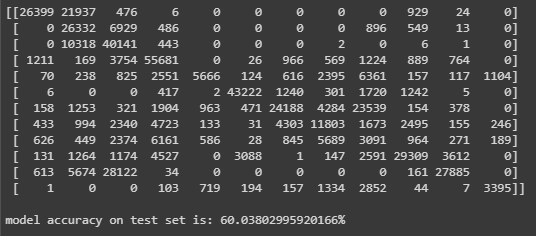
1. שימוש במשימות חיזוי ע"ב היסטוריית הערכים: מודל שחוזה את הדופק של הנבדק בנקודת זמן מסוימת בהסתמך עלX נקודות זמן קודמות. בכך נוכל לייצר אינסוף תצפיות שיהיו סט האימון של המודל המסווג את הפעילות הנוכחית של הנבדק.
2. שימוש במשימות חיזוי ע"ב ערכים של מדדים אחרים: מודל שחוזה את הדופק של הנבדק בנקודת זמן מסוימת על בסיס ערכי המדדים האחרים באותה נקודת זמן, למשל שימוש בחום הגוף ע"מ לחזות את הדופק של הנבדק באותו רגע.

**חלק 2:**

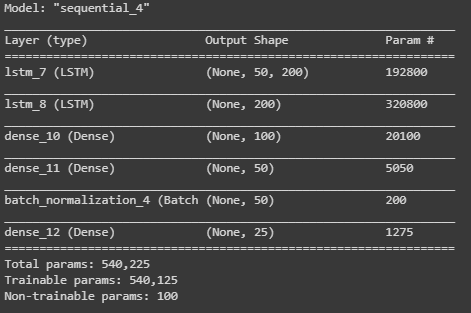
1. **אסטרטגיית וולידציה לאימון המודל:** כמתבקש בעבודה השתמשנו בנדבקים 7 ו- 8 כסט הבדיקה, ובנבדקים 1-6 ו-9 כסט האימון. את הוולידציה עשינו ע"י חלוקת סט האימון לפי המקדם 0.2 כלומר, 80% לאימון ו- 20% לוולידציה.
2. **Naïve baseline solution:** בתחילה נמצא את הפעולה עם ההסתברות הגבוהה ביותר – הנפוץ ביותר בסט האימון. נקבע את סיווג כלל התצפיות להיות הפעולה הנ"ל, ונבדוק את אחוז הדיוק מול המידע המקורי. נציג את אחוז הדיוק על סט האימון ועל סט הבדיקה:



ניתן לראות שהפעולה הנפוצה ביותר היא גיהוץ. בנוסף נשים לב שאחוז הדיוק נמוך יחסית אך זאת ניתן להסביר בכך שכל הפעולות מקבלות את אותו הסיווג וזה בוודאות יוביל לשגיאות רבות.

1. **משימת machine learning קלאסית:** בחרנו להשתמש במודל Logistic Regression ליצירת מודל ML קלאסי, מודל זה ישמש בעבורנו כ- solid benchmark למודל שנבנה בעצמנו.
2. . לאחר הרצת המודל על כלל סט האימון קיבלנו 60.038 אחוזי דיוק.
3. **מודל רשת נוירונים בסיסית:**

הכנת הנתונים:

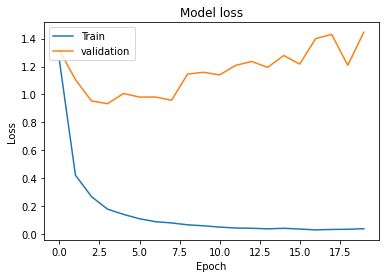
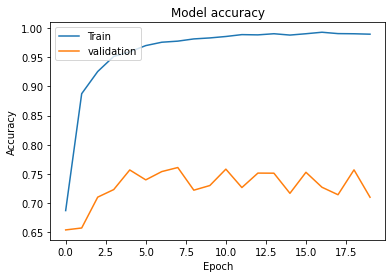
1. הסרת התצפיות המסווגות כפעולה שה- ID שלה הוא 0 (פעולת מעבר – לא פעולה מוגדרת).
2. השלמת ערכים חסרים במאגר המידע בשיטת אינטרפולציה.
3. בניית הקלט למודל ע"י הרכבת תצפית מ-50 נקודות זמן המסווגות לאותה פעולה ונוצרו בעבור אותו נבדק.

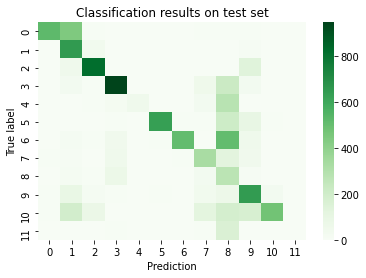
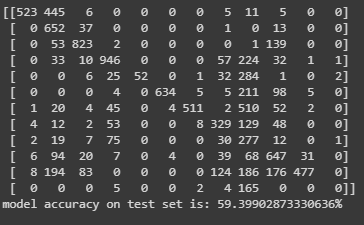
מבנה המודל:

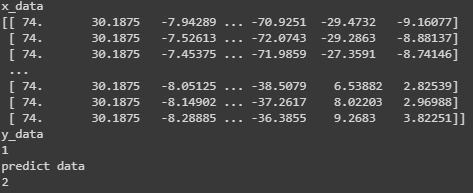
1. 2 שכבות LSTM
2. 2 שכבות Dense
3. שכבת BatchNormalization
4. השכבה האחרונה הינה שכבת dense המכילה את פונקציית האקטיבציה "softmax" המחלקת ל-25 קטגוריות שונות.

הגדרת המודל: נבחר בפונקציית ההפסד 'categorical\_crossentropy' זאת מכיוון שהמשימה שלנו היא משימת סיווג לסוג הפעולה (הקטגוריה), נבחן את המודל שלנו ע"פ אחוז הדיוק.

התוצאות:

אחוז הדיוק של המודל: 59.4%

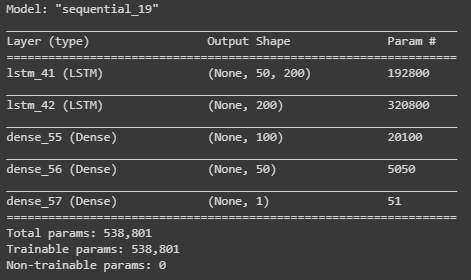


דוגמא לסיוג שגוי: ניתן לראות שהמודל סיווג תצפית זאת כפעולה מספר 2 אך בפועל סיווגה של תצפית זו הינו 1. בנוסף ניתן לראות את המידע על תצפית זו.

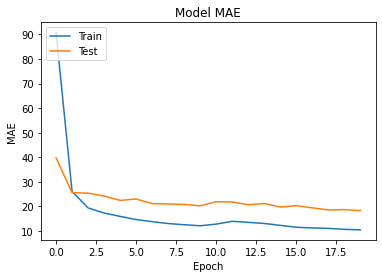
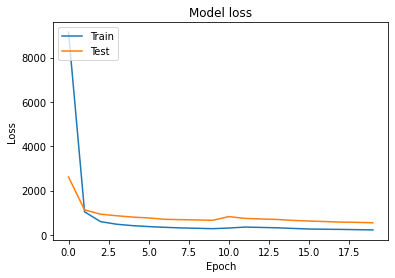
שגיאות עלולות לקרות עקב כך שאימנו את המודל ע"פ תצפיות המורכבות מ- 50 נקודות זמן, זה עלול להיות מעט מידי זמן שלא מביא לייצוג נכון של המצב.

1. **משימת Self-supervised**: נבחר לאמן מודל על משימה שונה מהמשימה המוגדרת. נאמן מודל לחזות את ערך הדופק של הנדבק בתצפית מסוימת על בסיס יתר ערכי המדדים באותה תצפית.

עדכון מבנה הנתונים: קובץ ה- X מכיל את כל הנתונים משאר המדדים ומספר הנבדק ללא עמודת הדופק, קובץ ה- Y מורכב ממספר הנבדק ועמודת הדופק.



המודל: בסעיף זה נשתמש במודל זהה לסעיף הקודם פרט לכך שפונקציית האקטיבציה של שכבת ה- LSTM הראשונה ושל שכבת ה- dense האחרונה היא "relu" וזה בגלל שהמשימה כעת היא משימת רגרסיה ולא סיווג. בנוסף נעריך את המודל שלנו ע"פ ערך ה- MAE ופונקציית ההפסד היא "MSE".

תוצאת המודל:

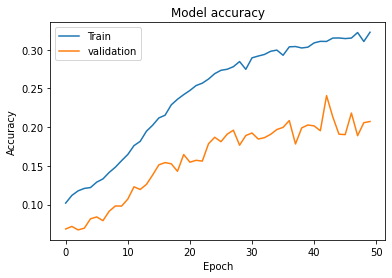
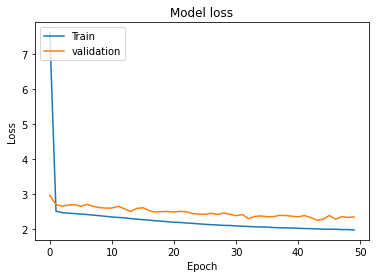
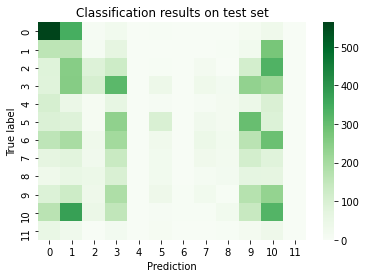
התאמת המודל המאומן למשימת הסיווג המוגדרת:

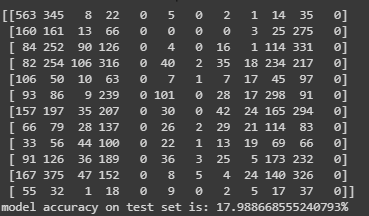
תחילה נקבע את משקלי כל השכבות של המודל שאימנו במשימת הרגרסיה, לאחר מכן נבטל את השכבה האחרונה (שהכילה את פונקציית האקטיבציה "relu") ונוסיף שכבה אחרת של Dense המכילה את פונקציית האקטיבציה "softmax" המחלקת את התצפיות ל- 25 קטגוריות (פעולות שונות).

כעת נקבע שמשימת המודל שוב הינה סיווג ולכן פונקציית ההפסד תהיה "categorical\_crossentropy", והמדד לטיב המודל הוא שוב דיוק (accuracy).

תוצאות המודל המותאם:

אחוז הדיוק של המודל: 17.988%



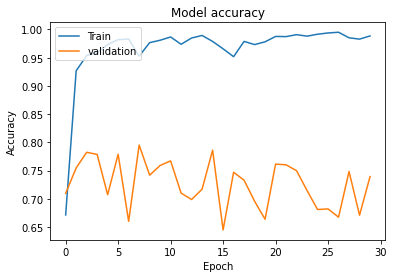
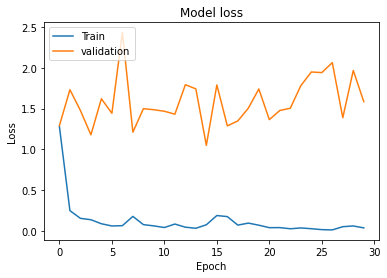
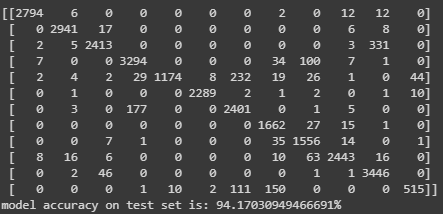
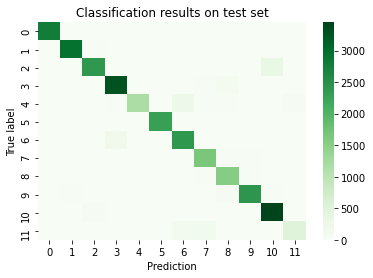


כפי שניתן לראות אחוז הדיוק של המודל הוא בערך 18%, אחוז לא גבוה במיוחד. תוצאות לא טובות אלו אפשר להסביר בכך שאולי משימת הרגרסיה של דופק הנבדקים הינה משימה קשה והמודל לומד בצורה שונה שלא מתאימה למשימת הסיווג.

1. **הצעות לשיפור תוצאות המודל:**
2. הגדלת מספר התצפיות שעל בסיסן אנו מייצרים את התצפיות שבסופו של דבר מרכיבות את סט האימון והבדיקה, כלומר שימוש ביותר נקודות זמן.
3. נרמול הנתונים ע"פ הממוצע וסטיית התקן.
4. מיון המדדים והקצאת משקל גבוה יותר למדדים המשפיעים יותר.
5. **מימוש שתי הצעות מהסעיף הנ"ל:** בחרנו לממש את שתי ההצעות הראשונות – שימוש ב-250 נקודות זמן ע"מ ליצור את התצפיות לסט האימון והבדיקה, ונרמול הנתונים לפי הממוצע וסטיית התקן. הורדנו את שכבת ה- BatchNormalization ע"מ לא לנרמל שוב את הערכים.

אלו התוצאות:

אחוז הדיוק של המודל: 94.2%



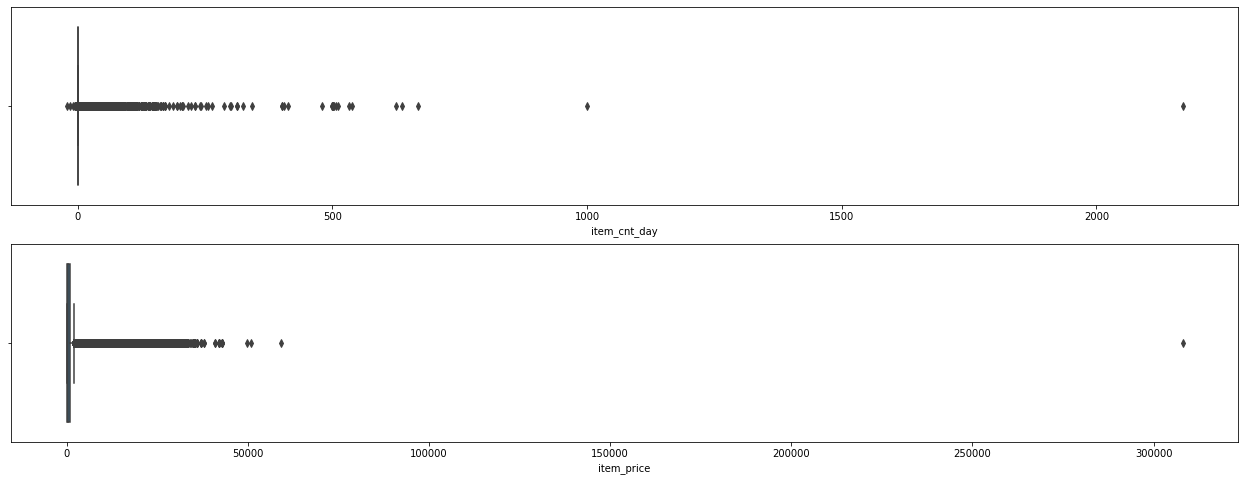
נשים לב שתוצאות מודל זה טובות ואיכותיות הרבה יותר משאר המודלים, מכך ניתן לומר שנרמול הנתונים והגדלת כמות נקודות הזמן המרכיבות את התצפיות ל- 250 נקודות אכן עזר והמודל לומד יותר טוב ומסווג בצורה טובה מאוד את הפעולה הנעשית ע"י הנבדק.

נשווה תוצאות מודל זה לתוצאות מודל ה- Logistic Regression שמשמש בעבורו כ- solid benchmark, מהשוואה זו ניתן לומר ללא צל של ספק שהמודל שבנינו מתאים יותר לפתרון המשימה המוגדרת (סיווג סט ערכי מדדים לפעולות).

**חלק 3:**

1. הצטרפנו לקאגל, ושם הקבוצה שבחרנו הוא BGU-DL Horrible & Bitter.

החלטנו לבצע את כל תהליך הpreprocessing ותהליך הfeature engineering בתחילת המחברת.

תחילה רצינו לראות כיצד נראית התפלגות המכירות והמחירים בכל הדאטה.

קל לראות כי ישנם outliers בדאטה הן במחירים והן במכירות היומיות. במכירות היומיות ניתן לראות שקיימות דגימות בהן מספר המוצרים שנמכר הוא שלילי. אנו מניחים כי מדובר או בטעות בהזנת הנתונים, או שהדאטה מייצג גם החזרות של פריטים לחנויות.

בכל מקרה מדובר בתצפיות חריגות, שלא יעזור לנו להיבן את המקרה הכללי, ולכן נפטר מהן. הורדנו מהדאטה דגימות שבהן מחיר הפריט עלה על 10,000, ודגימות שבהן מספר המחירות היומי הוא שלילי, או גדול מ1500.

לפני הניקיון היו לנו 2,935,849 תצפיות, ולאחריו נותרנו עם 2,928,491 תצפיות . סה"כ איבדנו 7,358 דגימות.

מתוך עמודת התאריך שבקובץ המכירות נוכל לחלץ את החודש בשנה עבור כל רשומה, וכן את השנה עצמה. כמו כן, בהתבוננות בערכי שמות החנויות בקובץ החנויות, ניתן לראות ששם החנות מתחיל דווקא בעיר שבה נמצאת החנות. מידע זה עשוי לסייע לנו, ולכן נחלץ גם את הפיצ'ר הזה. עבור הפיצ'ר הזה ביצענו Label Encoding, שכן אין לנו שימוש בשם העיר עצמה, אלא רק ביכולת להבחין בין ערים שונות.

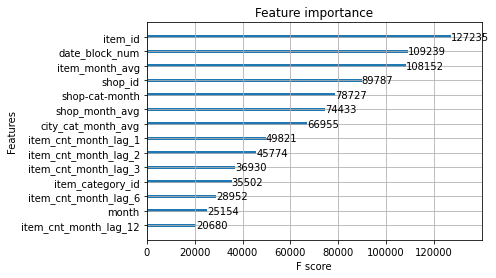
בנוסף, בקובץ הפריטים ישנה עמודה נוספת של קטגורית הפריט. סביר להניח שמכירות הפריטים ששיכים לאותה קטגורייה מתנהגים באופן דומה, ולכן גם פיצ'ר זה יתרום לנו.

עד לשלב זה עסקנו בפיצ'רים קטגוריאליים, שאיתם נצטרך להתמודד בהמשך המשימה. לסיכום, כעת יש לנו את הפיצ'רים הקטגוריאליים הבאים: date\_block\_num, shop\_id, item\_id,month ,year ,item\_category\_id ,city.

כעת חישבנו את משתנה המטרה – מספר מכירות חודשיות עבור <פריט, חנות>. כאמור, הדאטה מגיע כאשר נתון לנו מספר המכירות היומי. בנוסף, הוספנו פיצ'רים שמתארים עבור כל <פריט, חנות> את מספר היח' שנמכרו בחודש הקודם לדגימה, חודשיים, שלושה חודשים, חצי שנה ושנה.

כעת ניגשנו להנדס פיצ'רים חדשים המתבססים על משתנה המטרה שלנו. סך הכל הוספנו 4 פיצ'רים – ממוצע מכירות פריטים בחנות, ממוצע מכירות של פריט בחודש, ממוצע מכירות של של קטגורייה בעיר בחודש נתון, וממוצע מכירות של קטגורייה בחנות בחודש נתון. כאן סיימנו לעסוק בFeature engineering, ולאחר Downcast לפיצ'רים שלנו כדי לחסוך בזיכרון ולזרז את תהליך הלמידה, עברנו לבניית המודלים וההשוואות ביניהם.

1. **בניית מודל Machine Learning קלאסי:** בחרנו בשלב זה לממש מודל XGBoost שמתאים מאוד לדאטה טבלאי עם מספר גדול של פיצ'רים. הגדרנו מודל עם 100 איטרציות ועומק מקסימלי של 7. בסוף שלב האימון הגענו ל RMSE על הולידציה של 1.34. זה יהיה הbenchmark שלנו עבור המודלים הבאים.

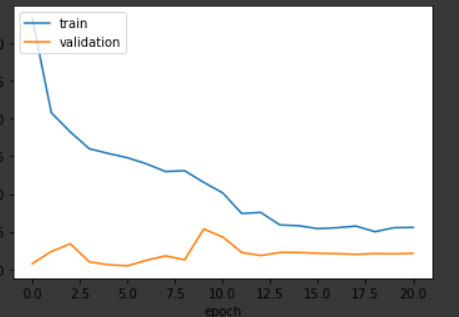
לאחר אימון המודל, ציירנו גרף שמתאר את חשיבות הפיצ'רים לחישוב הרגרסיה: 

ניתן להתרשם ששני הפיצ'רים המרכזיים הם הפריט והחודש ("החודש" כאן במובן של הid של החודש מתוך שאר הדגימות, ולא חודש בשנה). העובדה שהפיצ'ר item\_month\_avg נמצא במקום השלישי מרמזת על עונתיות חזקה בדאטה, עוד יותר מאשר החנות עצמה בה נמכר הפריט.

כפי שניתן היה לצפות, ניתן לראות שמבין נתוני המכירות ההיסטוריים הרלוונטי ביותר הוא הנתון מהחודש שעבר, ולאחר בסדר יורד שאר הנתונים.

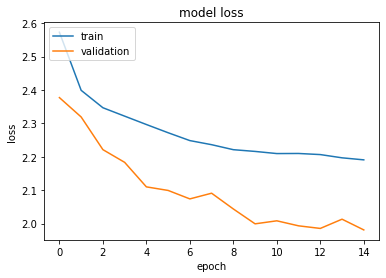
1. **בניית מודל Deep Learning בהתבסס רק על Embedding:** כעת בנינו רשת פשוטה יחסית עם שכבות Embedding עבור הפיצ'רים הקטגוריילי. בבחירת מימד הEmbedding הלכנו לפי כלל האצבע שהוצג הרצאה

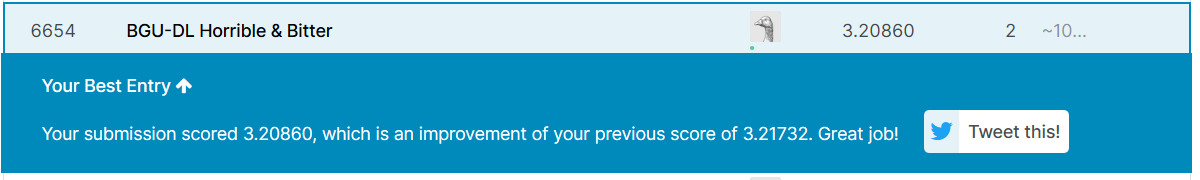
.את ה-Loss הגדרנו להיות RMSE.

בשלב זה נתקלנו בקושי גדול. למרות שהמודל התקמפל כצפוי, וההכנות שעשינו לדאטה, והידיעה שיש לנו מודל XGBoost עם ביצועים לא רעים, הרשת לא סיפקה ביצועים טובים, ותהליך הלמידה שלה היה משונה ולא אופייני. בתמונה המצורפת ניתן לראות את גרף הLoss, ולהתרשם שערכי הloss עבור הValidation היו נמוכים בהרבה מאלו של הTrain, ובמקום להשתפר הם הפכו גרועים.

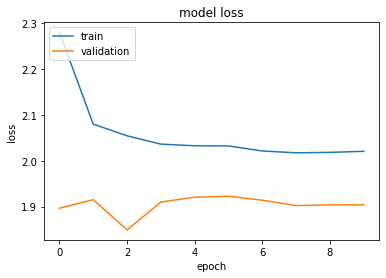
לאחר זמן מה של חקירות, הצלחנו להבין שאנחנו מזינים את הרשת בפיצ'רים בסדר שהיא לא מצפה להם, וכך מרחב האמבדינג שטרחנו להגדיר עבור כל פיצ'ר אינו מייצג את מספר הקטגוריות שבפועל מוזן לרשת.

לאחר שסידרנו את הקלט מחדש כך שיתאים לרשת שהגדרנו, קיבלנו תוצאות הגיוניות ותהליך למידה איכותי.

למרות זאת, בצועי המודל עדיין אינם טובים כמו מודל הXGBoost שבנינו בהתחלה. אנחנו מעריכים שזה בגלל שהארכיטקטורה שבנינו מתעלמת מהמון מידע שמודל הXGBoost ידע לנצל היטב. ניתן לראות זאת אם נתבונן שוב בדירוג חשיבות הפיצ'רים של מודל הXGBoost, שם הרוב המוחלט של הפיצ'רים החשובים הם דווקא בפיצ'רים הנומריים.

בתמונה ניתן לראות את גרף הLoss לאחר סידור הקלט. ברשת זו קיבלנו את הביצועים הבאים:loss: 2.1908 val\_loss: 1.9808. בתמונה מKaggle ניתן לראות שהשגנו RMSE של 3.21.

1. בשלב זה הוספנו את הפיצ'רים הנומריים הנוספים. ארכיקטקטורת הרשת נותרה זהה למעט עובדה זאת.

מודל זה סיפק את הביצועים הבאים: loss: 2.0206, val\_loss: 1.9037. ניתן לראות שאכן הביצועים השתפרו משמעותית בהשוואה למודל שהשתמש רק במשתנים הקטגוריאליים, אך עדיין מדובר בביצועים נחותים בהשוואה למדול הXGBoost. אנחנו מעריכים שזה כך לאור העובדה שהרשת שלנו היא אחת, בעוד XGBoost הוא סוג של אנסמבל שמתפתח כדי לדייק את המודל.

בנוסף, ממש מתבקש להוסיף למודל שכבת LSTM שתתחשב באופי הTimeseries-י של הדאטה, אך במסגרת לוח הזמנים להגשת המטלה, לא הצלחנו לגרום למודל לעבוד.

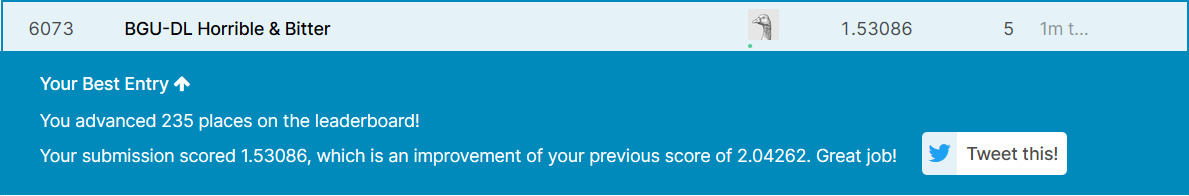
1. ראו פירוט במחברת עצמה.
2. **Feature Extraction:**

כעת עשינו שני נסיונות לאימון Classifier חדש על גבי ארכיקטקטורת הרשת שבנינו. כמובן שבחרנו ברשת המוצלחת יותר שמתחשבת גם בפיצ'רים הנומריים.

בניסיון הראשון, החלפנו רק את הClassifier של הרשת, ובמקומו שמנו מודל XGBoost, שהנחנו שיציג ביצועים טובים. הגדרנו 200 איטרציות, ועומק מירבי של 5.

בניסיון השני החלטנו לנסות לתת לXGBoost את מירב המידע, וחיברנו אותו לשכבת הConcatenate, כך חשוף בפניו כלל המידע המגיע מהאינפוט ומהEmbedding של הפיצ'רים הקטגוריאליים. הגדרנו 200 איטרציות, ועומק מירבי של 15, כדי לשקף את העובדה שכעת יש הרבה יותר פיצ'רים.

בסופו של דבר, למרות עושר המידע של המודל השני, דווקא הראשון הציג ביצועים טובים יותר, והצליח לעקוף את ביצועי הVanilla XGBoost מסעיף ג. הביצועים של המודל validation\_0-rmse:1.03445 validation\_1 rmse:1.32153. בתמונה ניתן לראות את הביצועים ביחס לתחרות בKaggle.



המודל השני הציג ביצועים מעט פחות טובים, אבל בעיקר תפס המון זיכרון ומשך האימון שלו היו ארוך מאוד בגלל עושר המידע. validation\_0-rmse:1.30085 validation\_1-rmse:1.34006