פרוייקט מעשי בקורס מבוא לבינה מלאכותית / סמסטר ב' ה'תשפ"ג

מרצה: פרופ' ראובן כהן
מתרגל: דניאל אביצרור
מגישים: עדן ברדה (208932202) ואבישי דמס (322884560)

1. נתוני מסגרת:

- קישור למסד הנתונים:

https://www.kaggle.com/datasets/syuzai/perth-house-prices

- תיאור הנתונים:

הנתונים הם נתוני בתים שנמכרו בעיר פרת' באוסטרליה בין השנים 1988 ל-2020. במסד הנתונים 33,656 דגימות של בתים. על פי הגרפים שאראה בהמשך, התפלגותם על פי כלל הפרמטרים רחבה ולכן ניתן להעריך כי הם מייצגים את כלל הנכסים בעיר.

- העמודות הקיימות:

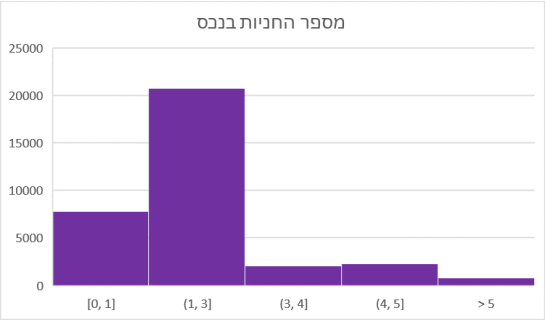
1. *address* - כתובת הבית (שם רחוב ומספר בית).
2. *suburb* - שם השכונה.
3. *price* - מחיר בדולרים אוסטרליים.
4. *bedrooms* - מספר חדרי השינה.
5. *bathrooms* - מספר חדרי האמבטיה.
6. *garage* - מספר החניות.
7. *land_area* - שטח הקרקע במ"מ.
8. *floor_area* - השטח הבנוי במ"מ.
9. *build_year* - שנת בנייה.
10. *cbd_dist* - המרחק ממרכז העיר.
11. *nearest_stn* - תחנת הרכבת הקרובה ביותר.
12. *nearest_stn_dist* - המרחק מתחנת הרכבת הקרובה.
13. *month_sold* - חודש מכירה.
14. *year_sold* - שנת מכירה.
15. *postcode* - המיקוד.
16. *latitude* - קו רוחב בנ.צ שמתאר את המיקום.
17. *longitude* - קו אורך בנ.צ שמתאר את המיקום.
18. *nearest_sch* - שמו של בית הספר הקרוב.
19. *nearest_sch_dist* - מרחק מבית הספר.
20. *nearest_sch_rank* - דירוג בית הספר.

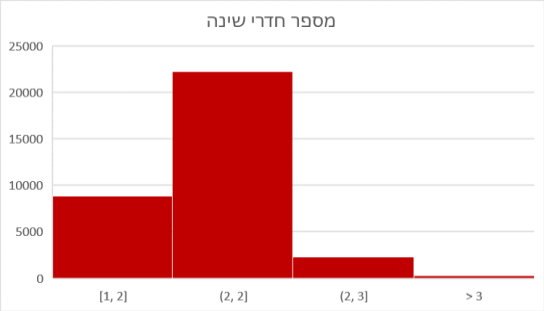
- הפיצ'רים בהם השתמשנו:

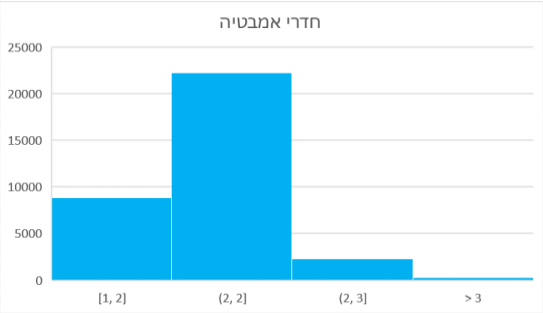
bedrooms, bathrooms, garage, land_area, floor_area, build_year, cbd_dist, nearest_stn_dist, month_sold, year_sold, latitude, longitude, nearest_sch_dist.

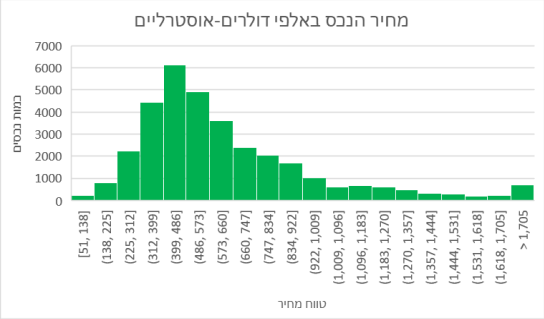
מתוך הבנה שאלו הפיצ'רים שמחזיקים מידע רלוונטי על עלותו של הנכס (המיקום שלו, הרכיבים שלו וסביבתו, והזמן שבו נבנה ונמכר).

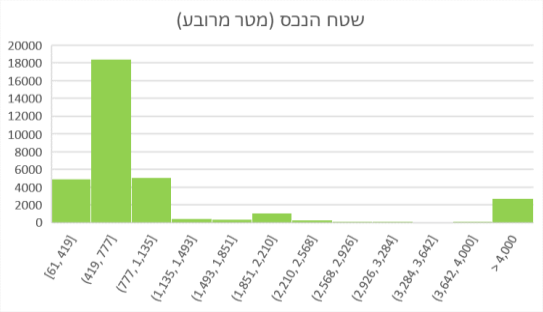
- התפלגות נתוני הפיצ'רים העיקריים:

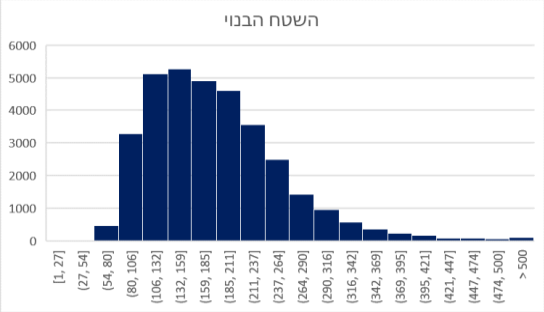


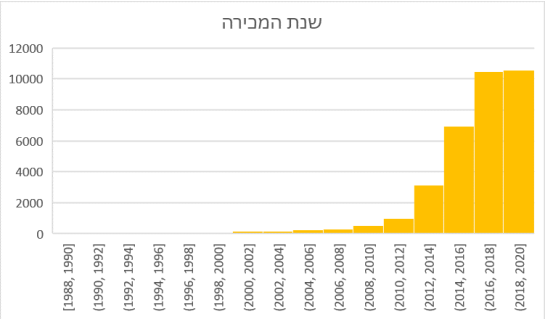


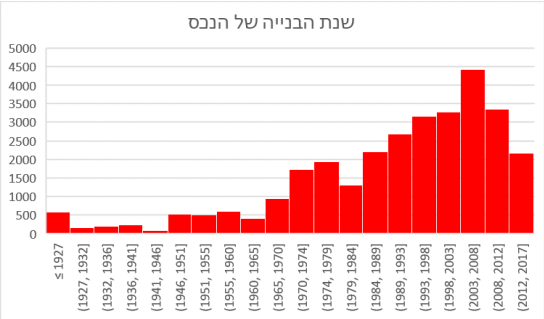






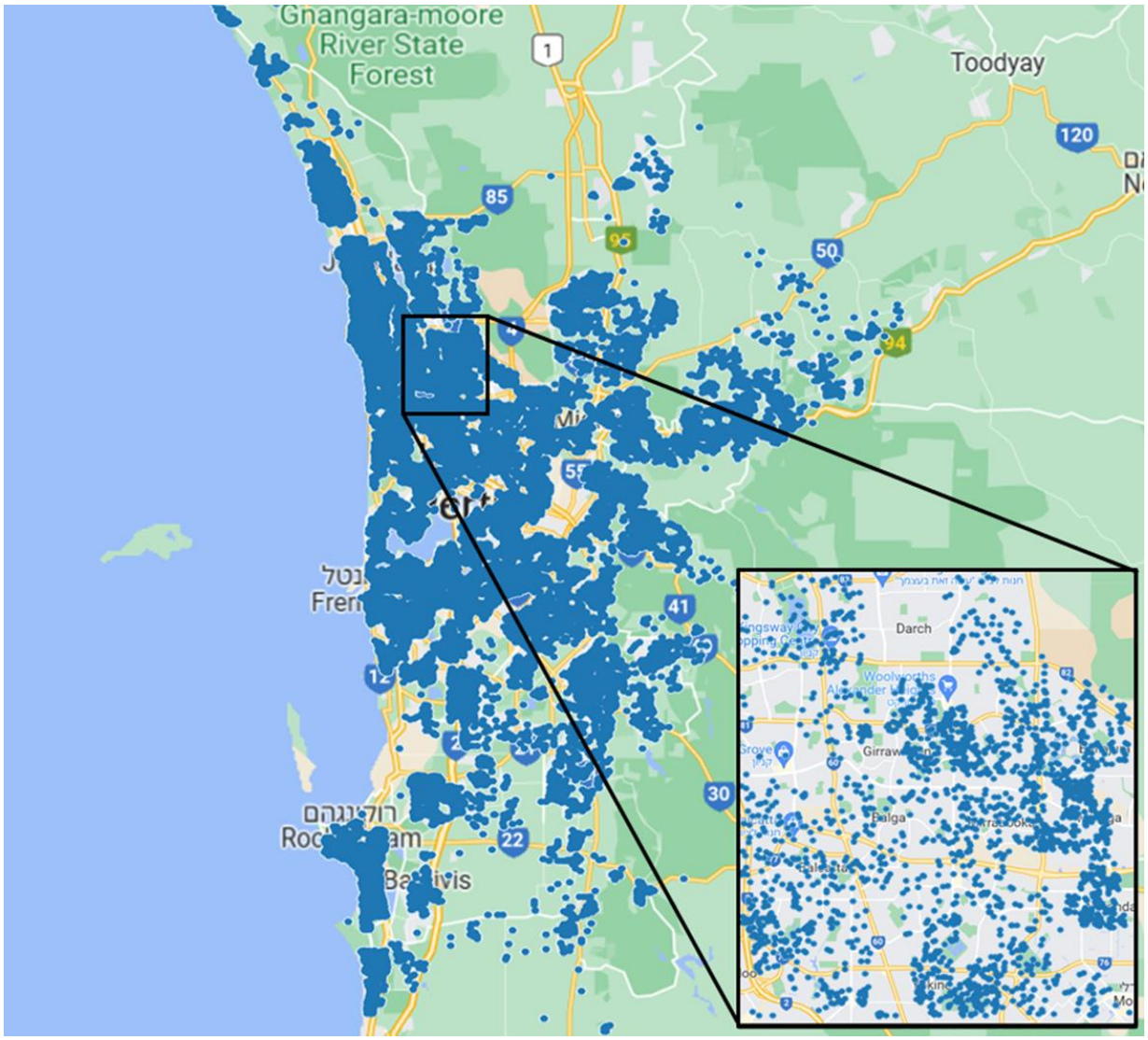






- התפלגות במרחב:

רצינו לדעת איך נראית ההתפזרות של הנתונים במרחב. ראשית, בכדי לאשש שהנתונים אמינים, ושנית, בכדי לדעת האם הם מייצגים את כלל העיר או לחלופין מתמקדים באזור מסוים. אז ייצאנו את הנתונים כפונקציה של קו האורך והרוחב והצגנו אותם על מפה של *Google Maps*. כך נראית התוצאה:



מתוצאה זו ניתן להסיק שהנתונים מתפלגים בצורה טובה בין חלקי העיר. על אף שבפאתי העיר ישנם פחות דוגמאות, ניתן להבין זאת מכך שבאמת יש פחות צפיפות נכסים בערים ככל שמתרחקים ממרכזם.

- העבודה עם הדאטה:

בתחילה "ניקינו" את הנתונים משורות בהם היה חסר מידע ונשארנו עם 28,256 דגימות של בתים. לאחר מכן חילקנו את עמודת התאריך לחודשים ושנים בשביל שהאלגוריתם יוכל לעבוד עם המידע.

2. <u>שיטות הלמידה</u>:

- הבעיה:

זוגות צעירים באוסטרליה (כמו בישראל) מתקשים לקנות דירה, ולכן נניח שזוג צעיר היה מעוניין לרכוש דירה באוסטרליה בעיר הנחמדה והיפה פרת' (באותה מידה יכול היה להיות זוג ישראלי, אבל אין מספיק נתונים). אבל מה לעשות, בפרת' (כמו בישראל) לא מוכנים לומר לך מה יהיה המחיר של הנכס לפני הגעה ומשא ומתן מתקדם.

בשביל לחסוך לזוג הצעיר שלנו זמן, בנינו אלגוריתם שמחשב עבור דירה האם סביר להניח שתהיה בתקציב של הזוג, או לחלופין, שאין טעם אפילו להתקשר לסוכן הנדל"ן.

- מטרת המודל:

לדעת בסבירות גבוהה האם בית עם פרמטרים מסוימים (שטח, מס' חדרים וכו') הוא בטווח המחירים של הזוג המדובר. כמקרה בוחן, בחרנו בטווח $400{,}000$ עד $500{,}000$ דולר אוסטרלי.

- האלגוריתמים:

בתחילה, ניסינו לפתור את הבעיה על ידי ניסיון להעריך את המחיר של כל דירה ואז לבדוק האם היא בטווח המחירים. אך גילינו שלמידה בינארית במקרה הזה קיבלה תוצאות מדויקות יותר (המודל הראשון שחוזה מחירים באמצעות ANN גם נמצא בין קבצי ההגשה: "$how\_much\_it\_costs\_ann$").

השתמשנו בשני אלגוריתמים לבצע למידה של הנתונים, רשת נוירונים (ANN) ו-$Random\ Forest$. כיוון שהנתונים מרובים וסבוכים (כמו למשל נ.צ) חשבנו שיהיה נכון ללמוד את הנתונים באמצעות רשת נוירונים (בקובץ המצורף "$worth\_a\_visit\_ann$"). עם זאת, הנחנו שכאשר שמאי מנסה לתמחר נכס הוא יעבור על רשימת היתרונות והחסרונות שלו ועבור כל יתרון או חסרון הוא יעלה או יוריד את ערך הנכס, זאת בדומה לעץ החלטה (בקובץ המצורף "$worth\_a\_visit\_random\_forest$"). ולכן בסה"כ החלטנו שיהיה נכון ללמוד את הנתונים באמצעות רשת נוירונים ולהשוואות את הצלחת המודל לזו המתקבלת מלמידה של $Random\ Forest$.

3. <u>הערכת ביצועים</u>:

- פונקציית הסיכון:

על אף שזאת כביכול משימת אשכול (האם זה בתקציב או לא), מניסויים שערכנו פונקציית $Binary\ Cross-Entropy$ לא החזירה ציונים טובים (גם אחרי טיוב ההיפר-פרמטרים). נראה שזה בגלל שבפועל מה שאנחנו מחפשים סוג של חוזה מחיר ולא רק מאשכל. ולכן בחרנו בפונקציית הסיכון $Mean\ Squared\ Error$, שטובה בתחזיות על פי נתוני עבר ובאמת קיבלנו תוצאות טובות.

- הערכת ביצועים:

את הערכת הביצועים שלנו עשינו לפי הצלחת מדד ה-AUC שבוחן עד כמה המודל מזהה דוגמאות אמת בתור אמת ושקר בתור שקר (כלומר הצלחה בו מונעת $false-positive$, ו-$false-negative$). כיוון שהמודל שלנו אמור לתת תשובה בינארית (האם המחיר צפוי להיות בטווח או לא) זה המדד שהכי היה חשוב לנו לבדוק. ובאמת הצלחנו להגיע בו לציונים טובים (~0.84 מתוך 1).

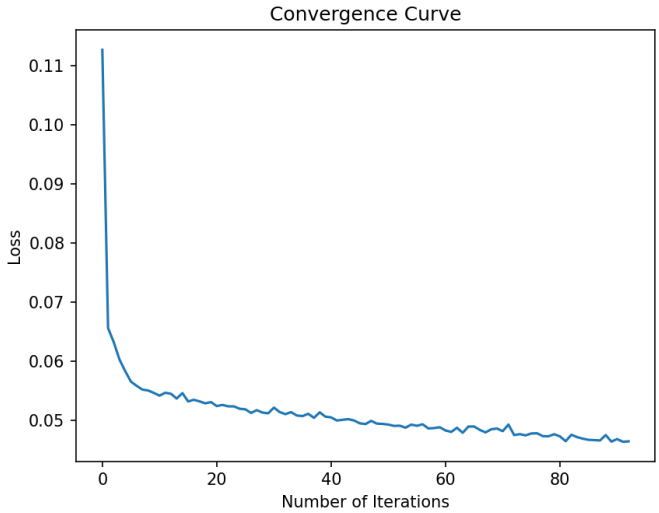
- פרמטרים:

את בחירת הפרמטרים (משקלים, הטיות, חלוקת הדוגמאות ל-$train$, $validation$ ו-$test$) עשינו אקראית בכדי שהמודל שלנו יהיה מאומן אופטימלית ולא יושפע מגורמים חיצוניים. אך, בכדי שנוכל לעקוב בצורה מדויקת אחר שינוי התוצאות למול טיוב האלגוריתם עשינו אקראיות שהשתמרה עבור כל פעולה ($random\_state = 42$).

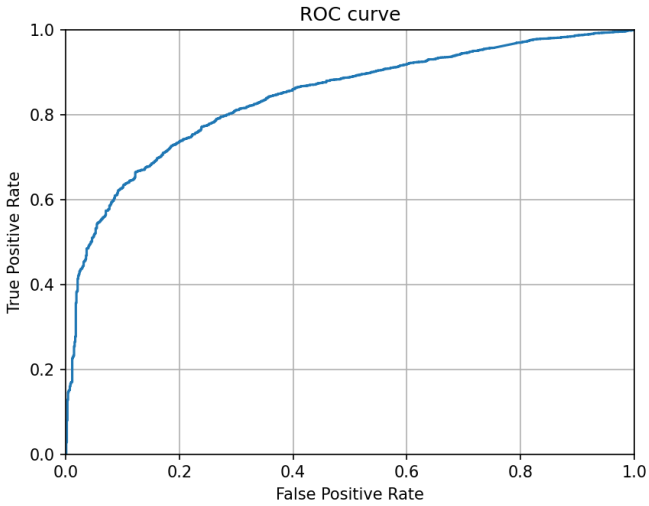
4. <u>ניתוח התוצאות</u>:

ברשת הנוירונים:

- זמן ריצה: 47 שניות
- מספר איטרציות: 92



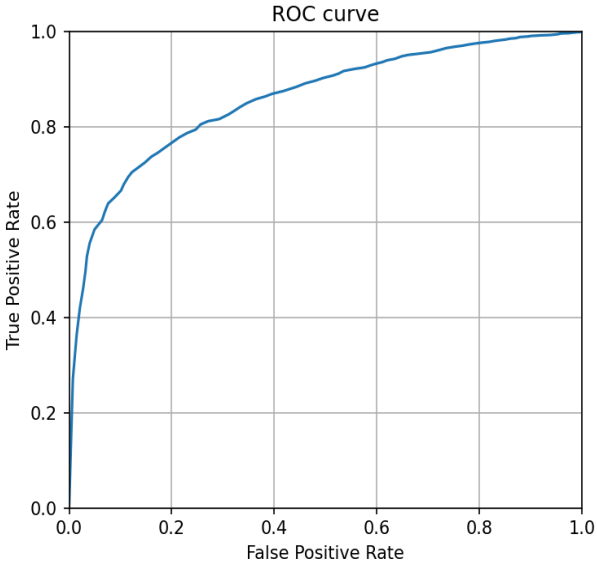
קצב ההתכנסות



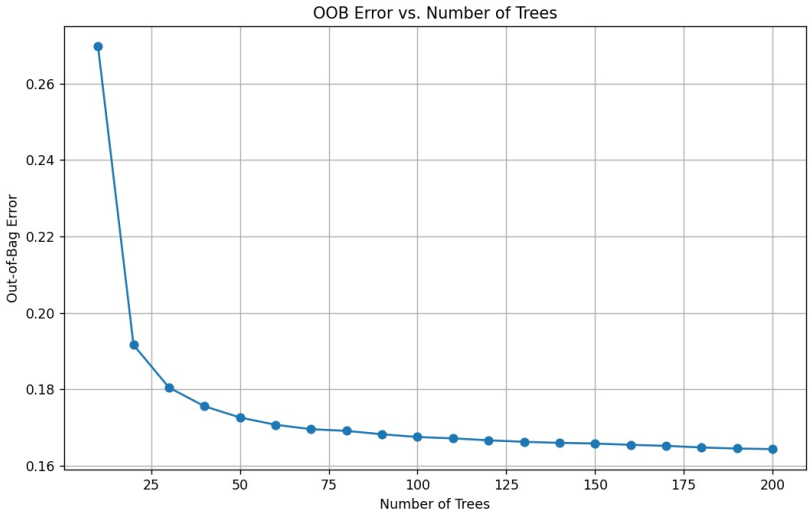
עקומת *ROC* של *ANN* (*AUC*: 0.84071)

- בסה"כ הכל, קיבלנו שהאלגוריתם קיבל תוצאות איכותיות ועל כן הוא אמין ומסוגל לתת לנו אינדיקציה טובה על האם דירה מסוימת תהיה בטווח המחירים שלנו או לא.

אף על פי כן, ב-*Random Forest* ה-*AUC* היה מעט גבוה יותר: 0.8595 (ב-26 שניות). וגם בו נצליח לקבל תשובה טובה לשאלה שלנו.



עקומת *ROC* של *Random Forest* (*AUC*: 0.85954)



מדד השגיאה על *Random Forest* ביחס למספר העצים

בסה"כ, שני האלגוריתמים ביצעו למידה טובה של הדאטה ויעריכו נכון בסבירות גבוהה פרטים ביתיים חדשים שיקבלו.

5. <u>חלוקת הדאטה</u>:

את הדאטה חילקנו ל-*Training, Validation* ו-*Test*.

- *Training data*:

כמות: 72% ממרחב הדוגמאות (20,345).

תיאור: הדאטה עליו האלגוריתם למד ובכך שיפר את פונקציית ההפסד שלנו.

שימוש: לימדנו מודל בסיסי את הדאטה בשביל שבהמשך נוכל למצוא עליו היפר-פרמטרים אופטימליים ולבדוק את איכות הלמידה שלו.

- *Validation data*:

כמות: 8% ממרחב הדוגמאות (2,260).

תיאור: הדאטה עליו בדקנו את איכות הלמידה למול מרחב היפר-פרמטרים בכדי למצוא היפר-פרמטרים אופטימליים לדאטה שלנו.

שימוש: עשינו מספר ניסיונות, בכל ניסיון בחרנו וקטור מממד $3-6$ שתיאר את גודל ומספר השכבות ברשת הנוירונים שבנינו. בנוסף בחרנו אקראית (לרוב על פי ברירת המחדל) נתוני התחלה להיפר-פרמטרים נוספים (כמו למשל: גודל חבילות הלמידה, סוג הלמידה, פונקציית אקטיבציה, גודל הצעד בעדכון כל אחד מהמשקלים).

אז, הגדרנו רשת של אפשרויות קרובות לבחירה האקראית ונתנו לאלגוריתם להחליט מי לומד את הדאטה בצורה הכי טובה (עבור הפונקציה AUC). אחרי קבלת התוצאות, אם תוצאה לא הייתה בפנים המרחב שהגדרנו, הוספנו לרשת אופציות נוספות וקרובות יותר משני צדדיה והרצנו שוב.

כעת, קיבלנו תוצאה שהיא לכל הפחות מקסימום מקומי. מכיוון שעשינו זאת מספר פעמים קיבלנו כמה כאלה. עבור כל אחת, ניסינו "לנער" טיפה את התוצאות שלה בניסיון טיוב של פרמטר אחד (נתנו הרבה אופציות להיפר-פרמטר יחיד וניסינו לראות האם מתקבל פתרון טוב יותר). אחרי שקיבלנו את כלל התוצאות בחרנו את התוצאה הכי טובה וכך קיבלנו את ההיפר-פרמטרים הכי טובים עבור הדאטה והבעיה שלנו באמצעות שימוש ב-*Validation data*.

- *Test data*:

כמות: 20% ממרחב הדוגמאות (5,651).

תיאור: הדאטה שעליו בדקנו את איכות הלמידה הסופית שלנו, בשביל שנוכל לדרג עד כמה הלמידה הייתה טובה.

שימוש: אחרי שקיבלנו את ההיפר-פרמטרים ניסינו לחזות, עבור דוגמאות שאותם המודל לא ראה וכך יכלנו לתת דירוג סופי למודל שלנו וליכולת שלו לפתור את הבעיה (כי הבעיות שנתנו לו כעת אינן תלויות במה שהוא מכיר ולכן התוצאה שנקבל היא אובייקטיבית)

6. בעיית ה-*Over fitting*:

- תיאור הבעיה:

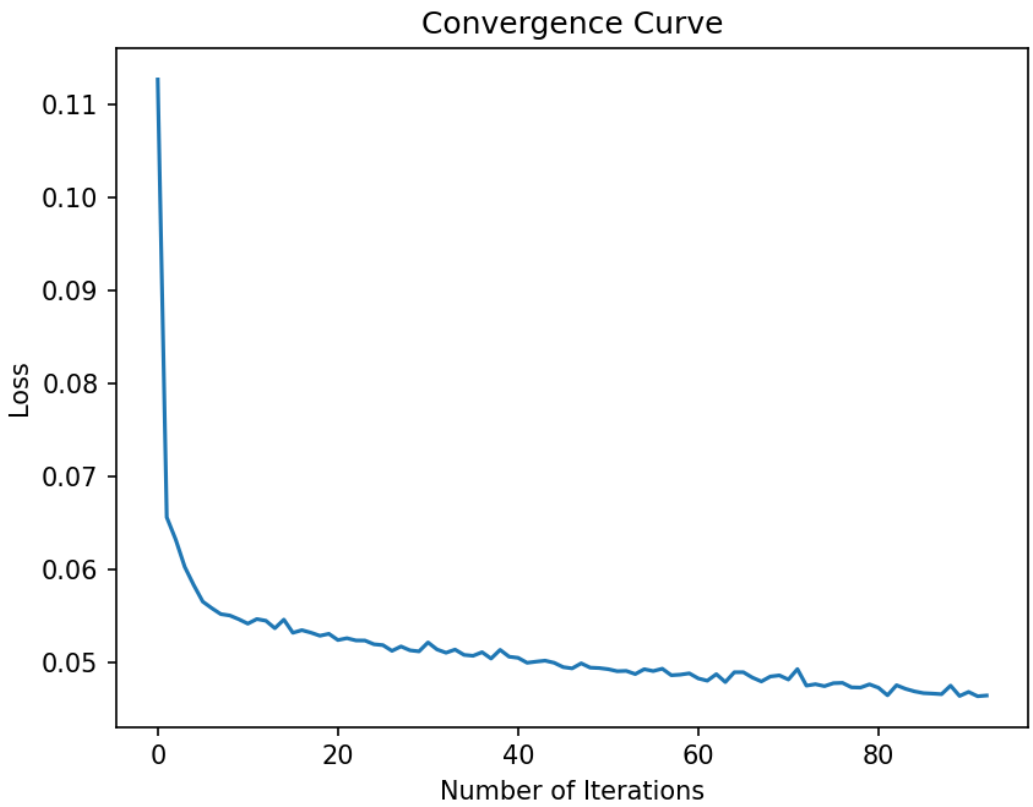
הבעיה מתארת מצב שבו המודל לומד את הנתונים האימון שלו בצורה כל כך טובה, שהוא מתחיל ללמוד גם את השגיאות והרעש שבנתונים. זה מוביל לכך שהמודל אינו מסוגל להכליל את הידע שלו לנתונים חדשים, והוא מתחיל להציג ביצועים ירודים על נתונים שלא ראה במהלך האימון.

- זיהוי הבעיה

ניתן לזהות שהבעיה נוצרה ע"י התבוננות בגרף של ביצועי המודל. אם ביצועי המודל מתחילים להידרדר לאחר מספר מסוים של איטרציות, זה עשוי להיות סימן לכך שהמודל סובל מ-*Over fitting* .

- במודל שלנו:

בתחילה, חשבנו שמתקבלת בעיה כזו אצלנו, כיוון ששימוש ב-$early\ stopping$ שיפר במקצת את התוצאות (ב-0.02 במדד הAUC). אך לאחר אופטימיזציה של ההיפר-פרמטרים הפער הפסיק להתקיים (ואף היה הפוך). בנוסף לפי בדיקת איכות הלמידה על קבוצת ה-$test$, ניתן לראות שפונקציית הסיכון שלו שואפת ל-0 ואינה נפגעת במהלך האיטרציות, כלומר במודל שלנו בעיית ה-$Over\ fitting$ לא מתקיימת.



7. <u>היפר-פרמטרים הכי טובים לדאטה שלנו</u>:

hidden_layer_sizes	מספר השכבות וכמה נוירונים? בכל שכבה	(42, 35, 13, 12, 6)
max_iter	מס' איטרציות מקסימלי, הגבלה לא עזרה	500
early_stopping	האם לעצור לפני סיום למידת כלל הדוגמאות, אם יש חשש ל-*Over fitting*?	*False*
n_iter_no_change	אם *early_stopping* עובד	10
batch_size	מספר הדוגמאות בכל חבילת למידה	600
learning_rate	סוג התקדמות הלמידה	*adaptive*
learning_rate_init	קצב התקדמות הלמידה למול הסוג	0.02
activation	פונקציית האקטיבציה	*ReLU*

בשביל למצוא את ההיפר-פרמטרים הכי טובים לדאטה התייחסנו לזה כבעיית מציאת מקסימום. בחרנו מספר אופציות של חלוקה לשכבות של רשת הנוירונים (מגדלים שונים), ועבור כל אחת מהן הגדרנו היפר-פרמטרים נוספים (באקראי).

לימדנו את התוכנית באמצעות ה-$training$ (72%), וכמו שתיארתי קודם, הרצנו על מאין סביבה שלהם (רצנו מספר רב של פעמים על כל אופציה קרובה; מצורפת דוגמה בסוף) את ההיפר-פרמטרים על ה-$validation$ (8%) וחיפשנו מתי מתקבל בה מקסימום על הלימדה של ה-$test$ (20%).

אם לא קיבלנו מקסימום בפנים של הערכים (כלומר, קיבלנו שהמקסימום הוא בערכי הקצה), הזזנו טיפה את מרחב הערכים והצבנו סביבה נוספת. אם קיבלנו את המקסימום בפנים, הצבנו ערכי קרובים יותר למקסימום הזה בשביל לדייק אותו (בעזרת "שיטת החצייה").

ואחרי זמן מה (לפעמים זה רץ במשך כל הלילה על כמה אלפי קומבינציות של סביבת ההיפר-פרמטרים שהגדרנו) הגענו למקסימום לוקלי. מכיוון שהמקסימום הוא רק לוקלי, עשינו את זה עבור מספר אופציות, וכך הגענו למספר "נקודות" מקסימום (שחלקן אפילו התכנסו לאותו מקסימום). ואז בחרנו את הטוב מכולם וכך הגענו להיפר-פרמטרים טובים מאד.

דוגמא ל"סביבה" של היפר-פרמטרים:

```
hidden_layer = [(i, j, k, l, m, n) for i in range(60, 64) for j in range(36, 40) for k in range(11, 15)
for l in
range(8, 15) for m in range(3, 8) for n in range(3,12)]
grid = {
   "hidden_layer_sizes": hidden_layer,
   "batch_size": [130, 135, 140],
   "solver": ["adam", "sgd"],
   "learning_rate": ["constant", "adaptive"],
   "validation_fraction": [0.2, 0.15, 0.25],
   "learning_rate_init": [0.01, 0.02, 0.005],
   "n_iter_no_change": [10, 7, 15],
   "activation": ["relu", "logistic, tanh"],
}
```