פרויקט לימוד מכונה

חלק ב

ירדן עיני 204371082

פלג אליהו 318356995

Decision Tree

1. לאחר שחילקנו את הנתונים ל-90% סט אימון ולסט ו-10% סט ולידציה, בנינו עץ החלטה מלא באמצעות סט האימון ובחנו את אחוזי הדיוק המתקבלים על סט האימון וסט הוולידציה. ניתן לראות כי עבור סט האימון קיבלנו ערך דיוק של 1, כלומר המודל הגיע למצב של Over-Fitting, ועבור סט הוולידציה קיבלנו דיוק של 0.898. ניתן לראות כי בסט הוולידציה אחוזי הדיוק שהתקבלו מעט יותר נמוכים מאשר של סט האימון, וזאת מאחר והמודל לא אומן על סט הוולידציה ולכן מניב שם תוצאות פחות טובות.

בנוסף, בדקנו את מדד ה-f1 וקיבלנו ערך של 1 על סט האימון ו-0.6124 על סט הוולידציה. בחרנו להסתכל על מדד זה מאחר וערכי משתנה המטרה שלנו לא מאוזנים, ו-f1 נחשבת מטריקה טובה עבור נתונים לא מאוזנים. מדד זה שומר על האיזון בין ה-precision לבין ה-recall ונהוג להשתמש בו כאשר התפלגות המחלקות (נכס בסיסי ונכס יוקרה) אינה מאוזנת. ניתן לראות שגם עבור מדד זה קיבלנו ערך גבוה יותר בסט האימון.

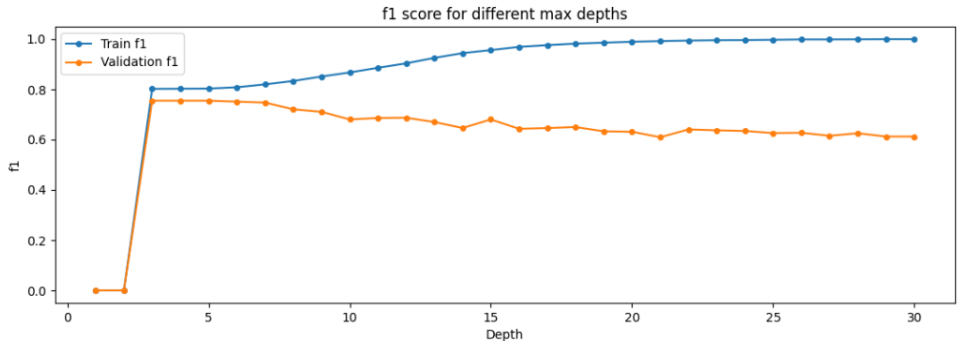
נספח 1 - גרף העץ המלא

```
Full tree on training set:
Accuracy: 1.0000
f1: 1.0000
Full tree on validation set:
Accuracy: 0.8980
f1: 0.6124
```

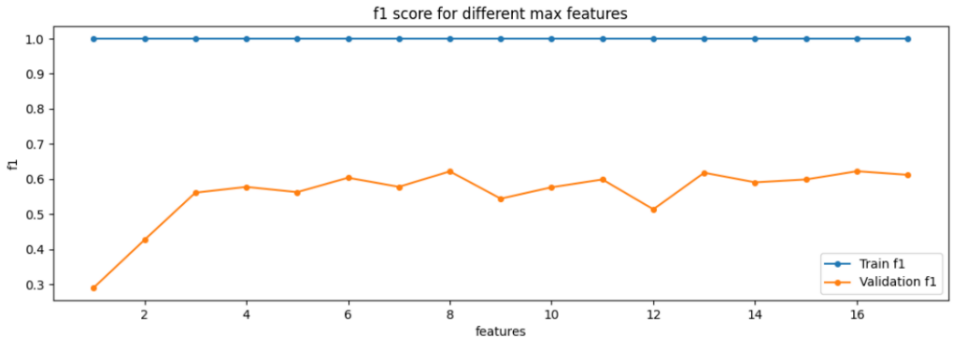
2. כוונון פרמטרים:

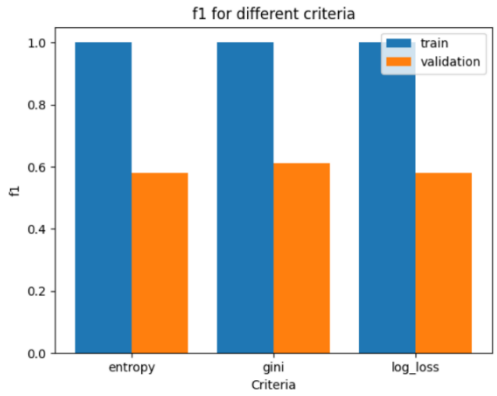
בחרנו לכוונן את היפר-הפרמטרים עומק העץ, הקריטריון לפיו מתבצעת החלוקה בעץ ומספר הפרמטרים במודל.

Max_depth – היפר-פרמטר זה קובע את עומק העץ עבור המודל. עבור עמקי עץ גדולים מידי, המודל עלול להגיע למצב של over-fitting, ועבור עמקי עץ קטנים מידי המודל יהיה במצב של under-fitting. לכן, על מנת לשלוט בשני ההיבטים הללו, בחרנו לכוונן את הפרמטר, ובחרנו לבחון את הטווח שבין [1,30], כלומר בין עץ בעל צומת החלטה אחד לבין עומק העץ המלא שקיבלנו. ניתן לראות כי עד לאזור של עומק עץ של 6 ציון ה-f1 של סט האימון והוולידציה עולים יחסית ביחד, ולאחר מכן סט האימון ממשיך לעלות ומגיע למצב של over-fitting וה-f1 של סט הוולידציה מתחיל לרדת.



Max_features – היפר-פרמטר זה קובע את מספר המשתנים אשר יוכנסו למודל. עבור מספר משתנים גדול מידי, המודל עלול להגיע למצב של over-fitting, ועבור מספר משתנים מועט המודל יהיה במצב של under-fitting. לכן, על מנת לשלוט בשני ההיבטים הללו, בחרנו לכוונן את הפרמטר, ובחרנו לבחון את השורש הריבועי של מספר הפרמטרים, לוג-2 שלהם וללא הגבלה. ניתן לראות כי עבור סט הוולידציה מקבלים ערכי f1 יחסית דומים אך תנודתיים עבור כמות משתנים גדולה מ-3.





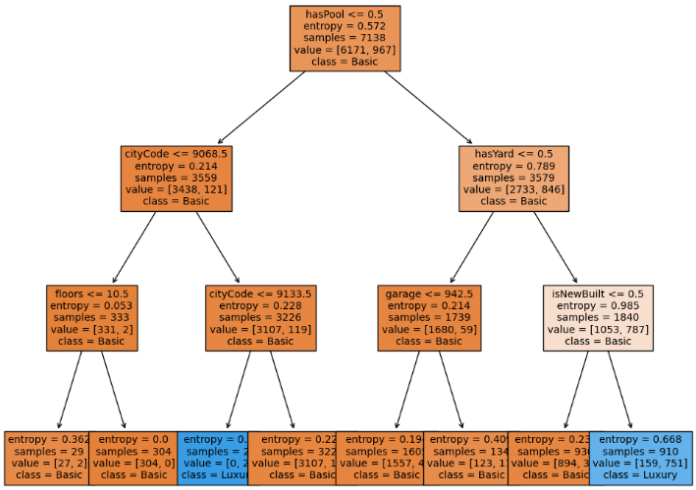
<u>Criterion</u> – היפר-פרמטר זה מהווה את קריטריון הבחירה לפיו יוכרע איזה מהמאפיינים ייבחר בכל פיצול, כך שהמאפיינים שיביאו לערך המדד הטוב ביותר ייבחרו ראשונים. נרצה לכוונן פרמטר זה על מנת לקבל את קריטריון הפיצול אשר יביא לערך ה-f1 הטוב ביותר. בחרנו לבחון את הקריטריונים entropy, gini, log_loss. ניתן לראות שערכי f1 המתקבלים עבור שלושת הקריטריונים השונים יחסית דומים, עם יתרון קל לקריטריון gini.

לאחר הכוונון בעזרת שיטת grid-search עם cross validation של 10, התקבלו הערכים הבאים: חלוקת העץ לפי קריטריון האנטרופיה, עומק עץ מיטבי 3 ומספר פרמטרים לא מוגבל.

3. לאחר קבלת הקונפיגורציה הנ"ל מה-grid-search, כיוונו עץ החלטה עם הקונפיגורציה שהתקבלה. עבור קונפיגורציה זו קיבלנו ציון f1 של 0.80149 על סט האימון ו- 0.755102 על סט הוולידציה. ניתן לראות כי קיבלנו הבדלים משמעותיים בין התוצאות החדשות עבור המודל שהתקבל, לבין המודל מסעיף 1, הן עבור סט האימון והן עבור סט הוולידציה.

ציון ה-f1 בסט האימון ירד מ-1 ל-0.80149, וזאת מפני שלא הגענו לעץ מלא ונמנענו ממצב של over-fitting. כמו כן, ערך ה-f1 של סט הוולידציה עלה מ-0.6124 ל-0.755102. עץ ההחלטה המלא שקיבלנו יצא התאמת יתר עבור סט האימון, דבר שלא בהכרח תואם את סט הוולידציה. על כן, שימוש ב-cross validation ב-grid search מאפשרת חלוקה שונה בכל פעם של סט האימון, ובכך מתאימה את המודל לסט נתונים עתידי שאינו ידוע.

גרף העץ שהתקבל:

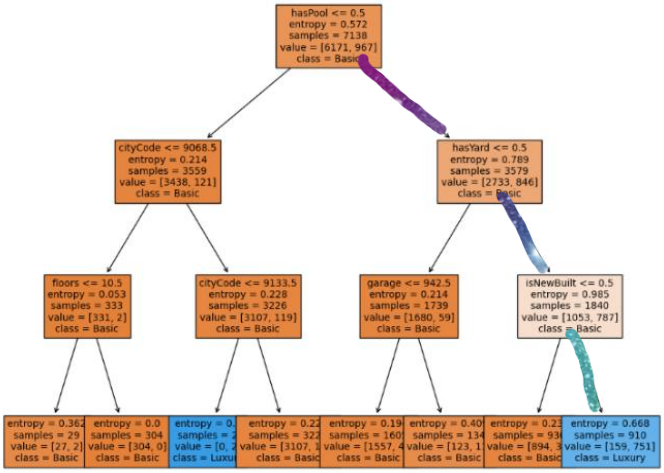


על מנת לנסות לשפר את המודל, ניסינו לאזן את סט האימון בעזרת פונקציית SMOTE, כך שיכיל מספר דומה של דירות בסיס ויוקרה ועליו להריץ grid search למציאת הפרמטרים האופטימליים לעץ ההחלטה שיאומן על סט האימון המאוזן. לאחר איזון סט האימון והרצת ה-grid-seach, התקבלו הפרמטרים הבאים:

criterion= 'entropy', max_depth= 6, max_features= None, min_samples_split= 2, splitter= 'best'

לאחר קבלת הפרמטרים האופטימליים, הרצנו עץ החלטה עם הפרמטרים הללו על סט הוולידציה, וקיבלנו עבורו ערך f1 של 0.7512, תוצאה נמוכה במעט מזו שהתקבלה בכוונון הפרמטרים הקודם, לכן נשאר עם העץ הקודם.

```
sm = SMOTE(random_state=42)
X_res, y_res = sm.fit_resample(X_train, y_train)
```



העברת רשומה בעץ

נבחן את רשומה מספר 4 בעץ שקיבלנו. עבור רשומה זו, יש בריכה ולכן בפיצול הראשון ערך המשתנה של הבריכה הוא 1 ולכן פנינו לצד ימין כי הוא לא קטן מ-0.5. לרשומה זו יש גינה (1) ולכן נפנה לצד ימין כי זה גדול מ-0.5. לבסוף, לרשומה זו ערך הבנייה החדשה הוא 1 (גדול מ-0.5) ולכן נפנה ימינה ונקבל כי העץ יסווג רשומה זו כנכס יוקרה, ואכן היא מוגדרת כנכס יוקרה בסט הנתונים.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1		Unnamed:	squareMet	numberOf	hasYard	hasPool	floors	cityCode	cityPartRa	numPrevO	made	isNewBuilt	hasStormP	basement	attic	garage	hasStorag	hasGuestR	price	category
2	0	0	82948	95	0	0	71	25098	5	7	1992	1	0	7708	1456	429	0	8	8299141	Basic
3	1	1	91757	82	0	0	78	92525	8	4	2012	1	1	5258	1106	628	0	6	9182239	Basic
4	2	2	55757	6	0	1	51	19826	9	10	2020	0	0	8020	4350	339	1	9	5578966	Basic
5	3	3	10683	79	0	0	50	78265	2	6	2005	0	0	8178	5473	195	0	5	1071101	Basic
6	4	4	36904	99	1	1	68	41334	1	9	2020	1	0	4683	5461	527	0	4	3698721	Luxury

Feature importance

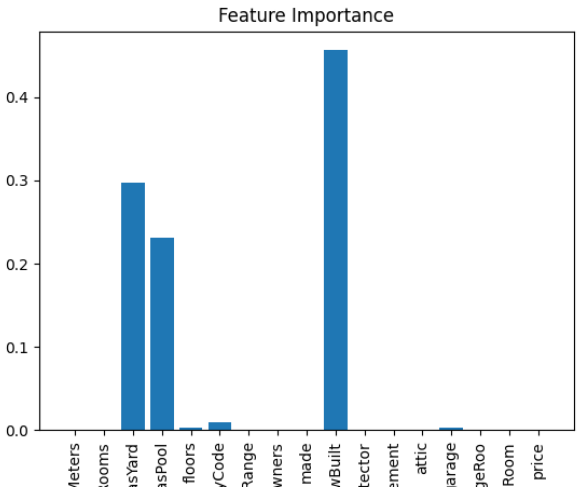
ע"י התבוננות במבנה העץ, ניתן לזהות את המשתנים אשר משפיעים על החלטת הסיווג וההפרדה בין המחלקות לפי קרבתם לשורש העץ. ניתן לראות כי משתנה הבריכה מפריד בצורה הטובה ביותר בין נכסים בסיסיים לנכסי יוקרה. במידה וקיימת בריכה, המשתנה אשר מפריד בצורה הטובה ביותר הוא משתנה הגינה, ובמידה ואין בריכה זהו משתנה המיקוד, וכך הלאה כאשר יורדים בעומק העץ.

בניגוד להשערתנו טרם בניית עץ ההחלטה, ניתן לראות כי חשיבות משתנה המיקוד יחסית גבוהה. מעץ ההחלטה שקיבלנו ניתן להבין על בעיית סיווג הנכסים בפריז כי לעצם קיומם של בריכה וגינה בנכס יש השפעה על סיווגו כנכס יוקרה.

לאחר שימוש בפונקציית חשיבות המשתנים של המודל, הפקנו את הגרף והטבלה הבאים. ניתן לראות כי משתנה הבנייה החדשה קיבל את החשיבות הגבוהה ביותר, בניגוד לעץ שקיבלנו, בו משתנה הבריכה נמצא בשורש העץ. כמו כן ניתן לראות כי משתנה הבריכה והגינה ממוקמים במקומות יחסית גבוהים.

המשתנה המשפיע ביותר בעץ הוא משתנה הבריכה. הדבר מתיישב עם האינטואיציה שלנו מאחר ונכסים עם בריכה לרוב נחשבים יותר אקסקלוסיביים וסביר שיסווגו כנכסי יוקרה. לעומת זאת, משתנה המיקוד קיבל חשיבות יחסית גבוהה בעץ בניגוד לאינטואיציה שלנו, מאחר ולא הנחנו כי לערך המספרי של המיקוד, שעל פניו לא מייצג משהו פיזי, קיימת חשיבות גדולה בקשר ליכולת הסיווג בין סוגי הנכסים. ייתכן ובסט הנתונים הזה קיימת משמעות לערך המיקוד עצמו, וייתכן כי הוא מצביע על אזורים שונים בעלי רמת יוקרתיות שונה. משתנה הבנייה החדשה נמצא במקום נמוך בעץ ולכן חשיבותו בעץ יחסית נמוכה, בניגוד לאינטואיציה שלנו מאחר ואנו מניחים כי נכס שנבנה בבנייה חדשה יהיה עמיד, חדיש ומכאן יוקרתי יותר. בנוסף, ציפינו שמשתנה המחיר יכנס לעץ ההחלטה במקום יחסית גבוה, מאחר ואנו מניחים כי למחיר קשר חזק עם סיווגו של נכס כנכס יוקרתי, וכי הוא יכול לזהות נכסי יוקרה בצורה טובה. לעומת זאת, משתנה זה לא נכנס לעץ ההחלטה האופטימלי שקיבלנו, ולכן חשיבותו כנראה נמוכה ביחס למשתנים שנכנסו. ייתכן ובסט הנתונים הזה למחיר אין השפעה גדולה על סווג הנכס כיוקרתי, וייתכן כי המשתנים שמשפיעים על יוקרתיות הנכס הם יותר מאפייני הנכס הפיזיים (גינה, בריכה), יותר מאשר העלות שלו.

	Features _names	Importance
9	isNewBuilt	0.456742
2	hasYard	0.296979
3	hasPool	0.231689
5	cityCode	0.008707
4	floors	0.003312
13	garage	0.002572
0	squareMeters	0.000000
11	basement	0.000000
15	hasGuestRoom	0.000000
14	hasStorageRoo	0.000000
12	attic	0.000000
8	made	0.000000
10	hasStormProtector	0.000000
1	numberOfRooms	0.000000
7	numPrevOwners	0.000000
6	cityPartRange	0.000000
16	price	0.000000

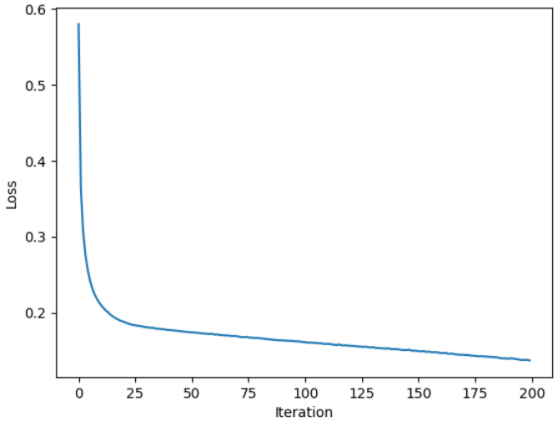


Neural Network

לפני הרצת רשת הנוירונים, ביצענו נרמול לנתונים מסוג standard scaler על מנת שכלל המאפיינים יהיו באותו טווח ערכים של התפלגות נורמלית סטנדרטית עם תוחלת 0 ושונות 1.

1. הרצנו רשת נוירונים עם ערכי ברירת המחדל: 17 נוירונים בשכבת הכניסה (כמספר המשתנים במודל), שכבה חבויה אחת, hidden_layer_size=100, learning_rate_init=0.001, max_iter=200, activation=relu.

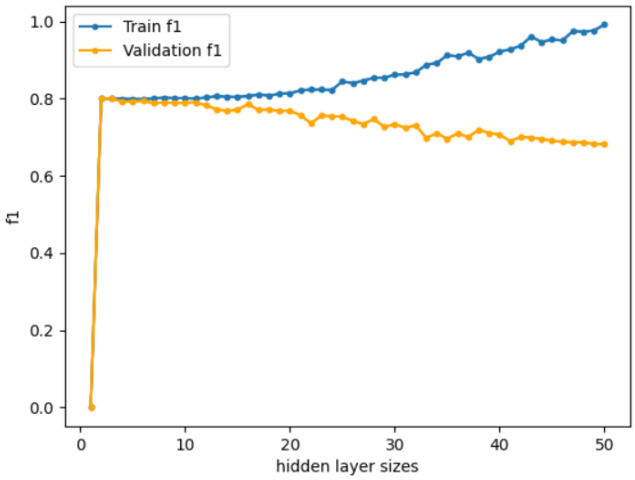
לאחר הרצת הרשת עם ערכי ברירת המחדל, קיבלנו ערך f1 של 0.818 עבור סט האימון ו-0.739 עבור סט הוולידציה. ניתן לראות כי ערך ה-f1 של סט האימון גבוה יותר מאשר זה של סט הוולידציה וזאת מאחר והמודל אומן על סט האימון ולכן מגיע עבורו לתוצאות טובות יותר בהשוואה לסט הוולידציה, אותו הוא פוגש לראשונה. המודל לא הגיע ל-over fitting עבור סט האימון מאחר והמודל מוגבל למספר איטרציות מקסימלי של 200, לכן אם היינו מגדילים את מספר האיטרציות, היינו מצמצים את השגיאה ובכך מגיעים לערך f1 גבוה יותר. כמו שניתן לראות בגרף זה, כאשר מספר האיטרציות הוא 200, השגיאה אינה אפסית ולכן המודל לא נמצא ב-over-fitting, ואכן ככל שמספר האיטרציות גדל, השגיאה קטנה.



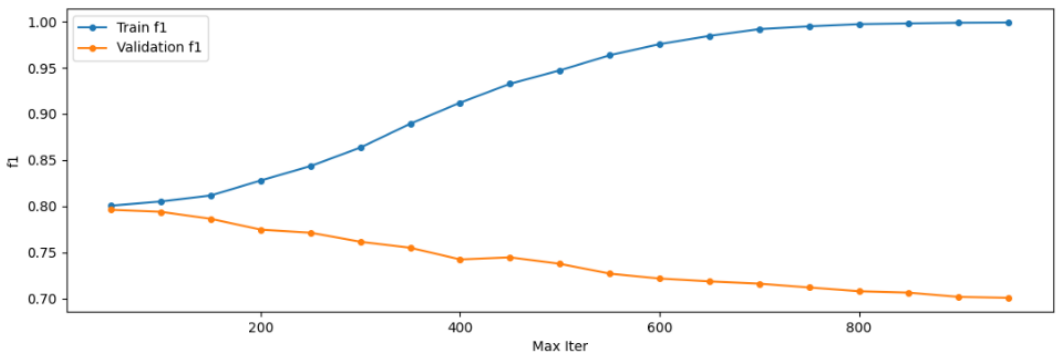

```
MLPClassifier-Train f1: 0.818281
MLPClassifier-Validation f1: 0.739583
```

2. <u>כוונון פרמטרים</u>

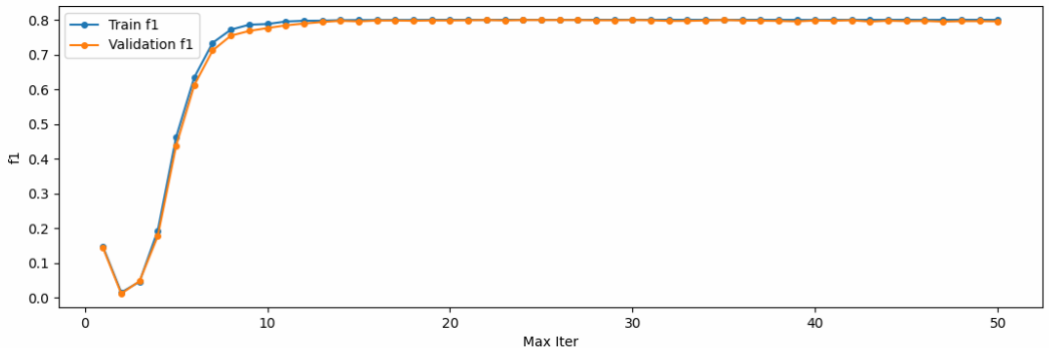
<u>Hidden_layer_size</u> – היפר פרמטר זה מגדיר את מבנה רשת הנוירונים, כמה שכבות יהיו וכמה נוירונים יהיו בכל שכבה חבויה. ככל שמספר השכבות והנוירונים בכל שכבה גדול יותר, כך גדלה היכולת לפתור בעיות מורכבות יותר, אך מנגד זמן הריצה מתארך ועלולים להגיע למצב של Over-fitting. יצרנו גרף המציג את ערך ה-f1 כתלות במספר הנוירונים בשכבה, כאשר קיימות שתי שכבות בעלות מספר נוירונים זהה, בין 1-50. ניתן לראות כי ככל שעולים במספר הנוירונים בכל שכבה, סט האימון מתקרב למצב של over-fitting אך מנגד, ערך ה-f1 של סט הוולידציה דועך, והוא מקבל את ערכו המירבי עבור בין 2-10 נוירונים בכל אחת משתי השכבות.



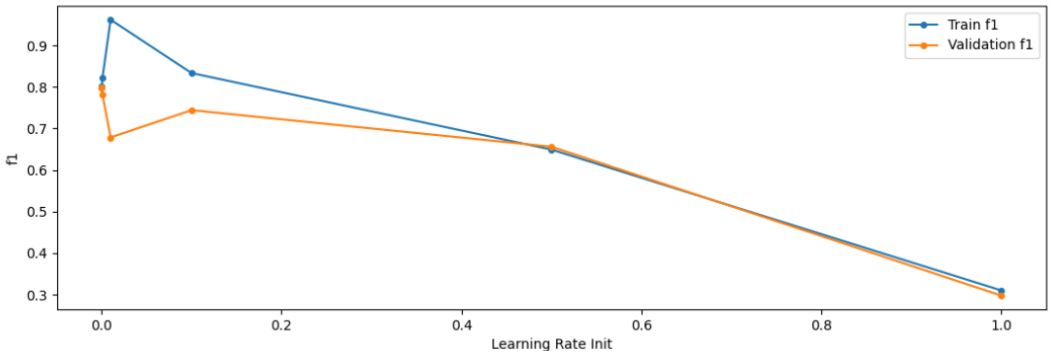
<u>Max_iter</u> – היפר פרמטר זה קובע את מספר האיטרציות המקסימלי שנותן למודל לתקן את עצמו ולהקטין את השגיאה. ככל שערך זה גדול יותר, השגיאה עשויה לקטון ולשפר את המודל, אך מנגד, יעלה את זמן הריצה ועבור ערך גדול מידי להגיע למצב של Over fitting. יצרנו גרף אשר מציג את ערך ה-f1 כתלות במספר האיטרציות שהמודל מבצע, כאשר בחנו טווח של בין 50-800 איטרציות בקפיצות של 50.

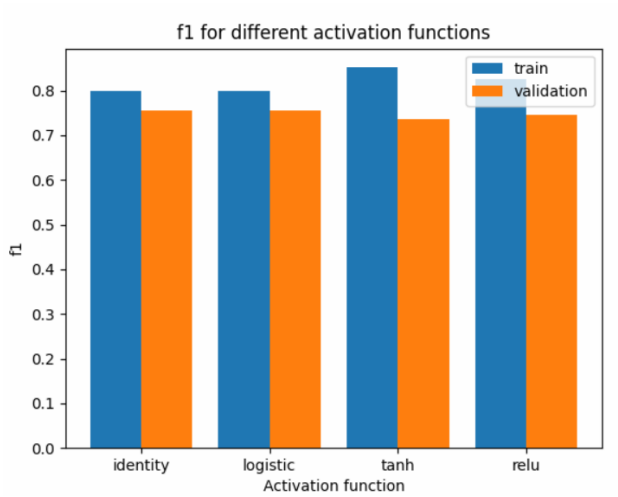


ניתן לראות כי ערך ה-f1 הטוב ביותר בסט הולידציה התקבל דווקא במספר איטרציות מקסימלי הקרוב ל-50, לכן נפיק גרף נוסף אשר יציג את ערך ה-f1 עבור מספר איטרציות מקסימלי בין 0-50. מגרף זה ניתן לראות כי מספר האיטרציות המקסימלי אשר מביא לערך ה-f1 הטוב ביותר בסט הוולידציה הוא באזור ה-50. יש לציין כי מדובר בגרף הבוחן את השפעת משתנה זה בלבד, ועל כן זהו אינו בהכרח הערך המיטבי של הפרמטר, אשר יתקבל בעזרת grid-search.

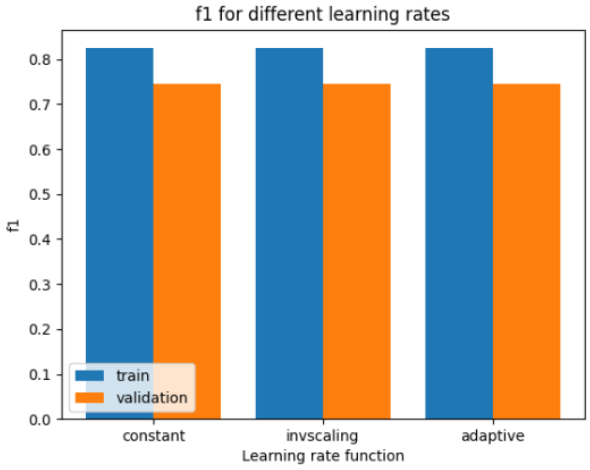


<u>Learning rate init</u> – היפר פרמטר זה קובע את קצב הלמידה ואת מהירות השינוי של ערכי המשקולות בהתאם לטעות הנאמדת. ערך גבוה מידי של קצב הלמידה יוביל לשינויים גדולים בערכי המשקולות, דבר שעשוי להוביל להגעה מהירה יותר לערך האופטימלי, אך מנגד, מעלה את הסיכוי לפספס את הערך האופטימלי של המשקולות אשר יביא לשגיאה הקטנה ביותר. בחרנו לבדוק את קצב הלמידה בקפיצות של מספר סדרי גודל בכל פעם, כאשר נבדקו הערכים [0.0001, 0.0001, 0.001, 0.01, 0.1, 0.5, 1]. ניתן לראות כי עבור סט הוולידיציה, קצב הלמידה אשר הביא לערך ה-f הגדול ביותר מבין הערכים שנבדקו הוא 0.0001, כלומר קיבלנו את התוצאה הטובה ביותר בקצב למידה איטי יחסית.





<u>Activation</u> – היפר פרמטר זה מייצג את פונקציית האקטיבציה המתבצעת בנוירונים בשכבות החבויות וממירה את הקלט לפלט. בחרנו לבחון את פונקציות האקטיבציה 'identity', 'logistic', 'tanh', 'relu'. המוטיבציה בכוונון פרמטר זה היא שטרנספורמציות שונות עשויות להשפיע בצורה שונה על תהליך הלמידה, ולכן נרצה לבחור את הטרנספורמציה אשר תביא לערך ה- f1 הטוב ביותר. מהגרף ניתן לראות כי פונקציות האקטיבציה אשר הביאו לערכי ה-f1 הגבוה ביותר בסט הוולידציה הן identity ו-logistic.



<u>Learning rate</u> – היפר פרמטר זה יוצר חיפוש עבור קצב הלמידה במרחב ערכים המשתנה לפי פונקציות שונות המופעלות על ה-learning rate init. המוטיבציה בכוונון פרמטר זה היא למצוא את פונקציית חיפוש קצב הלמידה אשר תמצא את קצב החיפוש האידיאלי. הערכים שנבחנו הם 'constant', 'invscaling', 'adaptive'. מהגרף ניתן לראות כי ערכי f1 שהתקבלו בסט הוולידציה ובסט האימון יחסית דומים בין הlearning rates השונים.

על מנת לבחור את הקונפיגורציה המיטבית עבור מודל רשת הנוירונים, הרצנו grid search עם ערכי הפרמטרים הבאים:

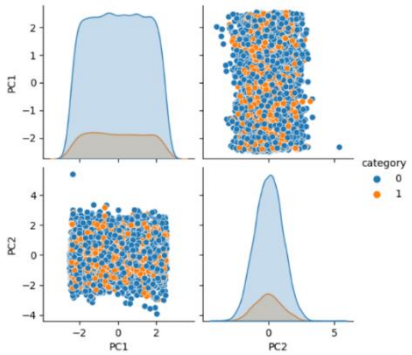
```
#grid search
hiddenLayer = [(5, ), (10, ), (11, ), (12, ), (9, ), (8, ), (15,), (20, ), (5, 5), (10, 10), (15, 15)]

# grid search params
param_grid = {'hidden_layer_sizes': hiddenLayer,
              'max_iter': [50, 100, 200, 500],
              'learning_rate_init': [0.0001, 0.0001, 0.001, 0.1],
               'learning_rate': ['constant', 'invscaling', 'adaptive'],
              'activation': ['identity', 'logistic', 'tanh', 'relu']
              }
```

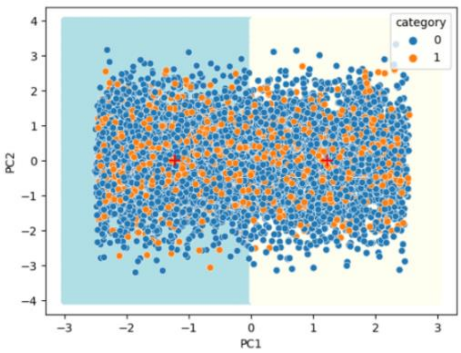
לאחר הכוונון בעזרת שיטת grid-search, התקבלו הערכים הבאים: שכבה אחת עם 10 נוירונים, מספר איטרציות מקסימלי: 200, קצב הלמידה ההתחלתי: 0.0001, פונקציית אקטיבציה: 'relu', learning rate: 'constant'. עבור קומבינציית הפרמטרים הללו קיבלנו ערך f1 של 0.795 בסט האימון ו-0.746 בסט הוולידציה. ניתן לראות שערך ה-f1 בסט הוולידציה השתפר בהשוואה למודל מסעיף 1. ניסנו לשפר את המודל בכך שאימנו את הרשת על גם על סט הנתונים המאוזן שהפקנו, אך ערך ה-f1 לא השתפר.

```
Train f1: 0.795038
Validation f1: 0.746114
```

k means



1. הרצנו מודל k means המחלק את התצפיות לאשכולות בלמידה לא מונחית עם ערכי ברירת המחדל (המאשכל הבסיסי) ושני אשכולות, בהתאם לבעיית הסיווג שלנו (דירת יוקרה/ דירה בסיסית).
תחילה, הרצנו את אלגוריתם PCA על מנת שנוכל להציג את הנתונים בתרשים ולהבין את הקשר בין משתנה המטרה לבין השונות שהמאפיינים החדשים מייצרים. בתרשים ניתן לראות את פיזור משתנה המטרה (דירות יוקרה ובסיס) ביחס לשני הגורמים הראשיים העיקריים שהתקבלו מהרצת ה-PCA.

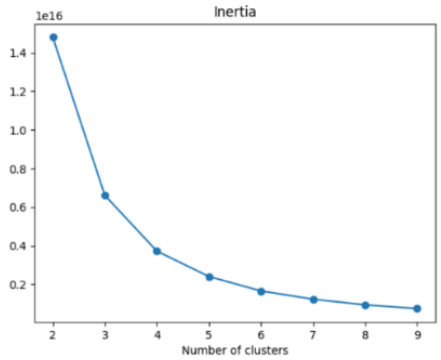


2. מאחר ואלגוריתם k-means מחזיר את האשכול של שתי הקבוצות אך אנו לא יודעים איזו מבין הקבוצות מייצגת את קבוצת ה"basic" ואיזו את קבוצת ה"luxury", בדקנו את ערכי הf1 המתקבלים על סט הנתונים, פעם אחת כאשר קבוצה 0 באשכול מייצגת את הדירות הבסיסיות וקבוצה 1 את דירות היוקרה, ופעם אחת ההיפך. ניתן לראות שעבור האופציה השנייה קיבלנו ערך f1 מעט גבוה יותר ביחס לאופציה הראשונה. על כן, ניקח את האשכול השני שהתקבל, אשר הביא לביצועים הטובים ביותר על סט הנתונים.

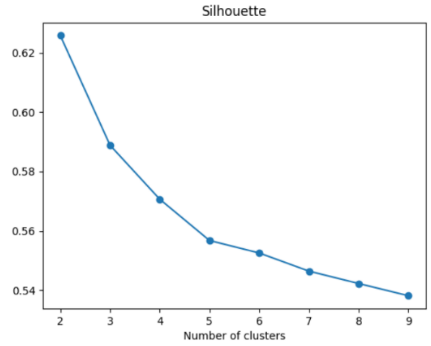
```
0 = Basic, 1 = Luxury:
f1: 0.210673

0 = Luxury, 1 = Basic:
f1: 0.21581
```

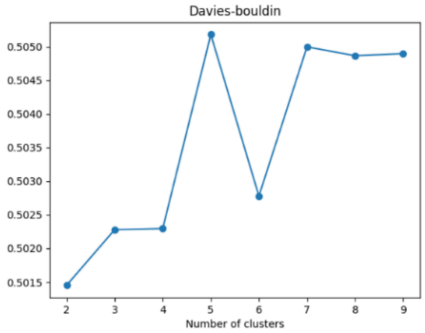
3. בסעיף זה הרצנו k means שמונה פעמים, בטווח שבין 2 אשכולות ל-10 אשכולות. עבור כל אחת מההרצות, שמרנו את ציוני המדדים interia, shihoutte, davies-bouldin והפקנו את הגרפים הבאים המציגים את ציון המדדים בהתאם למספר האשכולות המשתנה.



<u>Interia</u> – מדד זה מחשב את ריבוע סכום המרחקים של התצפיות למרכז האשכול קרוב ביותר. במדד זה יש trade-off בין מספר האשכולות לבין ערך המדד, וכאשר מספר האשכולות גדל, ערך המדד קטן. לפי מדד זה, נרצה לבחור את מספר האשכולות אשר מביא לערך מדד נמוך אך ללא הגדלה משמעותית במספר האשכולות, כלומר נרצה למצוא את הנקודה בה הירידה בערך המדד מתחילה להאט. נראה כי לפי מדד זה היינו בוחרים במספר אשכולות בין 4-5.



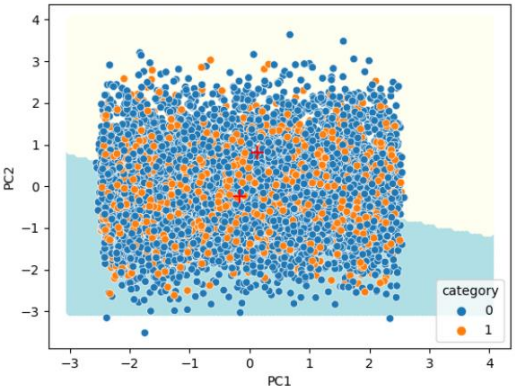
<u>Silhouette</u> – מדד זה מודד את יכולת ההפרדה והלכידות, ועד כמה תצפית דומה לאשכול שלה בהשוואה לאשכולות אחרים. מסיבה זו, נרצה למקסם מדד זה מאחר וערך גבוה שלו מצביע על כך שהתצפית סווגה היטב לאשכול המתאים לה ביחס לאשכולות האחרים, ובהתאם לכך נבחר על פי מדד זה בסיווג ל-2 אשכולות.



Davies-bouldin – מדד זה מודד את מדד הדמיון הממוצע של כל אשכול עם האשכול הדומה לו ביותר, כאשר הדמיון בין האשכולות מוגדר להיות היחס בין המרחקים בתוך האשכול לבין המרחקים מחוצה לו. לכן, נרצה למזער מדד זה, ונבחר ב-2 אשכולות.

על פי שלושת המדדים שבחנו, נראה כי נעדיף לסווג את התצפיות ל-2 אשכולות, מאחר וזו התוצאה שמתקבלת ב-2 מתוך שלושת המדדים, ובנוסף מדד ה-Interia אשר מביא לתוצאה אחרת נחשב לפחות מוצלח. ערך זה אכן מתקשר לסיפור המקרה שלנו, בו עלינו לסווג דירות ל-2 מחלקות, דירות יוקרה ודירות בסיסיות.

4. בחרנו להריץ מודל של k-medoids עם שני אשכולות, בהתאם לתוצאות שקיבלנו בסעיף 3. מאחר ואלגוריתם k-medoids מחזיר את האשכול של שתי הקבוצות אך אנו לא יודעים איזו מבין הקבוצות מייצגת את קבוצת ה"basic" ואיזו את קבוצת ה"luxury", בדקנו את ערכי הf1 המתקבלים על סט הנתונים, פעם אחת כאשר קבוצה 0 באשכול מייצגת את הדירות הבסיסיות וקבוצה 1 את דירות היוקרה, ופעם אחת ההיפך. ניתן לראות כי קיבלנו ערך f1 גבוה יותר בסט הנתונים בחלוקה השנייה (1= בסיס, 0= יוקרה), ולכן נתייחס אליה כאל "הסיווג" למחלקה. עבור חלוקה זו, ערך ה-f1 הוא 0.636.




```
f1 - option 1: 0.21227066482540935
f1 - option 2: 0.6369919823057782
```

ההבדל בין k-means ל-k-medoids הוא שב-k-means מחשבים ממוצעים ומשייכים לאשכולות לפי סכום המרחקים בריבוע מהממוצעים שהתקבלו, כאשר המטרה היא למזער את המרחקים בתהליך איטרטיבי. לעומת זאת, ב-k-medoids מחשבים את סכום המרחקים בין הנקודות באשכולות בתהליך איטרטיבי כך שמתקבל שהסנטרואיד הוא הנקודה בעלת המרחק הקטן ביותר לשאר הנקודות באותו אשכול. K-means מבוטא ע"י מרחק אוקלידי ולכן מכיוון שמעלים בריבוע את השגיאה, הוא תורם יותר לשגיאה עצמה, לעומת k-medoids שהוא סכום של ערכים מוחלטים ולכן הוא יותר עמיד לרעש. החיסרון של k-medoids הוא שהוא בעל זמן ריצה גדול יותר מאשר k-means.

```
K means on 2 cluster metrics:
Inertia:  110642.9155
Silhouette:  0.0836
Davies Bouldin Score:  3.1997

K medoids on 2 cluster metrics:
Inertia:  32583.6673
Silhouette:  0.0403
Davies Bouldin Score:  4.8674
```

על מנת להשוות בין שני האלגוריתמים בחלוקה ל-2 אשכולות, נשווה בין המדדים interia, shihoutte, davies-bouldin. לפי מדדי האיכות האשכול הללו, נעדיף את אלגוריתם k-means מכיוון שמדד ה-shihoutte יותר גבוה בו ומדד ה-davies-bouldin , יותר נמוך.

השוואה בין המודלים

1. בעיית זיהוי דירות היוקרה בפריז הינה בעיית סיווג, ועל כן המודלים אשר יתאימו לבעיה זו בצורה הטובה ביותר הם מודלי הסיווג בלמידה המונחית: עץ ההחלטה ורשת הנוירונים. שיטות אשכול, כמו k-means, לא מתייחסות לערך המחלקות של הרשומות ומבצעות למידה לא מונחית. שיטות אלו מנסות ליצור חלוקה למחלקות השונות על בסיס יכולת ההפרדה והקטנת השונות בתוך האשכולות, והן פחות מתאימות עבור משימות סיווג.
על מנת להשוות בין שלושת המודלים, נסתכל על מטריצות המבוכה המתקבלת עבור שלושת המודלים. ניתן לראות כי המודל בו המודל מצליח לזהות נכונה את המצב האמיתי של הדירות בצורה הטובה ביותר הוא עץ ההחלטות. עבור מודל זה, הצלחנו לזהות נכונה כ-98% מדירות הבסיס (672/686), וכ-69% מדירות היוקרה (74/108). בבעיית הסיווג שלנו, אנו רוצים להצליח לסווג נכונה את דירות היוקרה, וזהו המודל אשר מצליח לזהות אותן בצורה הטובה ביותר. מודל רשת הנוירונים מצליח לזהות דירות יוקרה במספרים דומים אך מעט נמוכים יותר, ומודל k-means טועה בסיווג עבור תצפיות רבות, ועל כן נעדיף את מודל עץ ההחלטה שקיבלנו.

מודל	F1 בסט האימון	F1 בסט הוולידציה	מטריצת מבוכה
Decision Tree	0.80149	0.755102	[[672 14] [34 74]]
Neural Network	0.79503	0.746114	[[673 13] [36 72]]
K-means	0.63699		[[330 356] [56 52]]

המודל הנבחר

המודל שבחרנו הוא עץ ההחלטה עם ערכי ההיפר-פרמטרים שקיבלנו בעזרת שיטת ה-grid-search.
הקונפיגורציה שהתקבלה עבור מודל זה: חלוקת העץ לפי קריטריון האנטרופיה, עומק עץ מיטבי 3 ומספר פרמטרים לא מוגבל. על מנת לבצע את החיזויים, העברנו את סט הבחינה את אותו תהליך העיבוד המקדים שהעברנו את סט האימון בחלק א.

		חיזוי המודל	
		0	**1**
ערכים במציאות	**0**	672	14
	1	34	74

ממטריצת המבוכה ניתן לראות כי המודל הצליח בחיזוי של 746 רשומות מתוך 794: 672 היו דירות בסיס והמודל אכן זיהה אותן כדירות בסיס, ו-74 היו דירות יוקרה והמודל אכן חזה אותן כדירות יוקרה. לעומת זאת, המודל טעה בחיזוי של 48 דירות: 14 מהן היו דירות בסיס והמודל חזה אותן כדירות יוקרה, ו-34 דירות היו דירות יוקרה והמודל חזה אותן כדירות בסיס. מתוצאות המטריצה ניתן לראות כי אחוז הפספוס בחיזוי של המודל של דירות יוקרה יותר גדול מאחוז הפספוס בחיזוי של דירות הבסיס.

נספחים:

נספח 1 – גרף העץ המלא