תוכן עניינים

הכנת הנתונים לאימון ובחינה

1. יישום מסקנות מחלק א':

- ביטלנו את הדיסקרטיזציה כדי להימנע מאיבוד מידע.
- השלמת הערך leg room service באמצעות הערך השכיח ביותר, ולא באמצעות ההתפלגות המשוערכת וזאת כדי להימנע מהכנסת רעש למודל.

2. התאמת המשתנים למודלי למידה:

כדי להכניס את המשתנים למודלי למידה עלינו להעביר אותם למשתנים מספרים:

- בינראזציה – המשתנה Gender : Female- 0, Male - 1

  המשתנה Customer Typer : disloyal Customer- 0, Loyal Customer– 1

  המשתנה Type of Travel: Personal Travel- 0, Business travel– 1

  משתנה המטרה satisfaction: neutral or dissatisfied- 0, dissatisfied – 1

- משתני דמה – באמצעות הפונקציה get_dummies של pandas הוספנו עמודה לכל קטגוריה:

  המשתנה Class הפך ל Class_Business, ,Class_Eco, ,Class_Unknown ,Class_Eco Plus.

3. חלוקה ל x ו y– חילקנו את סט האימון למסבירים ומוסברים.

4. בניית סט אימון ובחינה - באמצעות train_test_split של sklearn.

גודל הסט שבחרנו הוא 1000, כגודל סט האימון שהלקוח יבחן. בנוסף גודל סט הנתונים שלנו (8000) מאפשר לנו לקחת כמו גדולה כזו של נתונים שבאופן יחסי לסט הנתונים לא גדולה.

את הנתונים בחרנו באופן רנדומלי כדי לא ליצור השפעה בסט האימון/בחינה. כמו כן, היה חשוב לנו לבדוק שהסטים מאוזנים הן בתוך עצמם והן אחד מול השני כדי להימנע מהטיה כלשהי.

```
Train
--------
 0    0.564516
1    0.435484
```

```
Test
--------
 0    0.556
1    0.444
```

Decision Trees

1. **הכנה -** כדי לבנות את עץ ההחלטה השתמשנו DecisionTreeClassifier של sklearn.

עץ החלטה עובד רק עם מספרים, ביצענו את זה בסעיף הקודם.

כמו כן היה עלינו להגדיר את משתנה האקראיות random_state, מספר אקראי שמטרתו להבטיח חזרתיות של התוצאות.

2. **בחירת מדד ותוצאות:** בחרנו ב F1 כדי למדוד את איכות המודל שלנו, מכיוון שזה המדד שהנהוג אצל הלקוח שלנו ובנוסף כי הוא מהווה תוחלת הרמונית בין שני מדדים חשובים sensitivity ו-precision.

התוצאות:

```
Train F1: 1.0
Test F1: 0.8595
```

3. **ניתוח התוצאות:**

- ניתן להסיק מתוצאות אלה אינדיקציה ל overfitting - התוצאה על סט האימון גבוהה (1) ואילו התוצאה על סט הלא מוכר נמוכה ביחס לאימון (0.85) , כלומר המודל מתמודד היטב עם נתונים שהוא מכיר ופחות טוב עם נתונים שהוא אינו מכיר.
- המודל זקוק לכינון והתאמה – הפרש הגדול בין תוצאות סט האימון לתוצאות הבחינה מעידות על כך שהמודל זקוק לכינון על מנת לשפר את תוצאות ההכללה שלו, כלומר היכולת להתמודד עם נתונים לא מוכרים תעשה באמצעות כינון.
- עץ מלא תמיד יביא לתוצאה שכזו על סט האימון - מכיוון שמודל בצורת עץ יכול לפצל עצמו ולהתאים עצמו לסט עליו הוא מתאמן עד להפרדה מלאה.

4. **כוונון פרמטים**

תחילה, ביצענו כיוונון פרמטרים לפי GridSearchCV וזאת כדי למצוא את הקונפיגורציה האופטימלית שתמקסם את מדד הדיוק על סט האימון בצורה של K-Fold Cross Validation. עבדנו עם CV=10. בצורה זו, נמדוד את הדיוק על עשרה סטים שונים והתוצאות תתקבל לפי מקסום הממוצע.

1. **criterion:**

הקריטריון המשמש למדידת איכות הפיצול. אפשרויות:'gini, 'entropy' .

2. **max_features**:

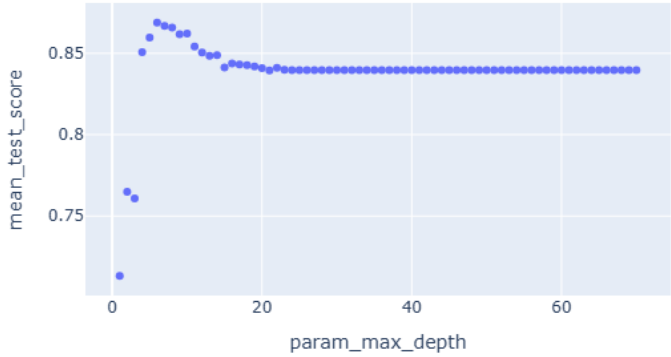
הפרמטר max_features קובע את מספר התכונות (features) שנשקלות בכל פיצול של העץ. כאשר העץ מחליט על פיצול, הוא בוחן רק תת-קבוצה אקראית של תכונות, שגודלה נקבע על ידי max_features.

שימוש ב max_features-מסייע במניעת overfitting על ידי הכנסת אקראיות לתהליך בניית העץ.

בתהליך כיוונון הפרמטרים, בחרנו ערכים שונים.

אנו מחפשים את האיזון הנכון בין מתן מספיק אפשרויות לעץ לבחור תכונות משמעותיות, לבין הגבלת הבחירה כדי למנוע התאמת יתר לנתוני האימון.

3. **max_depth** :



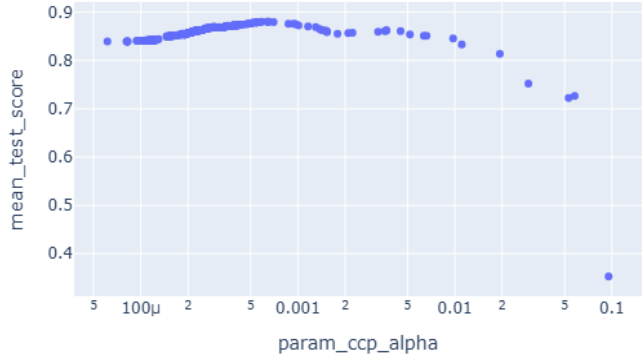
הפרמטר max_depth קובע את העומק המקסימלי של עץ ההחלטה.

הגבלת עומק העץ היא כלי חשוב למניעת overfitting, מאחר ועץ עמוק מדי נוטה "לזכור" את נתוני האימון במקום ללמוד כללים כלליים.

חיפשנו את העומק האופטימלי שמאפשר לעץ ללמוד דפוסים משמעותיים, ובמקביל להגביל את יכולתו ללימוד יתר.

כאשר התחלנו לבצע כיוונון, ביצענו מספר ניסויים ומתוך כך עלה שכאשר עומק העץ גדול מ-20, אין שינוי משמעותי בתוצאה.

4. **ccp_alpha** :



(Cost-Complexity Pruning alpha) משמש לגיזום עץ ההחלטה לאחר בנייתו. הוא מאזן בין מורכבות העץ לבין דיוקו על נתוני האימון.

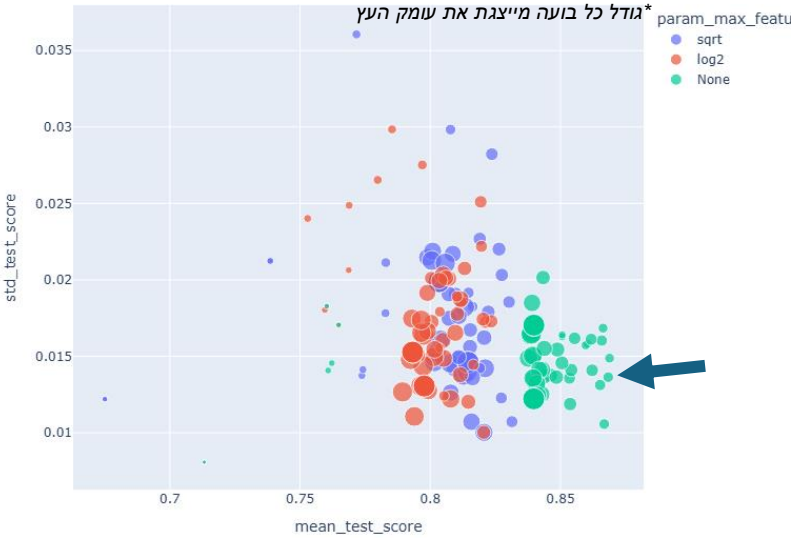
ככל שערך ה alpha-גבוה יותר, כך העץ נגזם יותר ונהיה פשוט יותר.

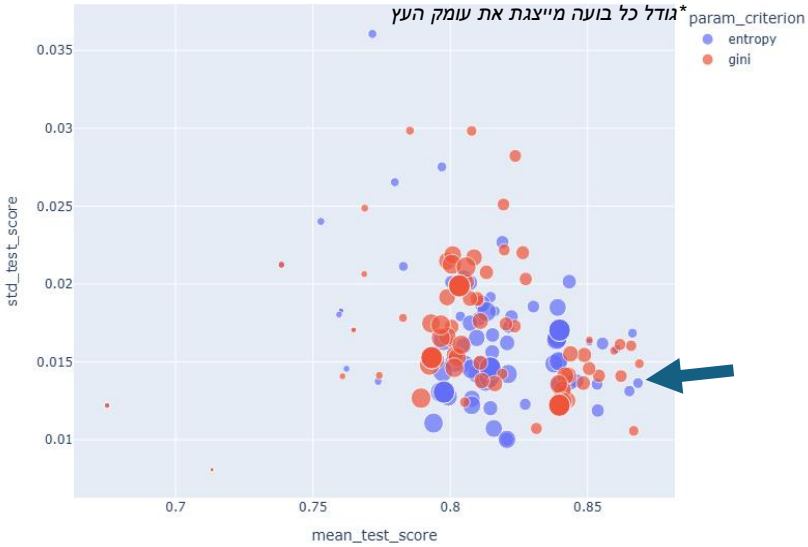
השתמשנו בפונקציה path.ccp_alphas שמחשבת רצף של ערכי אלפא שיכולים לשמש לגיזום העץ, כאשר כל ערך אלפא מייצג נקודת איזון בין מורכבות העץ לדיוקו. הפונקציה עוברת על העץ המלא מלמטה למעלה, ובכל שלב מחשבת את ערך האלפא שיגרום לגיזום החוליה החלשה הבאה.

חיפשנו את ערך ה-alpha האופטימלי שיאפשר גיזום מאוזן של העץ, כך שנשמור על דיוק מספק תוך הפחתת מורכבות העץ (דבר שיסיע למניעת התאמת יתר), נשפר את יכולת ההכללה שלו לנתונים חדשים.

במהלך חיפוש הקונפיגורציה המיטבית בחנו שתי גישות: הגבלת עומק העץ (כתנאי עצירה) העץ וקטימת העץ (לאחר סיום בניית העץ). בחרנו להציג את תוצאות ה gridsearch בתרשים בועות, המאפשר לנו להציג תוצאות עם מספר רב של מאפיינים בעת ובעונה אחת.

configuration results (Max depth &_Criterion & Max features)





מסקנות מהגרפים

- **Criterion:** מהגרף הימני ניתן לראות כי הקונפיגורציות שחולקו לפי קריטריון Gini או Entropy מעורבבות זו בזו ואין הבחנה איזה קריטריון בהכרח עדיף. מכאן הסקנו שאין השפעה ברורה של קריטריון הבחירה על התוצאה.
- **Max_depth:** משני הגרפים ניתן לחלק את גדלי הבועות לשלושה סוגים: קטן, בינוני וגדול.
  - בועות קטנות, עצים קצרים (עד עומק 5) נתנו תוצאות פחותות ביחס לשאר הקונפיגורציות וזאת בשל התאמת חוסר.
  - בועות גדולות, עצים ארוכים (עומק 10 ומעלה) שנוטים להתאמת יתר נתנו תוצאות טובות יותר מהעצים הקצרים אך לא מאלו של העצים הבינונים.
  - הבועות הבינונית אלו עם העצים בעומק (5-10) שממקומות בצד ימין תחתון נתנו את התוצאות האופטימליות הן מבחינת סטיית תקן נמוכה והן מבחינת ציון ממוצע.
- **Max_ features:** מהגרף השמאלי ניתן לראות כי ככל שהקונפיגורציה נמצאת תחת פחות הגבלות התוצאה שהיא תפיק תהיה גבוהה יותר. ניתן לראות שהקונפיגורציות מסודרות מהתוצאות הקטנות לגדולות בהתאם לאופי ההגבלה, log2, sqrt ולבסוף none.

בחירת קונפיגורציה אופטימלית

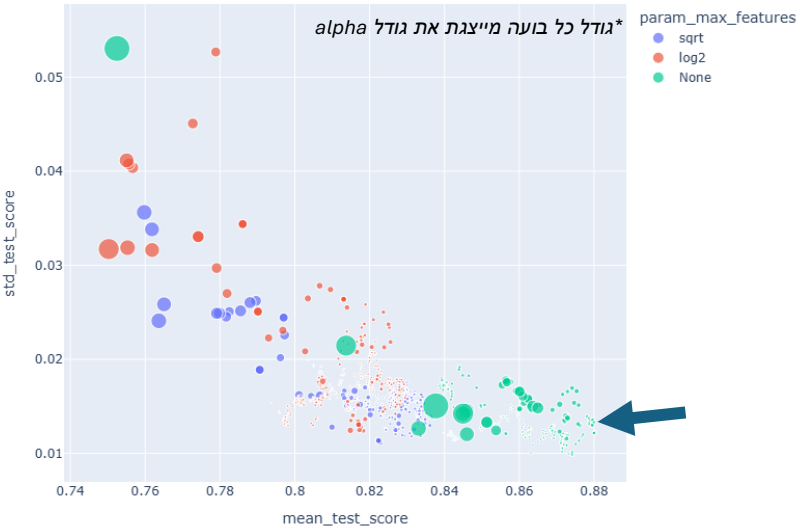
בגרפים אלו, נחפש קונפיגורציה עם סטית תקן נמוכה ככל הניתן וציון ממוצע גבוה ככל הניתן.

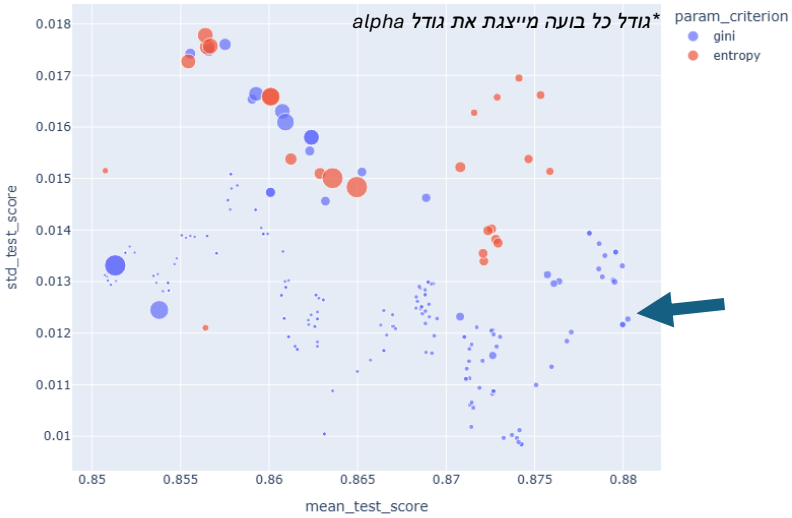
מכאן, בחנו את שלושת הקונפיגורציות הימניות תחתונות ביותר.

שמנו לב שלהבדיל מתוצאות ממוצע האימון בהם שלושת הקונפגורציות קיבלו ציון דומה, בלטה הקונפיגורציה הנבחרת בציון על סט המבחן עם הבדל יחסית גבוה ולכן נבחרה.

param_criterion=entropy
mean_test_score=0.8683455
std_test_score=0.01363899
param_max_depth=7
param_max_features=None
Test score = 0.90151

configuration results (CCP alpha & Criterion & Max features)





מסקנות מהגרפים

- **Criterion:** מהגרף הימני ניתן לראות כי הקונפיגורציות שחולקו לפי קריטריון Entropy נמצאות גבוהה על ציר הy ביחס לקונפיגורציות שחולקו לפי קריטריון Gini. כלומר, קריטריון Entropy נוטה לתת תוצאה בעל סטיית תקן גבוהה יותר מקריטריון Gini.
- **Ccp alpha:** קשה לשם לב למגמת גדלי הבועות בשני הגרפיים, עם זאת ניתן לשם לב בחלק הימני התחתון – ממוצע גבוהה וסטיית תקן נמוכה, נמצאים רק בועות קטנות בעלי alpha קטנה.
- **Max_ features**: מהגרף השמאלי ניתן לראות כי ככל שהקונפיגורציה נמצאת תחת פחות הגבלות התוצאה שהיא תפיק תהיה גבוהה יותר. ניתן לראות שהקונפיגורציות מסודרות מהתוצאות הקטנות לגדולות בהתאם לאופי ההגבלה, log2, sqrt ולבסוף none.

param_criterion=gini
mean_test_score=0.880262
std_test_score=0.01227283
param_ccp_alpha=0.00063356340877334
param_max_features=None
Test score = 0.905

בחירת קונפיגורציה אופטימלית

בגרפים אלו, נחפש קונפיגורציה עם סטית תקן נמוכה ככל הניתן וציון ממוצע גבוה ככל הניתן. מכאן, ניתן לראות שיש שני שני קונפיגרציות מועמדות. אך מכיון שההבדל בסטיית התקן הוא יחסית שולי, בחרנו את הקונפגרציה עם הממוצע הגבוהה ביותר

5. הקונפיגורציה האופטימלית:

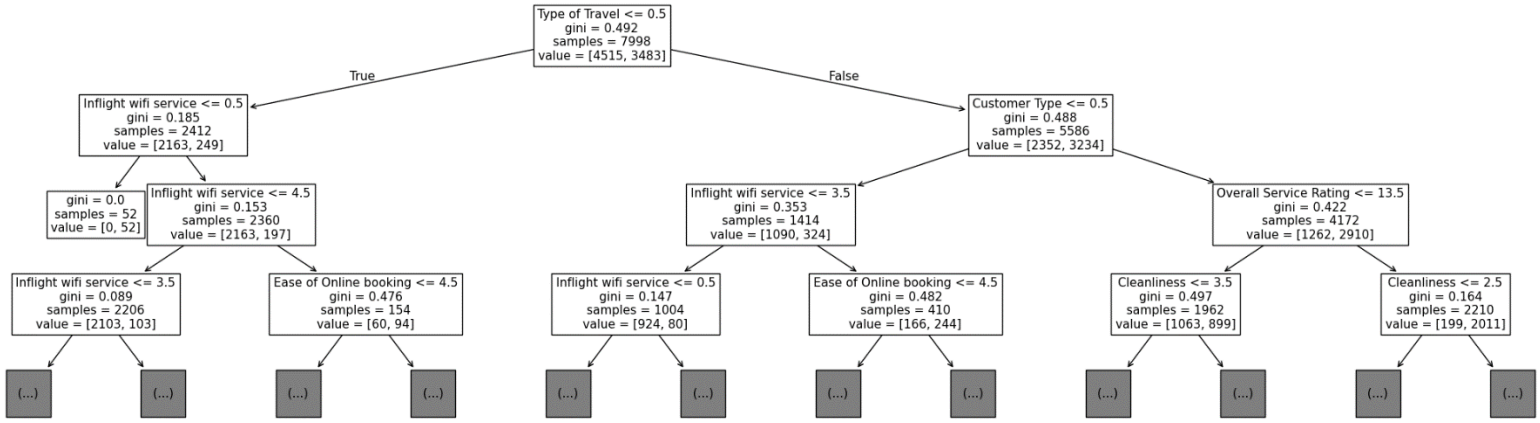
מבין שני הגישות שבחנו, בחרנו בגישה של העץ "המקוצץ" שהפיקה תוצאות טובות יותר.

```
best alpha's model parameters:
 {'ccp_alpha': 0.00063356340877334, 'class_weight': None, 'criterion': 'gini', 'max_depth': None, 'max_features': None, 'max_leaf_nodes': None, 'min_impur
ity_decrease': 0.0, 'min_samples_leaf': 1, 'min_samples_split': 2, 'min_weight_fraction_leaf': 0.0, 'monotonic_cst': None, 'random_state': 42, 'splitter':
'best'}
max depth:  10
```

```
best alpha's model Train F1: 0.89048

best alpha's Test Test F1:  0.90536
```

להלן העץ המתקבל,



מסקנות מהתבוננות בעץ:

- ה-node הראשון בעץ - התנאי הראשון שממוין על פיו כל הדגימות, הוא "Type of Travel". עובדה זו מצביעה על כך שמאפיין זה הוא המפריד המשמעותי ביותר בין התצפיות.
- מאפיינים בולטים - בשלבים הראשונים של העץ ניתן לראות את המאפיינים "Inflight wifi service" "Customer Type", "Overall Service Rating" ו-"Ease of Online booking", מיקום הראשוני מעיד על חשיבותם במיון התצפיות.
- חזרתיות - "Inflight wifi service" ו "Ease of Online booking", מופיעים במספר מקומות בעץ במספר רמות שונות, עוד אינדקציה לחשיבותם במודל.

מפונקציית feature importance:

הפונקציה מספקת את החשיבות היחסית של כל משתנה במודל עץ ההחלטות. הערכים המוצגים בגרף משקפים את התרומה היחסית של כל משתנה לעץ שלנו:

Type of Travel- זהו המשתנה החשוב ביותר במודל, עם החשיבות הגבוהה ביותר (כ-0.25). הוא המפריד הראשוני והמשמעותי ביותר בין הקבוצות השונות בנתונים.

Inflight wifi service- משתנה זה הוא השני בחשיבותו, עם ערך של כ-0.15.

Overall Service Rating, Customer Type ו Cleanliness- תורמים גם הם עם ערך של כ-0.10.

Ease of Online booking - משתנה זה תורם פחות מהאחרים, עם ערך של כ-0.05, אך עדיין נחשב כמשפיע.

יתר המשתנים תורמים בצורה מועטה יחסית למודל, עם חשיבות נמוכה מאוד.

הפלט של הפונקציה מתאים למסקנות שהסקנו מהתבוננות בעץ אכן המאפיינים שזיהינו במקום גבוהה בעץ וגם אלו שחוזרים על עצמם, בעלי חשיבות משמעותית במודל. עם זאת הפונקציה שופכת אור על הדירוג בין המאפיינים עם החשיבות. (לדוגמא ease of online booking הוא משמעותי אך לא כמו כפי שהסקנו מהתבוננות בעץ).

Neural Networks

1. הכנת הנתונים להכנסה לרשת נוירונים:

רשת נוירונים רגישה ל Feature Scaling ולכן מומלץ לתקנן את הנתונים. בחרנו להשתמש בפונקציית Standard Scaler. השימוש ב-Standard Scaler ברשתות נוירונים נועד בעיקר לשפר את ביצועי הרשת ולהאיץ את תהליך הלמידה. על ידי סטנדרטיזציה של נתונים (תוחלת 0 וסטיית תקן 1), אנו מונעים הטיה הנובעת מסקלות שונות של משתנים, ומתאימים את הקלט לטווח האופטימלי של פונקציות האקטיבציה. כל אלה מובילים לתהליך אימון מהיר יותר, דיוק גבוה יותר ויכולת הכללה משופרת של המודל.

2. הרשת בערכי ברירת מחדל:

הקונפיגורציה שנלמדה על ידי המודל היא:

מספר נוירונים בשכבת הכניסה – כמספר הפיצ'רים. במקרה שלנו 25 פיצ'רים.

שכבה אחת חבויה ובה 100 נוירונים חבויים.

פונקציית האקטיבציה היא Relu.

בחרנו את מדד הדיוק F1, ומדד הביצוע המתקבל על סט האימון הוא: 0.94502 ועל סט המבחן: 0.88132.

מסקנות

הקונפיגורציה והתוצאות מצביעות על מודל בעל ביצועים טובים, עם יכולת למידה והכללה סבירה. הפער בין ביצועי סט האימון וסט המבחן מרמז על מידה מסוימת של התאמת יתר. ננסה בשלבים הבאים, להגדיל את תוצאת המדד על סט המבחן על ידי כיוונון פרמטרים.

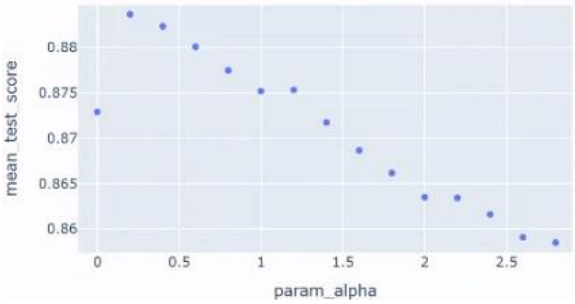
3. כיוונון פרמטרים

תחילה, ביצענו כיוונון פרמטרים לפי GridSearchCV וזאת כדי למצוא את הקונפיגורציה האופטימלית שתמקסם את מדד הדיוק על סט האימון בצורה של K-Fold Cross Validation. עבדנו עם CV=10. בצורה זו, נמדוד את הדיוק על עשרה סטים שונים והתוצאות תתקבל לפי מקסום הממוצע.

בחינה היפר פרמטרים

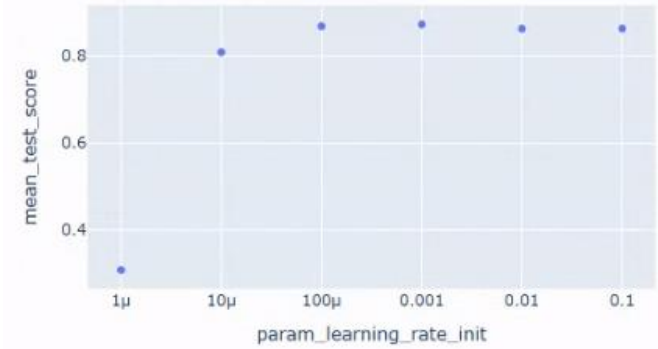
1. פרמטר α – ערך אלפא גבוה מעודד את המודל להשתמש במשקלים קטנים יותר, מה שמפחית את הסיכון להתאמת יתר ומשפר את יכולת ההכללה של המודל. ערך אלפא נמוך מאפשר למודל חופש רב יותר בקביעת המשקלים, מה שמוביל לפחות רגולריזציה. בחירת ערך אלפא מתאים היא חשובה לאיזון בין התאמה לנתוני האימון לבין הכללה טובה על נתונים חדשים.

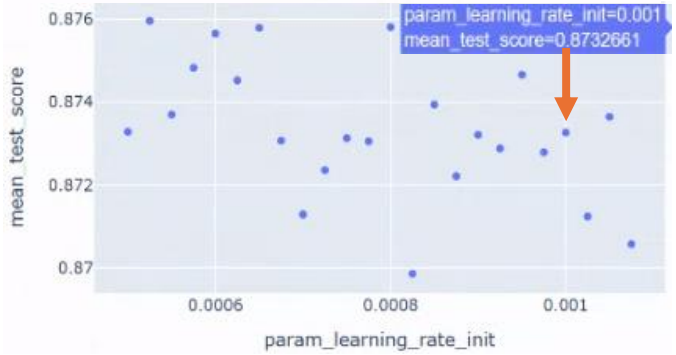
תחילה ביצענו ניסוי וטעיה על המודל כדי לחוש את הנתונים. התחלנו מטווח רחב של מינוס 3 עד 3 בקפיצות של 0.2. ראינו כי לטווח המספרים השלילים אין משמעות וקיבלנו סוג של קו לינארי שהנקודה



הגבוהה ביותר שלו קרובה ל0. לאחר תהליך זה, ביצענו מספר איטרציות כאשר לבסוף נוכחנו לדעת שטווח המספרים בין 0.1825 ל-0.3375 נותן את התוצאה הגבוהה ביותר.

על ידי תהליך זה שיפרנו את ביצועי מודל ברירת המחדל שעשה שימוש באלפא 0.0001, מתוצאה של 0.873 למעל 0.88.





2. <u>קצב הלמידה (learning rate)-</u> קובע את גודל הצעדים שהמודל לוקח בכיוון הגרדיאנט בזמן עדכון המשקלים. קצב למידה גבוה מדי עלול לגרום לאי-התכנסות, בעוד קצב נמוך מדי יכול להוביל לאימון איטי מאוד. בחירת קצב למידה מתאים היא קריטית לאימון יעיל ומוצלח של המודל.

בתחילת תהליך זה, בחנו נקודות סביב נקודת ברירת המחדל בקפיצות מעריכיות (0.1, 0.01, 0.001..). שמנו לב, שלנקודת ברירת המחדל תוצאת ביצועי המודל הגבוהה ביותר. מכאן ביצענו מספר איטרציות סביבה, עד אשר הגענו לטווח שתוצאות ביצועיו הניבו תוצאות גבוהות מזו של נקודת ברירת המחדל.

3. <u>שכבות חבויות -</u> שכבות חבויות הן השכבות הממוקמות בין שכבת הקלט לשכבת הפלט. הן מאפשרות למודל ללמוד ייצוגים מופשטים ומורכבים של הנתונים. הוספת שכבות חבויות מגדילה את עומק הרשת ואת יכולתה ללמוד פונקציות מורכבות יותר. מספר השכבות החבויות וגודלן משפיעים על יכולת הלמידה של הרשת ועל הסיכון להתאמת יתר\חסר.

param_hidden_layer_sizes	mean_test_score	std_test_score
(10, 8)	0.882577	0.017931
(12, 12)	0.879290	0.019628
(10, 12, 10)	0.878852	0.018922
(12, 10)	0.878588	0.018493
(10, 20)	0.878449	0.019539
(10, 10, 10)	0.878273	0.019180
(9, 9, 9)	0.877243	0.014448
(8, 10, 8)	0.876745	0.019337
(8, 10)	0.876670	0.018444
(8, 8, 8)	0.875810	0.018020
(9, 9)	0.875298	0.020094

תחילה, בדקנו מספר רשתות עם עומק 3. קיבלנו שהרשת שמניבה את התוצאה הגבוהה ביותר היא הרשת (10,10,10). לאחר מכן, בחנו מספר רשתות בעלות עומק דומה ומספר נוירונים בכל שכבה בין 8 ל-20. מהאיטרציה הזו עלה שהרשת (10,8) מפיקה את הדיוק המקסימלי עבור סט האימון, התוצאה היא 0.8825 עם סטיית תקן נמוכה 0.0179.

	param_activation	mean_test_score	std_test_score
1	logistic	0.883065	0.017081
3	relu	0.873266	0.018858
2	tanh	0.864736	0.013732
0	identity	0.821060	0.018478

4. <u>פונקציית אקטיבציה -</u> פונקציית אקטיבציה ברשתות נוירונים מכניסה אי-ליניאריות למודל, מאפשרת למודל ללמוד יחסים מורכבים בנתונים. היא קובעת את הפלט של כל נוירון בהתבסס על הקלט המשוקלל. בחירת פונקציית האקטיבציה המתאימה חשובה לביצועי המודל ויכולה להשפיע על מהירות הלמידה.

ניתן לראות כי פונקציית סיגמואיד (logistic) מפיקה את הדיוק המקסימלי, אך גם פונקציות relu ו-tanh הפיקו תוצאות גבוהות וגם אותן נבדוק בGridSearch המאוחד.

את טווחי הפרמטרים הטובים ביותר שמצאנו בנפרד, נבדוק כעת בGridSearch מאוחד (GridSearch בו ארבעת הפרמטרים נדבקים יחד). בנוסף לכך, הוספנו לכל פרמטר את ערך ברירת המחדל על מנת לאמוד אותם ביחס לפרמטרים האחרים.

לאחר הרצה אחת (שארכה קרוב ל-4 שעות ובדקה 588 מועמדים אפשריים), הסקנו מספר מסקנות:

- המודל הפיק את התוצאות הגבוהות ביותר תחת הפונקציות tanh ו-relu.
- המודל הפיק את התוצאות הגבוהות תחת רשת חבויה אחת בגודל 100 (,100).

בעקבות מסקנות ההרצה הראשונה, ביצענו מספר התאמות:

- בדקנו רק את הפונקציות tanh ו-relu
- הורדנו את השכבות החבויות שאנחנו מצאנו כאופטימליות (שכבות עמוקות בעומק 2 או 3 עם גודל שכבה 20 לכל היותר). בדקנו במקום זאת שכבות חבויות בסדר הגודל של שכבת ברירת מחדל (80, 90, 110..), והוספנו גם כן שכבות בגודל זהה אך בעומק 2.

התוצאות של הרצה זו הראו ששכבות חבויות בעומק 1 הפיקו את התוצאות הכי גבוהות. כמו כן, עבור כל אלפא שנבדקה הקומבציניות המיטביות הופקו תחת פונקציית relu.

לכן, בהרצה האחרונה בחנו רק את פונקציית relu והוספנו עוד רשתות חבויות בעומק 1.

4. <u>בחירת קונפיגורציה אופטימלית</u>

הערכים שהתקבלו עבור הקונפיגורציה הטובה ביותר:

מספר הנוירונים בשכבת הכניסה הוא כמספר הפיצ'רים (25)

מספר השכבות החבויות הוא 1, ומספר הנוירונים הוא 120. (,120)

פונקציית האקטיבציה היא relu.

קצב הלמידה הוא 0.001, והאלפא היא 0.2375.

התוצאה על סט באימון בCross Validation (באמצעות הGrid Search) היא 0.8869 וסטיית התקן 0.017.

תוצאת F1 שהתקבלה על סט האימון כולו היא 0.92699 והתוצאה שהתקבלה על סט המבחן היא 0.89815.

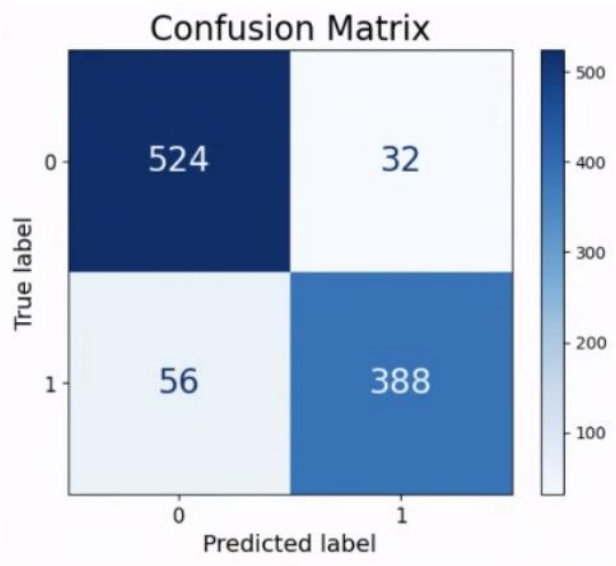
<u>הערה:</u>

בהרצה הראשונה של ה-GridSearch המאוחד שמנו לב כי הקומבניציה הטובה ביותר הניבה תוצאת F1 מעט טובה יותר על סט הבחינה - 0.90173. למרות זאת, בחרנו בקומבינציה שמצאנו בסוף כל התהליך. זאת מכיוון, שהיא ממקסמת את מדד F1 באמצעות Cross Validation. מדד שאנו סבורים שהוא מעט אמין יותר מתוצאת F1 שהתקבלה על סט המבחן מאחר ומדובר בעשרה סטים בלתי תלויים לעומת אחד.

מהתוצאות שעלו מהמודל ניתן לראות כי תוצאת F1 על סט האימון ירדה, ובמקביל תוצאת F1 על סט המבחן עלתה – כך בעיית התאמת היתר השתפרה והכללת המודל השתפרה גם כן.

מטריצת מבוכה

מטריצת הבלבול מראה שהמודל די מדויק. הוא זיהה נכון 524 מקרים שליליים ו-388 מקרים חיוביים. היו מעט טעויות: 32 false positives ו-56 false negative . בסך הכל, המודל צדק ב-912 מתוך 1000 מקרים, מה שמצביע על ביצועים טובים אך עם מקום מסוים לשיפור.



K-Means

מודל ברירת המחדל – מספר האשכולות = 2

לפני הרצת המודל, ביצענו סטנדרטיזציה לנתונים. השתמשנו בStandard Scaler על מנת לנרמלם.

הרצנו את המודל יחד עם ערכי ברירת המחדל כאשר K=2. בחרנו K=2 מאחר והוא יתאים לבעיית הסיווג שלנו ובנוסף נרצה לזהות שתי מגמות עיקריות.

כדי לקבוע איזה אשכול מייצג איזו קטגוריה במשתנה המטרה (Satisfaction), בחנו שתי אפשרויות סיווג.

באפשרות הראשונה, שייכנו את אשכול 0 לערך 0 במשתנה המטרה, ואת אשכול 1 לערך 1.

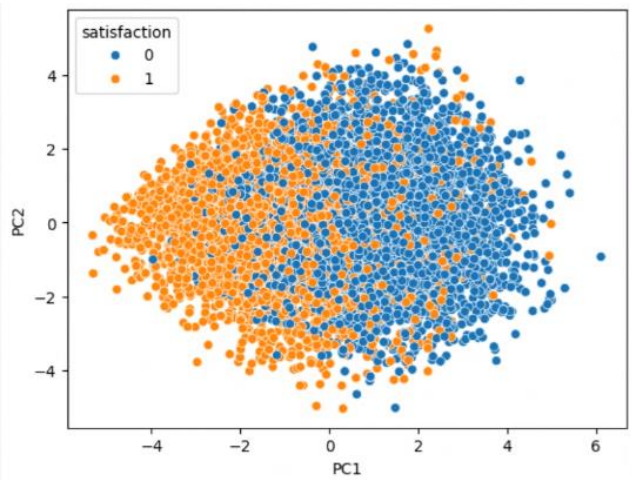
באפשרות השנייה, ההפך - אשכול 0 שויך לערך 1, ואשכול 1 לערך 0. לכל אפשרות חישבנו את ציון F1.

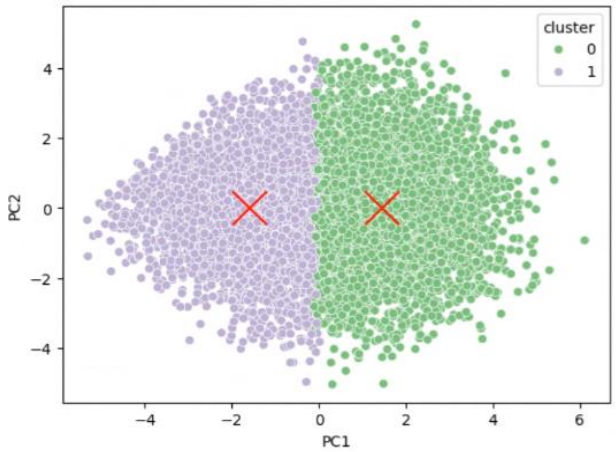
תוצאות הבדיקה הראו כי עבור האפשרות הראשונה, ציון F1 עמד על 0.72966. לעומת זאת, באפשרות השנייה, ציון F1 היה נמוך משמעותית ועמד על 0.21378.

התוצאות עלו בקנה אחד כפי שתיארנו לעצמו, אשכול 0 יהיה שייך לערך 0 וכך גם אשכול 1 לערך 1. יתר על כן, ציון F1 לא הפתיע אותנו מאחר ויש חריגים בכל אשכול.

לאור תוצאות אלו, בחרנו לאמץ את האפשרות הראשונה כשיטת הסיווג המועדפת. משמעות הדבר היא שאשכול 1 יתאים למחלקה 1 (Satisfied) במשתנה המטרה, ואילו אשכול 0 יתאים למחלקה 0 (Neutral/Dissatisfied).

בחירה זו מבטיחה את הדיוק הגבוה ביותר בסיווג התצפיות.





אימון מודל נוסף

במסגרת משימה זו, בחרנו להתמקד במודל XGBoost (eXtreme Gradient Boosting) כאלגוריתם סיווג מתקדם. XGBoost פועל על עיקרון של בניית מודל הדרגתית, המשלבת מספר רב של עצי החלטה חלשים לכדי מודל מאוחד חזק. כל עץ חדש במודל מתמקד בתיקון שגיאות של העצים הקודמים, תהליך המכונה boosting. גישה זו מאפשרת למודל לזהות ולייצג יחסים מורכבים בנתונים.

יתרונות המודל:

1. ביצועים גבוהים במגוון רחב של משימות סיווג ורגרסיה.
2. יעילות חישובית גבוהה, המאפשרת אימון וחיזוי מהירים בסטי נתונים גדולים.
3. יכולת מובנית לטיפול בערכים חסרים.
4. מנגנוני רגולריזציה מתקדמים למניעת overfitting.

פרמטרים עיקריים שבחנו:

- max_depth: קביעת עומק העצים
- n_estimators: מספר העצים במודל
- learning_rate: קצב הלמידה
- subsample ו-colsample_bytree: פרמטרים לרנדומיזציה ומניעת התאמת יתר
- min_child_weight ו-gamma: פרמטרים נוספים לרגולריזציה

תהליך הכיוונון וביצועים:

תחילה, בדקנו מהם ערכי ברירת המחדל שהמודל מציע. לאחר מכן, הוספנו לכל פרמטר מספרים בטווח של ברירת המחדל. מהאיטרציה הראשונה בחרנו את הקונפיגורציה בעלת ציון ה-CV הגבוה ביותר. מקונפיגורציה זו עלה כי קיים חשש להתאמת יתר מאחר ותוצאת F1 על סט האימון כולו היא 0.95913 ואילו על סט המבחן 0.90233.

לאחר ניסוי וטעיה, עלה כי הקונפיגורציה הסופית שנבחרה היא:

colsample_bytree=0.8, gamma=0, learning_rate=0.1, max_depth=5, min_child_weight=3, n_estimators=100, subsample=0.8, random_state=42

ביצועי המודל הסופיים:

- F1 Score על סט האימון: 0.92833
- F1 Score על סט המבחן: 0.91455

המסקנה המרכזית שעלתה משימוש במודל XGBoost הוא האיזון בין התאמה להכללה. הצלחנו להשיג איזון טוב בין ביצועים גבוהים על סט האימון לבין יכולת הכללה על נתונים חדשים. להפתעתנו, הצלחנו להשיג ציונים מאשר המודלים שבחנו קודם לכן בפרויקט.

השוואה בין מודלים

1. עצי החלטה (DT) ורשתות נוירונים (NN) מועדפים על פני K-Means למשימות סיווג מכמה סיבות מרכזיות. בעוד ש-DT ו-NN הם אלגוריתמים של למידה מונחית המתוכננים לסיווג, K-Means הוא אלגוריתם לא מונחה המיועד לקיבוץ. DT ו-NN מנצלים מידע מתויג, לומדים ישירות את הקשר בין מאפיינים לתוויות, ומסוגלים להתמודד עם גבולות החלטה מורכבים. למרות שניתן להשתמש ב-K-Means לסיווג בתנאים מסוימים, הוא פחות יעיל ומדויק במשימה זו בהשוואה ל-DT ו-NN.

2. השוואת ביצועי המודלים:

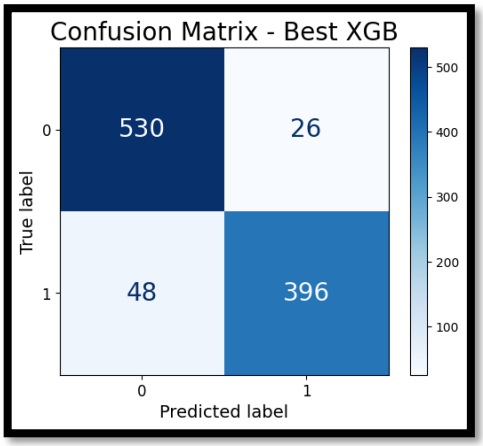
F1 \ Model	DT	NN	XGB
CV	0.88026	0.88691	0.89792
Train	0.89048	0.92699	0.92833
Test	0.90536	0.89815	0.91455

כפי שניתן לראות, מודל הXGBoost נתן את הביצועים הטובים ביותר במדד F1, הן באמצעות Cross-Validation של סט האימון והן על סט הבחינה הבלתי תלוי. לאור זאת, נבחר בXGB.

המודל הנבחר - XGB

1. כאמור, המודל הנבחר הוא XGBoost עם הפרמטרים הבאים:

colsample_bytree=0.8, gamma=0, learning_rate=0.1, max_depth=5, min_child_weight=3, n_estimators=100, subsample=0.8, random_state=42



2. מטריצת הבלבול מראה שהמודל די מדויק.
הוא זיהה נכון 530 מקרים שליליים ו-396 מקרים חיוביים. היו מעט טעויות: 26 false positives ו-48 false negative . בסך הכל, המודל צדק ב-920 מתוך 1000 מקרים, מה שמצביע על ביצועים טובים אך עם מקום מסוים לשיפור. עם זאת, יכל להיות מקום לשיפור בזיהוי תצפיות חיובית: כ-11.5% מהתצפיות שסווגו כשליליות היו צריכות להיות להיות מסווגות כחיוביות (לעומת 5% במצב ההפוך).