12.05.2022

אליהו אטין

רון קנדלשיין

**כריית מידע - חלק ב'**

**מטרת העל:**

יצירת מודל אשר יסייע לדרג הניהולי במחלקת השיווק להחליט לאילו לקוחות פוטנציאלית לפנות.

**הקדמה:**

1. מכיוון שהיה לנו מספר בעיות בהגשת חלק א' ערכנו את המידע (train data) וכמובן ביצענו עיבוד מתאים לtest data. על כן, לאחר עיון בהערות ומחשבה נוספת תוך כדי התהליך - שינינו את ייצוגו של חלק מהמידע. לשם הפשטות, והתייחסות למה שרלוונטי לתרגיל הנוכחי – נתייחס לכך שביצענו את חלק א' כפי במוצג להלן.
2. **עיבוד מקדים**

בפרק זה נציג את העיבוד המקדים שביצענו על מנת לייעל את המודל שיצרנו. נציג את הצעדים שביצענו ואת ההצדקה לביצועם.

עשינו שימוש במידע אודות 560 לקוחות – יש לציין כי המידע אינו שלם עבור כל לקוח.

**הערה**:

1. סיווג המידע

על מנת לייעל את עבודת המודל, במסגרת העיבוד המקדים, ביצענו סינון של מידע לא רלוונטי לפי מסקנות שהגענו אליהן בחלק א' ושינינו ייצוג של מידע מסוים.

בכדי ליצור מידע אשר מתאים לתהליך הסיווג העברנו את המידע בסינון ראשוני, אפיינו את המידע ראינו התפלגויות עבור כל הערכים ועל סמך זה את הסיווגים הבאים:

1. **Education:**

מידע זה הינו מידע קטגורי זה אשר הופיע בצורה של מחרוזת. המרנו לערך מספרי.

1. **Income:**

את המידע הרציף הנ"ל המרנו למידע קטגורי. הבחנו שמרבית המשכורות הינן בטווח של 0 עד מאה אלף ₪ (באופן דומה למידע האימון). על כן ביצענו את החלוקה באופן שנקבל עד כמה שאפשר חלוקה אחידה בין הקבוצות.

Chart, bar chart

Description automatically generated

1. **Age:**

את המידע הרציף הנ"ל המרנו למידע קטגורי. באופן זהה להכנסה, ניסינו ע"י ביצוע equal-length partition לחלק לקבוצות כך שהחלוקה תהיה אחידה.

Chart, bar chart

Description automatically generated

1. **Status**:

המרנו את המידע הקטגוריאלי בעמודה הנ"ל למידע קטגוריאלי מספרי לטובת המודל.

1. **חלוקת המידע**
2. חילקנו את המידע שלנו למאגר train ומאגר validation כדרוש. היחס שבחרנו להשתמש הינו 70% עבור train ו-30% עבור validation.
3. **cross validation:**

בהרצת האלגוריתמים נעשה שימוש בשיטת cross validation כנדרש.

על מנת לחלק את training data לtrain ו- validation , נחלק מראש את כל הtraining set למספר חלקים, וכל פעם חלק אחר יהיה סט הוולידציה. נעשה זאת על ידי שימוש בפונקציה KFold.

1. עשינו 2 הרצות מרכזיות של המודל:

**הרצה 1**

עמודת היעד שלנו, הינה איחוד (OR) של כלל הקמפיינים שנעשו עבור אותו הלקוח. בכך רצינו לבדוק את התגובתיות של הלקוחות החל מרישומם הראשוני.

**הרצה 2:**

עמידת היעד שלנו הייתה תגובתו של הלקוח לקמפיין האחרון שנערך.

1. **סיווג המידע**

בכדי למצוא מסווג מתאים לצורך שלנו עשינו שימוש במספר מסווגים שונים.

עבו כל מסווג ביצענו בדיקות בכדי לבדוק את רמת הדיוק בחיזוי שלו. החלוקה של כל מסווג הייתה זהה למצוין.

1. **Decision Tree**

המסווג הראשון שבחרנו הוא עץ החלטות.

1. אימון והרצה עם ערכים ראשוניים וחיזוי:

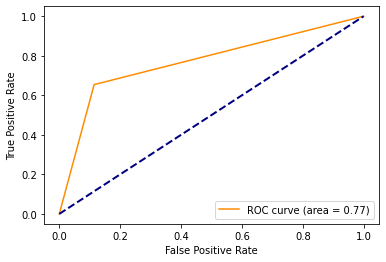
תוצאות האימון:

accuracy= 0.836

sensitivity= 0.6542056074766355

precision= 0.6086956521739131

והROC- לאחר האימון הוא:



תוצאות החיזוי על הtest: accuracy= 1.0

sensitivity= 1.0

precision= 1.0

ניתן לראות שקיבלנו חיזוי מאוד גבוה הודות להיווצרות של overfitting . כלומר היצמדות יתר של המסווג לערכים ויצירת מצג לא נכון בעקבות תכנון לקוי של המסווג. בכדי להימנע מכך, וכך גם עשינו במסווגים הבאים בדקנו את ההיפרפרמטרים של המסווג בכדי להגיע לתוצאה הכי יעילה ואמינה. בעץ החלטות ללא הגבלת ההיפרפרמטרים ניתן בקלות להגיע לאובר פיטינג מכיוון שהוא עובר לצומת הבאה עבור כל מספר קטן של דוגמאות עד שנשאר מספר מאוד קטן של דוגמאות בקצה. בעזרת שינוי ההיפרמטרים הגענו לתוצאה מדוייקת ואמינה יותר. נעזרנו בפונקצייה בהיפרפרמטר טיונינג בכדי להוביל אותנו לערכים שיבטיחו אמינות. בתהליך זה בחרנו טווח ערכים רחב עבור האפשרויות המרכזיות כפי שמובא כאן

max\_depth = [3,5,8,10,15,20]

criterion = ['gini','entropy']

splitter = ['best', 'random']

min\_samples\_split= [10,50,75,100,125,200, 500]

min\_samples\_leaf= [1,5,10,20,50,100, 200, 500]

הסבר על ההיפר פרמטרים שהתמקדנו בהם:

* Max\_depth: משפיע על כמה העץ יהיה עמוק ככל שהעץ עמוק יותר המסווג הדוק יותר וישפיע על רמת הדיוק שלנו.
* Min\_sample\_split: קובע עבור כמה דוגמאות יווצר פיצול חדש בעץ ובעצם קובע כמה פיצולים יהיו בעץ.
* Min\_sample\_leaf: מספר הדוגמאות שיהיו בעלה הסופי משפיע על הדיוק שלנו, לא נרצה להתייחס לעלים עם מספר קטן מידי של דוגמאות מהדאטא.
* Criterion: קובע איך נבחר את החלוקה לפי entropy/gini המסווג בודק איזו חלוקה נותנת תוצאות טובות יותר.
* ראשית השתמשנו באלגוריתם רנדומי למציאת ההיפר פרמטרים האופטימלייים וקיבלנו שהם:

{'splitter': 'random',

'min\_samples\_split': 50,

'min\_samples\_leaf': 10,

'max\_depth': 15,

'criterion': 'gini'}

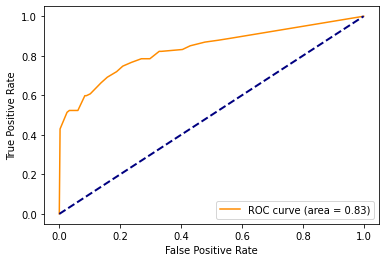
אימנו מחדש לפי ההיפר פרמטים האופטימליים ואלה תוצאות האימון:

accuracy= 0.85

sensitivity= 0.5233644859813084

precision= 0.7

והעקומת ROC המתאימה היא:



בדקנו את המודל החדש על הtest כדי לבדוק שאנו לא שוב בoverfitting וקיבלנו את התוצאות הבאות:

accuracy= 0.8935622317596567

sensitivity= 0.6050420168067226

precision= 0.8275862068965517

וניתן לראות שאכן הגענו לתוצאות טובות יחסית ודומות לתוצאות האימון הראשוני וכנראה שלא נוצר overfitting .

* לאחר מכן הפעלנו אלגוריתם greedy למציאת ההיפר פרמטרים:

{'splitter': [best]

min\_samples\_split [10,50,75,100,125,200, 500],

min\_samples\_leaf [1,5,10,20,50,100, 200, 500],

max\_depth [3,5,8,10,15,20],

criterion': ['gini', 'entropy']

וקיבלנו שהפרמטרים הטובים ביותר הם:

'criterion': 'entropy'

'max\_depth': 3

'min\_samples\_leaf': 1

'min\_samples\_split': 50

'splitter': 'best'

אימנו עם הפרמטרים הללו והרצנו חיזוי על הtest ואלה התוצאות:

accuracy= 0.85

sensitivity= 0.5233644859813084

precision= 0.7

נראה שקיבלנו תוצאות קצת פחות טובות מהאלגוריתם הרנדומי להיפר פרמרים אבל בכל מקרה המודל לא בoverfitting.

1. **Random Forest**

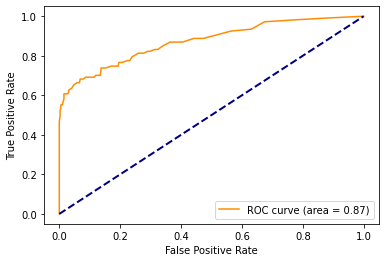
מודל שיוצר מספר סוגים של עצים משווה ביניהם ולוקח את העץ שנותן את התוצאה הטובה ביותר. הוא לא מבצע תהליך של תיקון אלא יוצר כמה עציh החלטה מחלקים שונים של הdata.

יתרון על מסווג עץ החלטה רגיל שכאן אנו מקבלים מספר עצים אם פיצולים שונים שלא היו נבחרים בעץ ההחלטה הרגיל ואז אנו בודקים מי העץ עם התוצאות הטובות ביותר.

ההיפר פרמטרים למסווג זה הם:

* Max\_depth: משפיע על כמה העץ יהיה עמוק ככל שהעץ עמוק יותר המסווג הדוק יותר וישפיע על רמת הדיוק שלנו.
* Min\_sample\_split: קובע עבור כמה דוגמאות יווצר פיצול חדש בעץ ובעצם קובע כמה פיצולים יהיו בעץ.
* Min\_sample\_leaf: מספר הדוגמאות שיהיו בעלה הסופי משפיע על הדיוק שלנו, לא נרצה להתייחס לעלים עם מספר קטן מידי של דוגמאות מהדאטא.
* N\_estimators: מספר העצים ביער.
* ראשית אימנו את המודל ללא כיוונון ההיפר פרמטרים ואלו תוצאות האימון:
* accuracy= 0.896
* sensitivity= 0.5514018691588785
* precision= 0.9365079365079365

עקומת הROC:



* לאחר מכן הרצנו חיזוי של המודל על הtest ואלה התוצאות שקיבלנו:

accuracy= 1.0

sensitivity= 1.0

precision= 1.0

ניתן לראות בבירור שהמודל בoverfitting ולכן חיפשנו את ההיפר פרמטרים לשינוי:

criterion: ['gini', 'entropy'],

max\_depth': [5, 10, 15, 20, 25, 30, 35, 40, 45, 50, None]

max\_features': ['auto', 'log2', 2, 5, 8, 16, 19]

max\_samples': [0.1, 0.3, 0.5, 0.7, 0.9]

min\_samples\_leaf': [10, 20, 50, 100]

min\_samples\_split': [10, 20, 50, 100]

n\_estimators': [100-1000 with jumps of 18]

זה הערכים שרצינו לבחון, השתשמנו באלגוריתם הרנדומי למציאת ההיפר פרמטרים האופטימליים וקיבלנו שהם:

n\_estimators: 742

min\_samples\_split: 50

min\_samples\_leaf: 10

max\_samples: 0.9

max\_features: 16

max\_depth: 35

criterion: 'gini'

* לאחר כיוונון ההיפר פרמטרים אימון מחד של המודל והרצתו על הtest אלה התוצאות שקיבלנו:

random model accuracy = 88.60%

random model sensitivity = 52.34%

random model precision = 90.32%

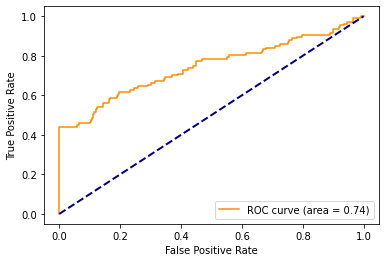
ניתן לראות שהתוצאות אכן קרובות לתוצאות האימון הראשוני וכנראה שהמודל לא בoverfitting.

1. **SVM classifier**

מודל זה מנסה לחלק את הדאטא לינארית מכיוון שהדאטא אינו בהכרח לינארי המודל מנסה לעלות את מימד הדאטא למימד שבו יהיה ניתן לבצע את החלוקה.

* ראשית אימנו את המודל על הtraining data שלנו ואלה תוצאות האימון:
* base model accuracy = 87.80%
* base model sensitivity = 43.93%
* base model precision = 97.92%

עקומת הROC:



* הרצנו את המודל על הtest ואלה התוצאות שקיבלנו:
* accuracy= 0.871244635193133
* sensitivity= 0.3739495798319328
* precision= 0.9888888888888889

לא לגמרי ברור עם המודל בoverfit בכל מקרה ניסינו לכוונן את ההיפר-פרמטרים לשיפור המודל ואלה הפרמטרים שבחרנו וטווח הערכים שלהם:

C: [0.1, 0.3, 0.5, 0.7, 0.9]

decision\_function\_shape: ['ovo', 'ovr']

kernel: ['linear', 'poly', 'rbf', 'sigmoid']

probability: [True]

kernel- פונקציית הגרעין להעלאת המימד.

C- גורם רגולריזציה שמשפיע על המשקלים בפונקצייה.

לאחר הרצת האלגוריתם הרנדומי לחיפוש ההיפר פרמטרים האופטימליים קיבלנו שהם:

probability: True

kernel: 'linear'

decision\_function\_shape: 'ovo'

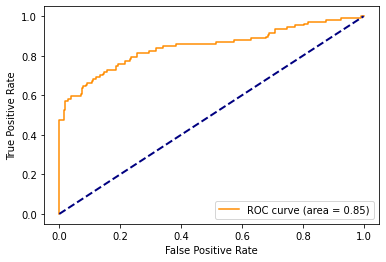
C: 0.5

-לאחר אימון עם ההיפר פרמטרים קיבלנו את תוצאות האימון הבאות:

random model accuracy = 88.60%

random model sensitivity = 43.93%

random model precision = 100.00%



-הרצנו את המודל על הtest ואלה התוצאות שקיבלנו:

accuracy= 0.8772532188841202

sensitivity= 0.40756302521008403

precision= 0.9797979797979798

1. **AdaBooster**

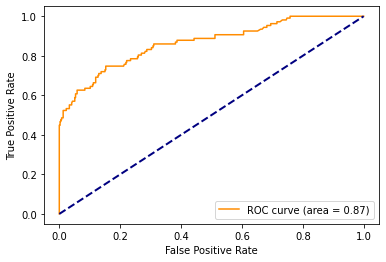
אלגוריתם adaptive boost- אלגוריתם למידה המחפש מספר קטן של מסווגים "חזקים" מתוך קבוצה של מסווגים "חלשים"ת האלגוריתם מעניק משקל גדול לשגיאות בזיהוי כאשר כל משקל מסמל את חשיבות התכונה.

* תוצאות אימון המודל:

accuracy= 0.87

sensitivity= 0.6074766355140186

precision= 0.7386363636363636



* תוצאות הרצת המודל על הtest:
* accuracy= 0.9201716738197425
* sensitivity= 0.6638655462184874
* precision= 0.9239766081871345
* היפרפרמטר שבחרנו לשם כיוונון המודל:

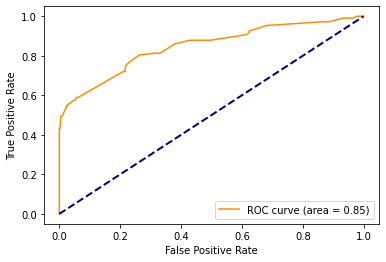
n\_estimators: [10, 20, 25, 30, 35, 50, 100, 150, 200, 400]

תוצאות האימון לאחר בחירת ההיפר פרמטר: :n\_estimators': 10

accuracy= 0.884

sensitivity= 0.5420560747663551

precision= 0.8656716417910447



* תוצאות האימון של המודל החדש על הtest:
* accuracy= 0.8918454935622318
* sensitivity= 0.5672268907563025
* precision= 0.8544303797468354