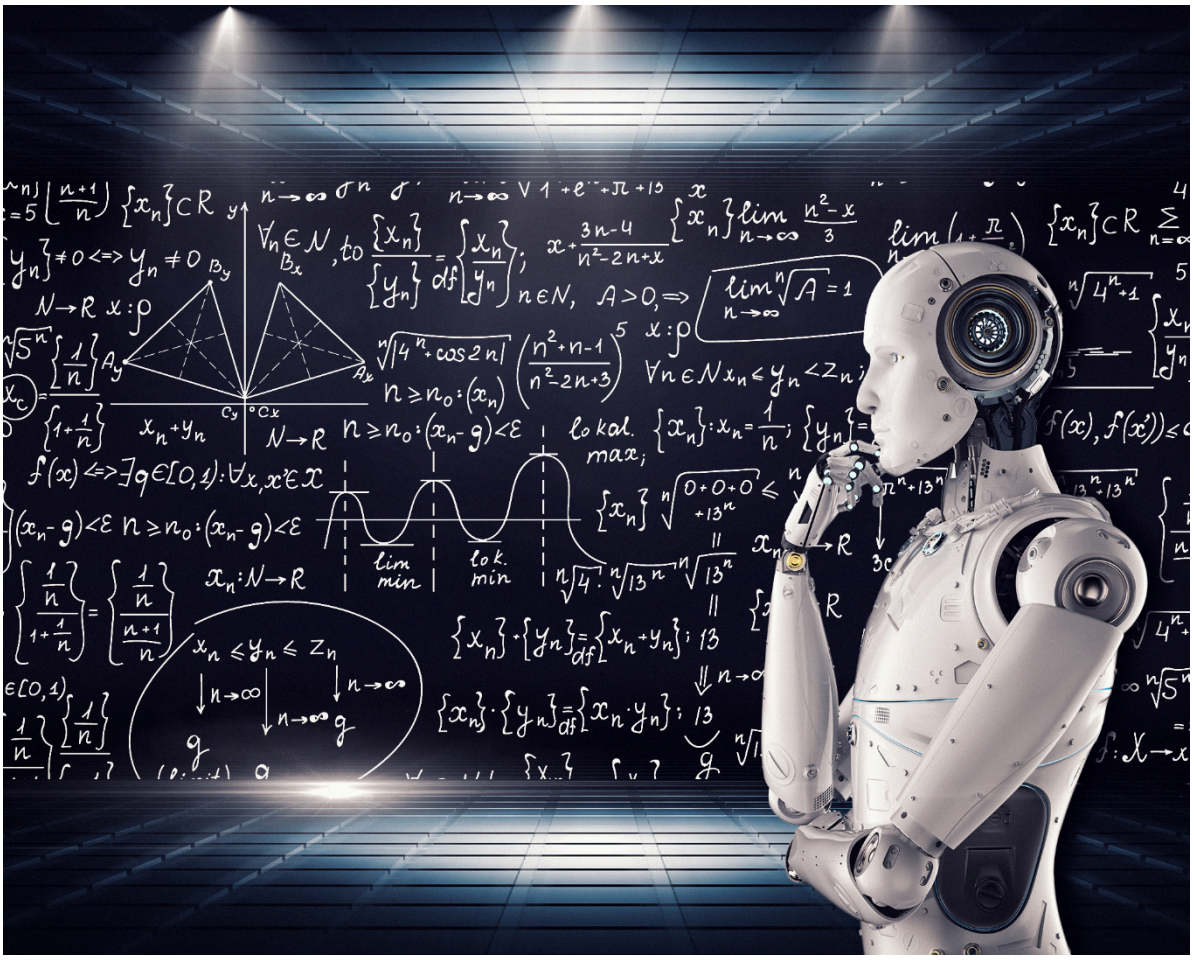
פרויקט במבוא ללמידת מכונה - בעיית סיווג

מגישים:

ארבל קציר , [redacted]

עומר אדרי, [redacted]

עידן כנת, [redacted]



תקציר מנהלים

בפרויקט זה, התמקדנו בבעית סיווג. ספציפית מהעולם העסקי, במטרה לנבא סיכויי הצטרפות של לקוח עתיד להרשם לתוכניות שהחברה מציעה, בהנתן מסבירים (כגון תוצאות של קמפיינים קודמים, גיל, מצב משפחתי, תאריכי התקשרות וכו') כלשהם. נעזרנו במודלי ML כדי לצבור תובנות משמעותיות ופרקטיות על הנתונים. וכן גם במטרה להחשף מעשית, להתנסות ולתרגל את החומר הנלמד במהלך הקורס. תחילה, ניתחנו את הנתונים באופן יסודי ומעמיק. על סמך הניתוח, יישמנו סדרת טרנספורמציות חיוניות עליהם. השינויים כללו המרת משתנים קטגוריאליים, הסרת ערכים חריגים, נירמול משתנים נומריים והשלמת ערכים חסרים. מה שהפך אותם לאינפורמטיביים יותר. בהמשך, אימנו מודלים מוכרים מהקורס בכמה שלבים, תוך שימוש בשיטות שנלמדו כגון train-test-split, K-Fold Cross Validation וכו'. הערכנו את ביצועי המודלים בשלל מובנים. לבסוף, בחרנו מודל אחד (עם היפרפרמטרים מסוימים) משיקולים מבוססי AUC. על סמך המודל המאומן, בחנו כיצד הוא מתמודד עם סט נתונים דומה שלא נראה בעבר.

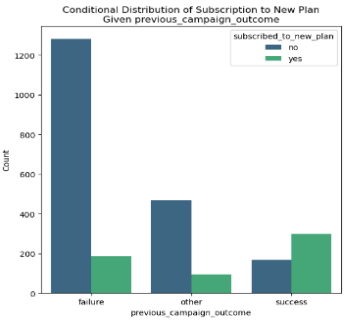
שלב 1 - ניתוח הנתונים:

תחילה, חיפשנו מענה לשאלות בסיסיות, אם כי רלוונטיות על הנתונים - כיצד הם מתפלגים? האם הם קורלטיביים זה עם זה? האם יש קורלציה בין כמות חריגים למשתנים המסבירים? כדי לקבל ידע מקדים אם יש צורך לבצע מניפולציות כלשהן עליהם, שיהיו רלוונטיות לשלב העיבוד המקדים (שלב 2). ראשית, הסרנו את עמודת customer_id מהנתונים, שכן סביר להניח שאין לה שום משמעות לחיזוי ערך המשתנה המוסבר. בנוסף, שינינו את שם העמודה day_of_week ל-day_in_month, שכן לדעתנו, השם החדש משקף באופן יותר נאמן למציאות את משמעות הערכים בעמודה זו (ערכים בין 1-31).

בדקנו תחילה כיצד המשתנים המסבירים המספריים מתפלגים (C & D נורמליים - C עם תוחלת סביב 6, D עם תוחלת סביב 100, הגאוסיאן שלהם נראה דומה (גם על סמך השנתות של ציר Y), מה שמעיד על סטיית תקן דומה), משתנים כמו: B, previous_campaign_contacts ו-current_campaign_contacts עם זנב ימני קיצוני, מה שיעיד על הסרת חריגים משמעותית שנצטרך לבצע שם. בדקנו גם כיצד המשתנים הקטגוריאליים מתפלגים - המסקנה הכללית שלנו היתה שהקמפיינים הקודמים ברובם נכשלו, שרוב הלקוחות מעדיפים קשר טלפוני, רובם נשואים, עובדים בעבודות צווארון כחול ועם השכלה תיכונית.

שמנו לב שיש עמודות שניתן לבצע על הערכים שלהן תיקונים מסוימים - למשל, בעמודת "Preferred Call Time". באופן טרויאלי, יש שקילות בין "eve", "evening" ו-"Evening", וכך גם בין "morning" ל-"Morning", ובין "Night" ל-"Night " (אלו ערכים שהופיעו בעמודה זו). לכן המרנו את כל אחד מהזמנים השקולים ביום לזמן השקול לו ביום, בלי רווחים ועם אות גדולה בהתחלה. כך שהערכים אחרי הטרנספורמציה הם: "Night", "Evening" ו-"Morning". גם בעמודת "has_device_payment_plan", הערכים "No" ו-"no" שקולים ולכן המרנו את "No" ל-"no". השינויים בעמודת "has_device_payment_plan" לא שינו את המגמה הכללית של התפלגות המשתנה (קיבלנו לרוב "No" לפני ואחרי הטרנספורמציה), אך הטרנספורמציה שינתה לחלוטין את התפלגות "Preferred Call Time" - לפני הטרנספורמציה, ההתפלגות נראתה אחידה. אחריה, ערך "Afternoon" נעשה משמעותית פחות נפוץ משאר הערכים, שנראו אחידים זה עם זה.

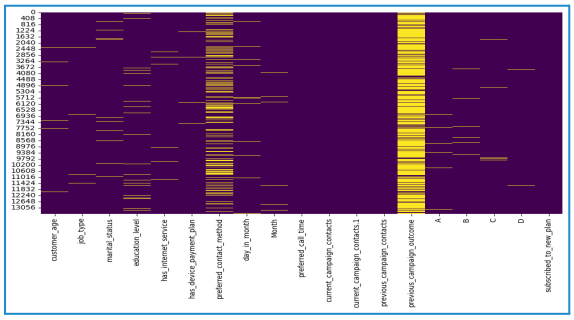
בנוסף, רצינו לבחון איך המשתנה המוסבר משפיע על התפלגויות מותנות - התוצאות שם עשויות להעיד על קשר מסוים בין המשתנה המוסבר לבין המסבירים. (כלומר, חילקנו את הנתונים לפי תוצאות המשתנה המוסבר, ובדקנו כיצד המשתנים המסבירים מתפלגים - בהנתן ערך של המשתנה המוסבר). לא שמנו לב לשינויים משמעותיים בין ההתפלגויות המותנות לבין ההתפלגויות המקוריות ברוב המקרים, חוץ ממקרה אחד: "Previous_Campaign_Outcome" - ניכר היה שכשהקמפיינים הקודמים נכשלו, אז גם ערך subscribed_to_new_plan היה לרוב שלילי, אבל כשהם הצליחו, אז גם ערך subscribed_to_new_plan היה ברובו חיובי.[1]



בחנו גם קורלציות של המשתנים המסבירים (בעצם של כל עמודות ה-data-frame) זה עם זה. תחילה, שמנו לב שבין המשתנים הנומריים הקורלציה מאוד נמוכה עד כדי לא קיימת (ערכים אי שליליים קרובים מאוד ל-0). בין המשתנים הקטגוריאליים המגמה דומה באופן כללי, אם כי הקורלציות בין חלק מהמשתנים לא נמוכה במיוחד (סביבות 0.3-0.4). בדיקת קורלציה משמעותית כי אנחנו רוצים להמנע מבעיית מולטיקולינאריות בה המשתנים המסבירים קורלטיביים זה עם זה. ברגרסיה לינארית זה מוביל לשונות מאוד גבוהה של האומדים.

כמו כן, רצינו לבחון את הקורלציה בין עמודות מסוימות לבין ערכים חסרים. שמנו לב שבעמודות ספציפיות - "Previous_Campaign_Outcome", רוב השורות היו עם ערכים חסרים, ובעמודת "Preferred_Contact_Method", חלק משמעותי מהערכים היו חסרים.[2]

שלב 2 - עיבוד מקדים – Preprocessing:



התמודדות עם ערכים חסרים: תחילה, שמנו לב שיש מס' מזערי של רשומות בהן ערכי "current_campaign_contacts.1" ו-"current_campaign_contacts" מתבדלים. לכן החלטנו להסיר את עמודת "current_campaign_contacts.1". אך לפני כן, במידה והיו ערכים חסרים ב-"current_campaign_contacts" שהיו לא ריקים בעמודה הכפולה לה, החלטנו להעתיק את הערכים הלא ריקים מהעמודה הכפולה לעמודת "current_campaign_contacts".

יתר על כן, אחוז מאוד גדול מהערכים בעמודת "previous_campaign_outcome" = ערכים ריקים (מעל 80%!). לכן החלטנו להסיר את עמודה זו. (תמונת האחוזים המדויקים - תופיע גם בנספח)[3]

```
Dropping 'previous_campaign_outcome'

# Getting the exact percentage of nulls in the 'previous_campaign_outcome' column:
df['previous_campaign_outcome'].isnull().sum()/df.shape[0]*100

81.44237918215613
```

עבור המסבירים הקטגוריאליים, החלטנו להחליף ערכים חסרים בשכיחים (אף על פי שטרנספורמציה שכזו עשויה לשנות את התפלגות המסבירים הקטגוריאליים), כיוון שמדובר באלגוריתם פשוט שמשמר תכונות אחרות בהתפלגות (השכיח של ההתפלגות נשמר באופן טרויאלי). עבור שאר המסבירים, השתמשנו ב-KNN-Imputer בשביל להתמודד עם ערכים חסרים נומריים, מה שדרש scaling שלהם, יצרנו העתק של הנתונים ונירמלנו אותם בהתאם מכיוון שKNN זה מודל מבוסס מרחק (הנירמול נעשה לפי שיטה שמפורטת בפסקת "נירמול") .

הסרת חריגים: לפי הויזואליזציות במחברת (boxplots & התפלגויות שוליות), הבנו שעלינו לבצע הסרת חריגים משמעותית בחלק מהעמודות. החלטנו להסיר חריגים בשיטת הטווח הבינרבעוני - כך שכל תצפית שלא בטווח: [Q1-1.5IQR, Q3+1.5IQR] תיחשב לחריגה ולכן תימחק. השתמשנו בשיטה זו ספציפית כיוון שהיא לא רגישה לחריגים - כדי לשנות את המספרים בטווח זה (Q1, Q3, IQR), יש צורך בשינוי של 25% מהנתונים (הן עבור Q1, Q3 ו-IQR). כמו כן, שיטה זו היא א-פרמטרית - לא מניחה כלום על ההתפלגות ולא דורשת שום פרמטרים של ההתפלגות. מדובר בשיטה פשוטה, שימושית למגוון רחב של יישומים, ולא רגישה לחריגים. עבור מסבירים שלא היו בהם המון חריגים מלכתחילה, השינויים לא היו משמעותיים (לכן גם השינויים של ההתפלגויות + ההתפלגויות המותנות ביחס לאלו שהצגנו בחלק 1 לא היו משמעותיים שם). אבל היו מסבירים עם המון חריגים כך שנעשה שיפור משמעותי. הסרה זו אמנם הסירה המון ערכים חריגים אך גרמה לנתונים שנשארו להיות אינפורמטיביים ואיכותיים יותר (זנבות פחות קיצוניים, טווחי ערכים פחות קיצוניים, יותר קל לנתח ולהבין את הנתונים שיש לנו ביד כרגע. שמנו לב שגם הקורלציות בין המסבירים הנומריים התקרבו ל-0 אחרי הסרת הערכים החסרים והחריגים, והמגמה ההפוכה חלה על המשתנים הקטגוריאליים - הקורלציות ביניהם התרחקו מ-0.

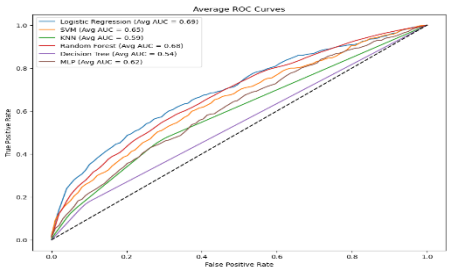
OHE – One Hot Encoding: כיוון שניתן להסיק שיוך לקטגוריה מסוימת עם / ללא OHE, וכיוון שהרבה מודלים של למידת מכונה לא יכולים להסתמך על נתונים קטגוריאליים (אי אפשר להריץ עליהם KNN למשל), החלטנו לבצע OHE על הנתונים שלנו, אף על פי שטרנספורמציה כזו מגדילה את מימדי הבעיה ומוסיפה המון מסבירים. לטעמנו, היא הכרחית מהסיבות הנ"ל. יתרה מכך, ביצענו OHE על מסבירים נומריים שהם אינטואיטיבית קטגוריאליים - למשל, יום בחודש - day_in_month (זה הגדיל את מימד הבעיה משמעותית - יש לנו 30 מסבירים על חשבון מסביר אחד). מטריצת הקורלציה החדשה התייחסה אל כל המסבירים כנומריים, מה שעזר לנו לבחון קורלציה בין מסבירים נומריים לבין מסבירים שבמקור היו קטגוריאליים (ולא בין קטגוריאליים לקטגוריאליים / בין נומריים לנומריים כמו בשלבים קודמים), ושמנו לב שהקורלציות בין המסבירים החדשים עדיין ברובן נמוכות - קרובות ל-0 ואי שליליות.

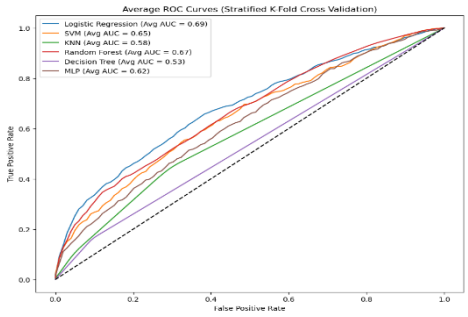
נירמול: בחרנו לבצע נירמול אחרי ביצוע OHE, כפי שנהוג. שכן התכוונו להשתמש ב-RobustScaler עבור המשתנים שלא באים מהתפלגות נורמלית. בניגוד לסקיילרים אחרים, כמו מינ-מקס סקיילר (Min-Max Scaler) או סטנדרד סקיילר (Standard (Scaler, רובסט סקיילר אינו מושפע כל כך מערכים חריגים. נירמול מסוג זה עבד הכי טוב על המסבירים הללו. ביניהם גם המשתנים הקטגוריאליים שעברו OHE, מה שהיה משמר את הבינאריות שלהם - הם אמנם לא היו מקבלים את הערכים {0,1} אלא {-0.5,0.5} אבל ההתאמה בין הערכים היא הפיכה - חח"ע ועל. (ערך 0.5- בתצוגה החדשה שקול לערך 0 בתצוגה הקודמת ולהפך, וערך 0.5 בתצוגה החדשה שקול לערך 1 בתצוגה הקודמת), מה ששקול להעתקה לינארית הפיכה - 0.5 – X. את שאר הנתונים נירמלנו לפי StandardScaler, שכן הם באים מהתפלגות נורמלית ולכן נירמול בצורה זו מתאים יותר. מגוון מודלים שנרצה להריץ מבוססים על נתונים מנורמלים - למשל, KNN, לכן נירמלנו את הנתונים.

בחירת המסבירים LassoCV: לאסו (Lasso) הוא טכניקה ברגרסיה שמבוססת על רגולריזציה מסוג L1, שמביאה לכיווץ המשקלות של חלק מהמאפיינים (features) לערך אפס. כתוצאה מכך, לאסו משמש ככלי יעיל לבחירת מאפיינים (feature selection) על ידי כך שהוא מסיר מאפיינים לא רלוונטיים, מקטין את מורכבות המודל (model complexity), ומונע אוברפיטינג (overfitting). זה מאפשר לבנות מודלים שמתרכזים רק במאפיינים החשובים ביותר ומספקים תוצאות מדויקות וקלות יותר לפירוש.

חלק 3 - הרצת המודלים:

תחילה, פיצלנו את הנתונים שלנו מסט האימון המקורי, לסט אימון חדש וסט ולידציה. על סט האימון החדש, ביצענו K-Fold Cross Validation וגם Stratified K-Fold Cross Validation (האופציה השניה משמרת איזון של הנתונים בניגוד לראשונה). בחרנו K = 10, ע"פ רוב ההמלצות שחקרנו אודותיהם באינטרנט K זה הכי טוב משיקולי bias-variance tradeoff, שכן K קטן מדי גורר שהמודל ילמד על מגוון נמוך יחסית של סטי אימון, מה שעשוי לגרור הטיה גבוהה, אך בכל סט יש יותר נתונים ולכן התוצאות שלו ייטו לא להשתנות בין ה-folds, מה שעשוי לגרור שונות נמוכה יחסית. עם זאת, K גבוה מדי (כמו ב-LOOCV) יכול לגרור הטיה נמוכה שכן המודל לומד על מגוון עצום של סטי אימון, אבל התוצאות של מודל שכזה ייטו להשתנות בין ה-folds שכן לכל סט אימון יש מעט נתונים, מה שעשוי לגרור שונות גבוהה יחסית. לאחר מכן, עבור כל אחד מ-6 המודלים, ובכל 1 מ-10 ה-folds, הדפסנו את המטריקות הרלוונטיות כמו AUC, accuracy, F1 score, Recall, עקומת ROC ו-Confusion Matrix, כדי לקבל הערכה של ביצועי המודלים שלנו על סט האימון. לאחר מכן, שמנו לב שגם בK-Fold CV וגם ב-Stratified K-Fold CV, ביצועי המודל היו מאוד דומים** (הסבר מפורט יותר על כך בהמשך פרק 3) . ב-2 שיטות ה-cross validation, בממוצע על כל ה-folds, רוב המודלים השיגו ביצועי accuracy ממוצעים מאוד גבוהים (0.8-0.93), ביצועי AUC סבירים (הטובים מביניהם - רגרסיה לוגיסטית ו-Random Forest השיגו ביצועים ממוצעים סביב 0.65-0.7, הבינוניים - SVM, MLP - השיגו accuracy דומה לקודמים וביצועי AUC סביב 0.6-0.65, והמודלים עם הביצועים הכי פחות טובים כמו KNN ועץ החלטה השיגו AUC סביב 0.55 - בין 0.53 ל-0.6 יחד עם accuracy סביב 0.85), וביצועי recall נמוכים במיוחד (פחות מ-0.2), מה שמעיד על יכולת לא טובה בחיזוי ערכים חיוביים, אך באופן כללי - יכולת החיזוי של הערכים השליליים בקרב המודלים טובה יותר, מה שמעיד על שילוב בין ערכי accuracy ממוצעים גבוהים לעומת ערכי recall ממוצעים נמוכים. המגמה הכללית שגילינו היא שיש התאמה בין המודלים המדויקים ביותר בין 2 השיטות (K Fold & Stratified K-Fold CV), כך גם עבור המודלים הטובים ביותר מבחינת AUC. עם זאת, ב-2 שיטות ה-cross validiation, דפוס ההתאמה עבור כל fold בין המודל הטוב ביותר מבחינת AUC לבין המודל המדויק ביותר (טוב ביותר מבחינת accuracy) היה אקראי לחלוטין. ב-2 השיטות, בחלק מה-folds היתה התאמה בין המודל הטוב ביותר למודל המדויק ביותר, ובחלק לא. וכשהיתה התאמה, המודלים אשר הביאו למקסימום את ערכי ה-accuracy וה-AUC לרוב היו רגרסיה לוגיסטית ו-Random Forest. בנוסף, ההבדלים במטריקות ה-accuracy וה-AUC הממוצעים, על פני 2 שיטות ה-cross validation, היו מזעריים עד כדי לא קיימים / רלוונטיים. (עקומות ה-ROC הממוצעות של 6 המודלים, על פני 2 שיטות הקרוס ולידציה - בנספחים).[4+5]





המטריקות הממוצעות של כל מודל על פני 2 שיטות הקרוס-ולידציה - בנספחים. מימין - השיטה המקורית. משמאל - סטרטיפייד.[6+7]

	Model	AUC	Accuracy	Recall
0	Decision Tree	0.539400	0.847319	0.169776
1	KNN	0.594493	0.913054	0.019098
2	Logistic Regression	0.694205	0.915761	0.044652
3	MLP	0.615753	0.878758	0.147009
4	Random Forest	0.676507	0.916363	0.022212
5	SVM	0.649195	0.916062	0.000000

	Model	AUC	Accuracy	Recall
0	Decision Tree	0.533868	0.847021	0.157857
1	KNN	0.580270	0.913056	0.014416
2	Logistic Regression	0.685502	0.915915	0.043182
3	MLP	0.624534	0.884177	0.154448
4	Random Forest	0.667017	0.915765	0.023377
5	SVM	0.649801	0.916065	0.000000

** עם זאת, שמנו לב להבדלים מאוד ספציפיים בין 2 שיטות ה-Cross Validation. אחרי מספר הרצות ואימונים של המודלים ב-2 השיטות, שמנו לב לתופעה כללית כך שה-AUC הממוצע של רוב המודלים גבוה מעט יותר ב-K-Fold-Cross Validation מבשיטת ה-Cross Validation השניה. תופעה זו מתהפכת כשמדובר ב-accuracy של רוב המודלים. אך כפי שציינו מקודם, המטריקות הממוצעות ב-2 שיטות ה-Cross Validation מאוד קרובות / דומות זו לזו. בעקבות הדמיון, החלטנו להתייחס בה"כ לתוצאות ולמטריקות לאחר K-Fold Cross Validation, כיוון שה-AUC הממוצע עבור רוב המודלים (באופן כללי) גבוה מעט יותר. עם זאת, אין לבחירה הזו משמעות פרקטית, כיוון שניכר כי אין הבדל גדול בשימוש בכל אחת מ-2 שיטות ה-Cross Validation. לכן הנחנו שניתן לבחון ולהתייחס למטריקות ממוצעות של כל המודלים ספציפית עבור אחת משיטות ה-Cross Validation.

להלן תיעוד (שיופיע גם בנספחים)[8]

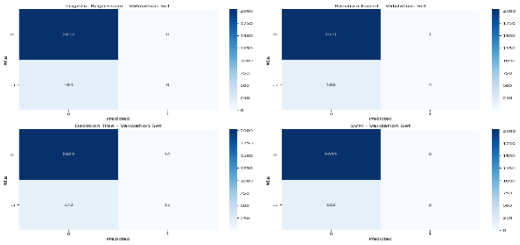
```
# Checking the difference in metrics between the models on the original K-Fold CV, and the ones on the stratified K-Fold CV:
print(avg_metrics.loc[:, ["AUC", "Accuracy", "Recall"]] >= avg_stratified_metrics.loc[:, ["AUC", "Accuracy", "Recall"]])
```

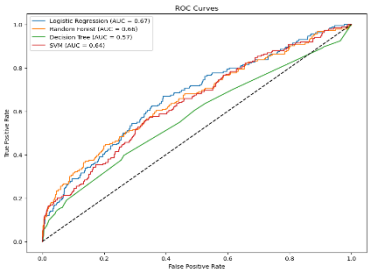
	AUC	Accuracy	Recall
0	True	True	True
1	True	False	True
2	True	False	True
3	False	False	False
4	True	True	False
5	False	False	True

חלק 4 - הערכת ביצועי המודל:

תחילה, כפי שתיארנו בחלק 3, שמנו לב שחלק מהמודלים מתאימים יותר לנתונים שלנו (מבחינת AUC ו-accuracy), וחלקם - פחות. חלק ההערכה של ביצועי המודל דרש התאמה של המודלים על סט האימון והערכת ביצועים על סט הולידציה. רק לאחר מכן, ניתן היה לבחור מודל אופטימלי על סמך AUC. לשם ביצוע הליך זה, היה צורך בשימוש באלגוריתם Grid Search שעובר על כל מודל וכל שילובי ההיפרפרמטרים האפשריים עבורו. הליך זה עשוי היה להיות ארוך בזמנים ויקר מאוד חישובית, לכן החלטנו להשמיט חלק מהמודלים שביצועיהם בסט האימון (ברמת ה-AUC וה-accuracy הממוצע ב-K-Fold CV) היו יחסית נמוכים, והיו יקרים יחסית חישובית - לכן החלטנו לא להמשיך לבחון את הביצועים של KNN ו-MLP משלב זה והלאה.

לאחר מכן, ביצענו את ה-grid search, אימנו את 4 המודלים על סט האימון המורחב (מורחב במובן שאין cross validation הפעם), ובדקנו את הביצועים שלהם על סט הולידציה. היה צורך בשימוש באלגוריתם זה כדי למצוא את שילוב ההיפרפרמטרים המתאים ביותר לנתונים שעל סט הולידציה, במובן ששילוב זה ממקסם את ה-AUC ביחס לכל שאר שילובי ההיפרפרמטרים האפשריים עבור אותו המודל. בנספחים - התוצאות (Confusion Matrix + מטריקות רלוונטיות על כל מודל יחד עם עקומת ROC עם AUC).[9+10+11]





	Model	best_params	Accuracy	Precision	Recall	F1 Score	AUC
0	Logistic Regression	{'C': 0.1, 'penalty': 'l2', 'solver': 'libline...	0.916968	0.00000	0.000000	0.000000	0.665480
1	Random Forest	{'criterion': 'gini', 'max_depth': 10, 'max_fe...	0.918321	0.80000	0.021739	0.042328	0.656654
2	Decision Tree	{'criterion': 'entropy', 'max_depth': 10, 'max...	0.917419	0.52381	0.059783	0.107317	0.574887
3	SVM	{'C': 1, 'gamma': 'scale', 'kernel': 'rbf'}	0.917870	1.00000	0.010870	0.021505	0.640826

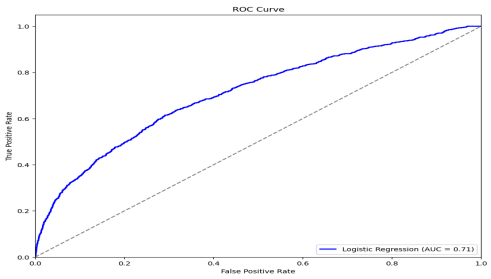
נשים לב כי בדומה לתוצאות בשלבים הקודמים, ל-Logistic Regression יש את ה-AUC הכי טוב, בפער זעום על שאר המודלים המתחרים (בטווח 0.65-0.7, באיטרציה אשר רואים בדו"ח - 0.66), רגרסיה לוגיסטית עבדה טוב באופן דומה לשלבים קודמים, כך גם Random Forest, עם ה-AUC השני הטוב ביותר על סט הולידציה. גם הביצועים של SVM נשארו דומים (מבחינת AUC), בטווח 0.6-0.65 כמו בשלבים קודמים. הביצועים של עץ ההחלטה נשארו דומים יחסית. המודלים שימרו את הדיוק שלהם משלבים קודמים (סביב 0.9), מה שמוכיח שהם ברובם טובים מאוד בסיווג נכון של רוב התצפיות. ניתן לשים לב לכך גם ב-confusion matrix של כל מודל. עם זאת, ממבט קצר על ה-confusion matrices, ניתן לראות כי המודלים בקושי מסווגים תצפיות חיוביות כחיוביות, מה שהוביל ל-recall נמוך יחסית אצל רובם. תוצאות כל המודלים בשלב זה, כשהם נבחנים על סט נתונים חדש דומה, דומות במיוחד לתוצאות בשלבים קודמים -לדוגמה, כל המטריקות הממוצעות ב-K-Fold CV של כל מודל כמעט שקולות למטריקות התואמות בשלב זה. (מה שמעיד על כך שאין פער דרסטי בין שלב אימון המודלים לשלב המבחן, כפי שנהוג ב-overfitting, וכן על בחירת K נכונה במובן מסוים, שכן בזמן אימון המודלים, ובחינת ביצועיהם על סטי האימון, לא היתה הטייה משמעותית יחסית לביצועי המודלים על סט הולידציה. וכן, על סמך מספר הרצות, ביצועי המודלים לא השתנו משמעותית - מה שמעיד על שונות נמוכה, ולכן - בחירת ה-K = 10 היתה ככל הנראה נכונה). לסיכום, על סמך מטריקת ה-AUC על סט הולידציה (שהיא אינטואיטיבית מייצגת סיווג טוב שכן ככל שעקומת ה-ROC תיראה דומה לריבוע 1*1, ותגיע לנק' (0,1), וכך השטח מתחתיה - ה-AUC, יתקרב ל-1, אז יכולת הסיווג של המודל על פניו טובה יותר), נבחר במודל הרגרסיה הלוגיסטית לשלבי ההמשך של הפרויקט. לאחר שאימנו אותו על כל סט האימון והולידציה, קיבלנו AUC גבוה משמעותית משלבים קודמים - 0.93! ואותה ה-accuracy (בערך)! גם הפעם, המודל בקושי מסווג תצפיות כחיוביות מה שגרר recall נמוך. ה-precision שלו בינוני - 0.5 - מה שאומר שהוא סיווג חצי מהתצפיות החיוביות נכון וחצי באופן שגוי. (עקומת ה-ROC וה-AUC של מודל הרגרסיה הלוגיסטית על סט האימון המלא, ה-Confusion Matrix שלו והמטריקות הרלוונטיות - בנספח)[12+13+14]

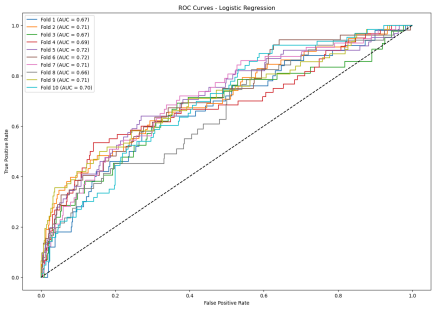
```
Best Model: Logistic Regression
Best penalty: l2
Best C: 0.1
Best solver: liblinear
```

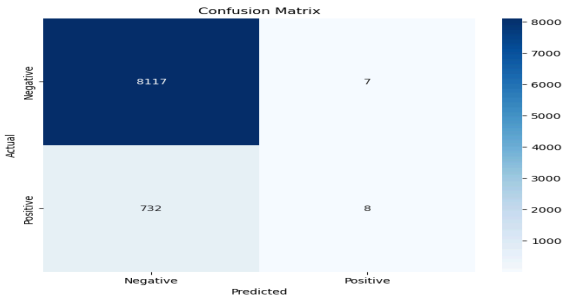
```
Accuracy: 0.92
Precision: 0.53
Recall: 0.01
F1 Score: 0.02
AUC: 0.71
```

בנוסף, ההיפרפרמטרים הטובים ביותר שהתקבלו עבור המודל הטוב ביותר (רגרסיה לוגיסטית, גם בנספח)[15]

פונק' הקנס שנבחרה היא מסוג Ridge Regression (L2), שפרופורציונית לסכום ריבועי מקדמי הרגרסיה. גודל קבוע ההופכי לקנס (C = 1/lambda) הקנס שנבחר הוא C = 0.1, כך שהקבוע שמבטא את חשיבות פונק' הקנס במזעור של בעית הרגרסיה הוא lambda = 10.הקבוע שמבטא את חשיבות פונק' הקנס בבעית המזעור במודל מבטא את חוזק הרגולריזציה (ככל ש-C קטן, lambda גדל ובעית הרגרסיה "מתמקדת" יותר במזעור סכום ריבועי מקדמי הרגרסיה, ביחס לבעית הרגרסיה ה-"מקורית"), "מתמרץ" את המקדמים להתכווץ ולהתקרב ל-0 אבל לא בהכרח להיות זהותית 0. מה שמונע overfitting ולא פוגע ביכולת ההכללה של המודל. בנוסף, עקומות ה-ROC של המודל הנבחר (רגרסיה לוגיסטית) בשלב ה-K-Fold-CV, בנספח:[16]







חלק 5 - הפרדיקציות:

בשלב זה, חשפנו את מודל הרגרסיה הלוגיסטית, אשר אומן בסוף שלב 4 (עם אותם היפרפרמטרים משלב הבחינה על סט הולידציה, ואותם מקדמים כפי שאומנו על סט האימון המלא), לסט נתונים "דומה". חובה היה להחיל על סט המבחן טרנספורמציות הזהות לאלו שחלו על סט האימון. בפועל, משמעות הדבר היתה להחיל חלקים נרחבים משלבים 1+2 גם בשלב זה - טרנספורמציות על עמודות, השלמת חסרים, הסרת חריגים, OHE, נירמול.. ולאחר מכן, לבחון את ביצועי המודל המאומן על סט הטסט, ולשמור את פלטי התחזיות בקובץ.

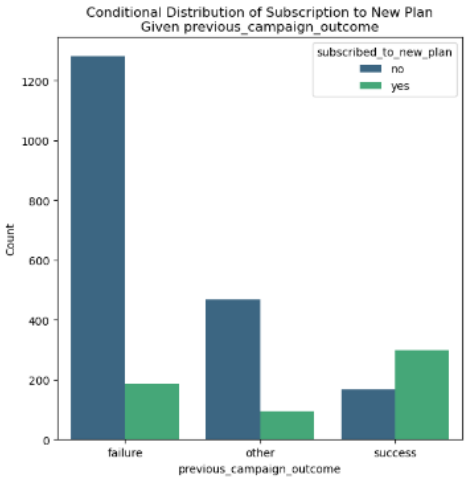
סיכום:

בפרויקט זה, נחשפנו לבעית סיווג מהעולם האמיתי. נאלצנו להתמודד איתה בעזרת אלגוריתמי למידת מכונה, תוך שימוש בשלל כלים שנלמדו בקורס. במהלך הפרויקט, צברנו תובנות משמעותיות על הנתונים ובחנו מגוון מודלי למידת מכונה כדי לנבא את הסבירות להרשמה לתוכניות. ניתוח הנתונים בשלב הראשוני היה הכרחי לצורך קבלת מידע חיוני שישמש אותנו בשלב 2 - העיבוד המקדים. ובשלב 2, כל צעד בשלב עיבוד הנתונים היה קריטי לצורך הכנה טובה של הנתונים למודלים. מהסרת החריגים, דרך השלמת הערכים החסרים, הטרנספורמציות ההכרחיות, עד לבחירת הפיצ'רים לצורך הורדת מימד הבעיה. כך גם בחירת המודל, וטיוב ההיפרפרמטרים בשלב 4 (פרמטרי הקנס - רגולריזציה ברגרסיה הלוגיסטית), לצורך הפחתת overfitting, מה שתרם לביצועים טובים על נתונים דומים שלא נראו קודם (יכולת הכללה טובה). במהלך כל השלבים, הרצות המודלים הצביעו על

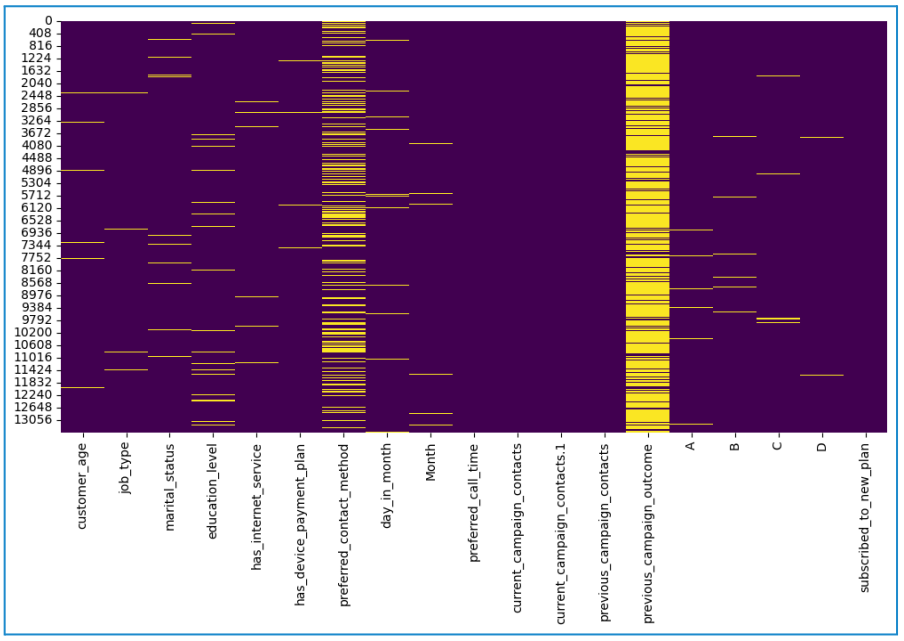
ביצועים טובים של הרבה מהם, ובפרט מודל ה-Random Forest והרגרסיה הלוגיסטית אותו בחרנו (להם היו ביצועים דומים מאוד מבחינת AUC ו-accuracy, בכל שלבי האימון שיכולנו להשוות ביניהם). הן מבחינת AUC והן מבחינת דיוק.

נספחים – תמונות בגדול:

1. התפלגות Previous-Campaign-Outcome בהנתן Subscription-to-new-plan (ניתן להסתכל על זה הפוך):



2. קורלציה בין עמודות ה-dataframe לערכים חסרים:



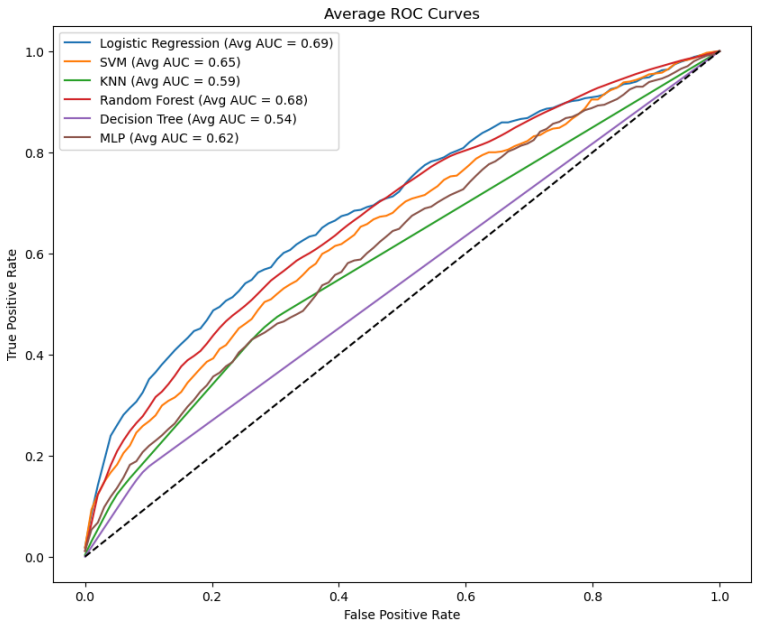
3. אחוזי חסרים ב-"Previous Campaign Outcome":

Dropping 'previous_campaign_outcome'

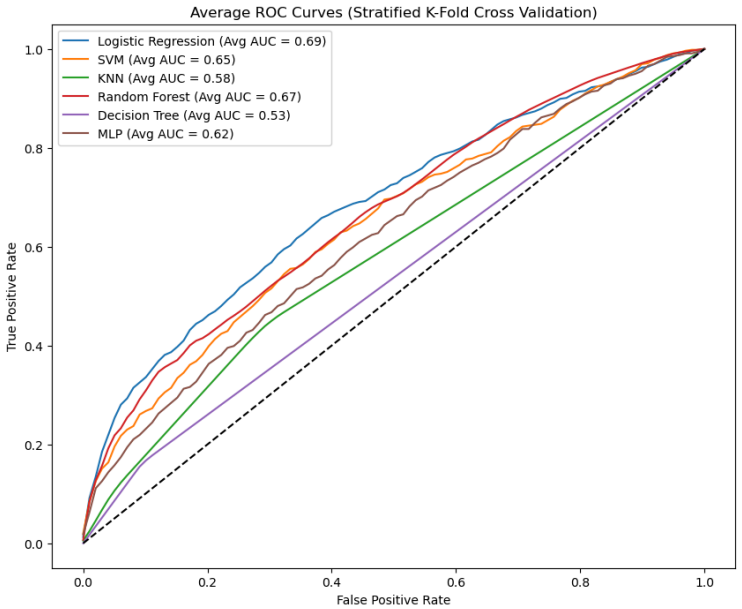
```
# Getting the exact percentage of nulls in the 'previous_campaign_outcome' column:
df['previous_campaign_outcome'].isnull().sum()/df.shape[0]*100
```

```
81.44237918215613
```

4. עקומות ROC ממוצעות - K-Fold-Cross Validation (K = 10):



5. עקומות ROC ממוצעות - Stratified K-Fold-Cross Validation (K = 10):



:

6. המטריקות הממוצעות של כל מודל - K-Fold Cross Validation (K = 10):

	Model	AUC	Accuracy	Recall
0	Decision Tree	0.539400	0.847319	0.169776
1	KNN	0.594493	0.913054	0.019098
2	Logistic Regression	0.694205	0.915761	0.044652
3	MLP	0.615753	0.878758	0.147009
4	Random Forest	0.676507	0.916363	0.022212
5	SVM	0.649195	0.916062	0.000000

7. המטריקות הממוצעות של כל מודל - Stratified K-Fold Cross Validation (K = 10):

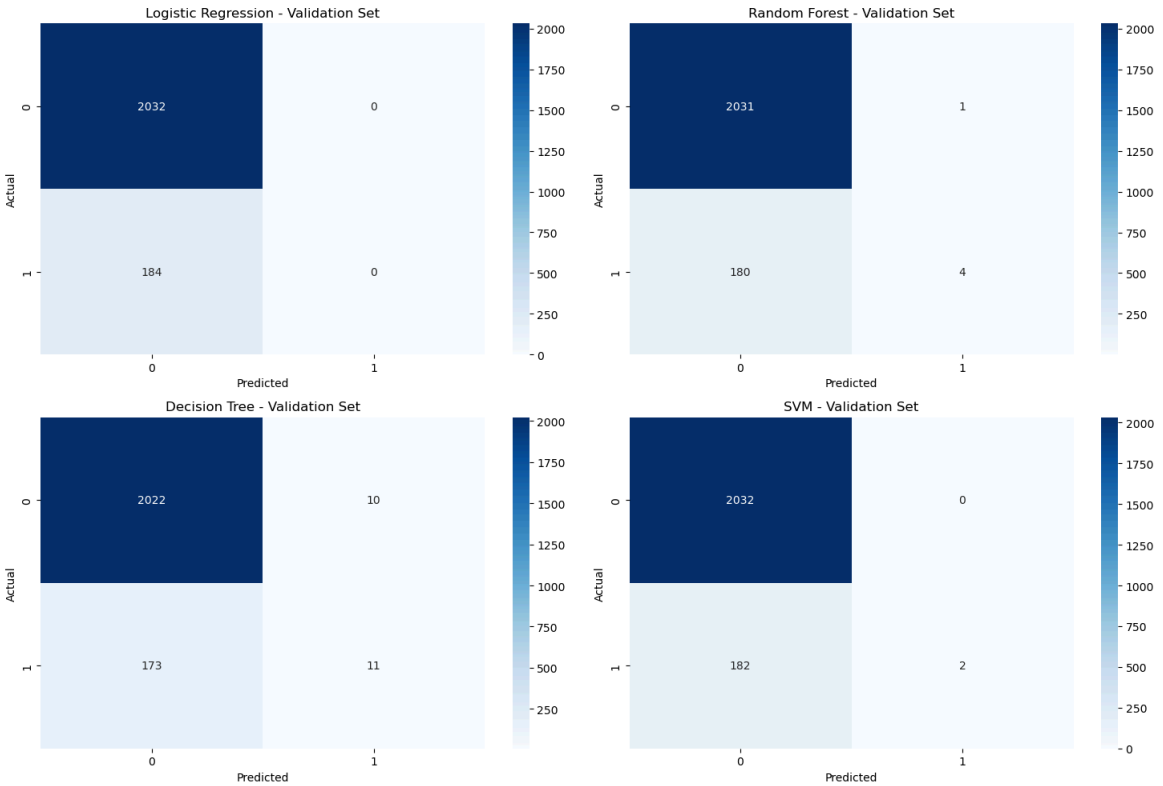
	Model	AUC	Accuracy	Recall
0	Decision Tree	0.533868	0.847021	0.157857
1	KNN	0.580270	0.913056	0.014416
2	Logistic Regression	0.685502	0.915915	0.043182
3	MLP	0.624534	0.884177	0.154448
4	Random Forest	0.667017	0.915765	0.023377
5	SVM	0.649801	0.916065	0.000000

8. הביצועים הממוצעים של המודלים - השוואה בין 2 שיטות ה-Cross Validation:

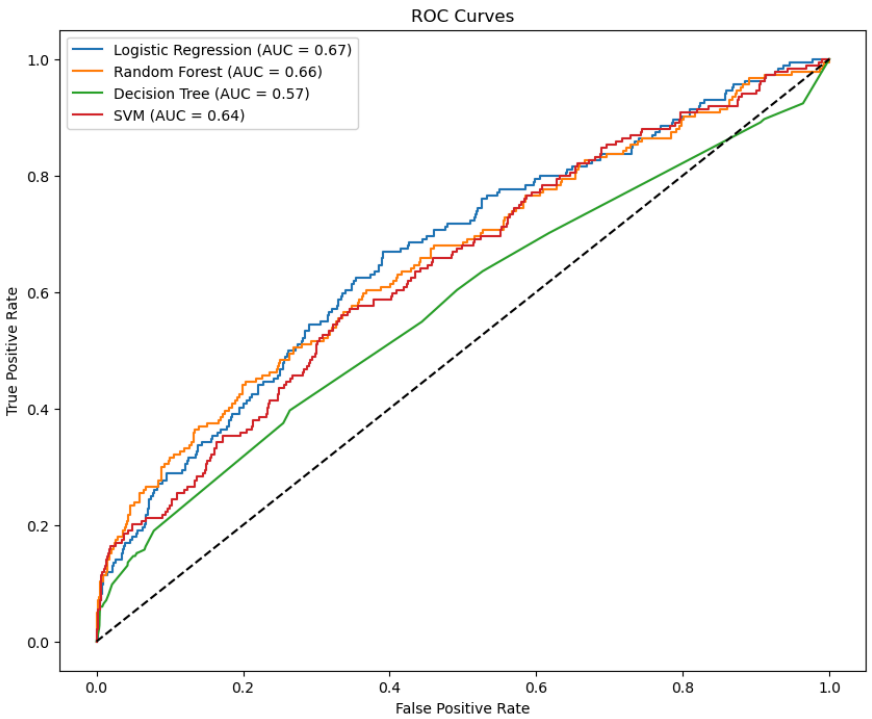
```
# Checking the difference in metrics between the models on the original K-Fold CV, and the ones on the stratified K-Fold CV:
print(avg_metrics.loc[:, ["AUC", "Accuracy", "Recall"]] >= avg_stratified_metrics.loc[:, ["AUC", "Accuracy", "Recall"]])
```

```
     AUC  Accuracy  Recall
0   True      True    True
1   True     False    True
2   True     False    True
3  False     False    True
4   True     False   False
5  False     False    True
```

9. Confusion Matrix של כל מודל על סט הולידציה:



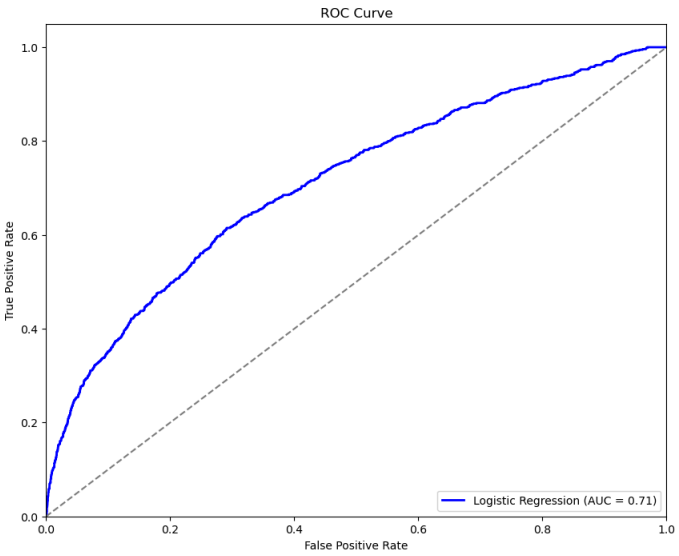
10. עקומות ROC של כל מודל על סט הולידציה:



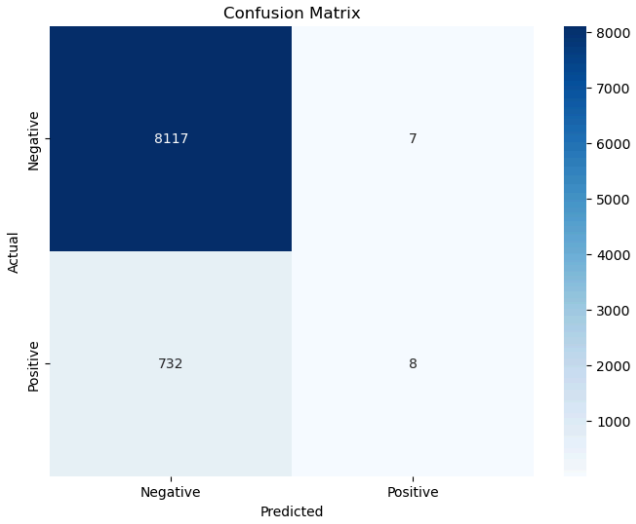
11. מטריקות המודלים על סט הולידציה:

	Model	best_params	Accuracy	Precision	Recall	F1 Score	AUC
0	Logistic Regression	{'C': 0.1, 'penalty': 'l2', 'solver': 'libline...	0.916968	0.00000	0.000000	0.000000	0.665480
1	Random Forest	{'criterion': 'gini', 'max_depth': 10, 'max_fe...	0.918321	0.80000	0.021739	0.042328	0.656654
2	Decision Tree	{'criterion': 'entropy', 'max_depth': 10, 'max...	0.917419	0.52381	0.059783	0.107317	0.574887
3	SVM	{'C': 1, 'gamma': 'scale', 'kernel': 'rbf'}	0.917870	1.00000	0.010870	0.021505	0.640826

12. עקומת ה-ROC של Random Forest על סט האימון המלא:



13. Confusion Matrix של Random Forest על סט האימון המלא:



14. המטריקות של המודל הטוב ביותר (Random Forest) - תוך אימון על כל ה-dataset של ה-train:

```
Accuracy: 0.92
Precision: 0.53
Recall: 0.01
F1 Score: 0.02
AUC: 0.71
```

15. ההיפרפרמטרים הטובים ביותר אשר התקבלו עבור המודל הטוב ביותר (Random Forest):

```
Best Model: Logistic Regression
Best penalty: l2
Best C: 0.1
Best solver: liblinear
```

16. עקומות ה-ROC של המודל הנבחר (רגרסיה לוגיסטית) בשלב ה-K-Fold-CV: