כריית מידע וייצוג מידע* – פרויקט חלק ב'

גילעד סבוראי ולירי בנזינו

* כל הגרפים והנתונים הסטטיסטיים נמצאים במחברת.

1) עיבוד קדם pre-process

כמו בחלק הקודם ביצענו pre-process גם על המידע החדש שלנו, ה-test. ניתוח דומה התבצע גם ל-train מהחלק הקודם.

תיקנו את הדברים הבאים מהחלק הקודם –

1. ראשית לא ביצענו PCA כי פחות נוח לעבוד עם נתונים אלו כי נאבדים לנו השמות של ה-feature-ים והמשמעות שיש להם בתוך ה-dataset המקורי. לכן נעבוד עם הנתונים לפני ביצוע ה-PCA.
2. כאשר לחברת הזנק ישנם ערכים חסרים בעמודות first/last_milestone_age, זה אומר שלמעשה לחברת ההזנק לא היו כלל אבני דרך, ולכן לא היה נכון למלא שם ערכים על פי רגרסיה לינארית.
לכן מילאנו עם הנוסחה הבאה:

$$\text{minOfColumn} - 10$$

אשר גם מתאר חוסר משמעות/הפרדה עבור המסווגים בערך זה, וגם לא יוצר הטיה לעמודות (בניגוד להכנסת הערך מינוס אינסוף).

3. החלטנו להחזיר את העמודות המייצגות longitude-latitude, ובנוסף שמנו לב לערכים חריגים בעמודות אלו – מדינות אשר אינן נמצאות בארה"ב. כיוון שהיו שורות מועטות שהיו חריגות, החלטנו להסיר אותן מה-datasets שלנו.

לבסוף עבדנו רק עם ה-datasets הנקיים :)

2) סיווג, מטריצות הערכה וחיפוש hyper-parameters

בחלק זה סיווגנו את המידע שלנו על ידי חמישה מודלי סיווג שונים –

`Decision Tree, Random Forest, Naïve Bayes, SVM, K-NN`

המודל שלא נלמד בהרצאה/בתרגול הינו K-NN.

את הסיווג הערכנו על ידי חמש מטריקות שונות –

`Accuracy, Precision, Recall, F-Score, AUC`

בחרנו במטריקות האלו כי לדעתנו הן נותנות לנו הרבה מידע על ביצועי המסווג. מדד ה-Accuracy מתאר לנו כמה מהתוצאות שהמסווג נותן הינן נכונות על בסיס ה-ground truth. מדד זה טוב על מנת להבין עד כמה המסווג מדייק בהחלטתו/מה הסיכוי שהמסווג טעה בסיווג.
כאשר חשוב לנו שהחלטת המסווג תהיה כמה שיותר נכונה (גם עבור סיווג חיובי וגם עבור סיווג שלילי), נרצה לדרג מודלים על פי מדד זה.

מדד ה-Precision מתאר מה ההסתברות שלנו להצליח לסווג נכון חברה שתצליח, בהינתן שסיווגנו שהיא תצליח. כמה מבין החברות שסיווגנו חיוביות באמת תפסנו נכון.
כאשר אין לנו הרבה משאבים ונרצה שכמה שיותר סיווגים שלנו שהם חיוביים יהיו באמת חיוביים, נרצה שמדד זה יהיה גבוה.

מדד ה-Recall הנקרא גם Sensitivity מתאר כמה הצלחנו לסווג כחיובי מתוך כל אוכלוסיית החיוביים.
נרצה שמדד זה יהיה גבוה כאשר אנו מעוניינים לתפוס כמה שיותר מאוכלוסיית החיוביים.

F-Score הינו ממוצע משוקלל בין מטריקת ה-Precision וה-Recall. הוא נותן הערכה מאוזנת לשני המדדים האלו ורלוונטי כאשר אין לנו העדפה מיוחדת עבור אף אחד משני אלו.

Area Under Curve הינו השטח שכלוא מתחת לגרף ה-ROC ונרצה שהשטח יהיה כמה שיותר קרוב לאחת מכיוון שאז המודל שנבחר הינו איכותי יותר ופחות רגיש לשינויים. גרף ה-ROC מציג את ביצועי המודל בכל סף סיווג, לכן נותן מדד על איכות המסווג.

לדעתנו המטריקה שהכי מייצגת את טיב המסווג ושעל פיה נרצה להעדיף מודל על פני רעיהו הינה Accuracy, כי היא מתארת בכמה צדקנו – ולמעשה זה מה שמעניין אותנו, כמה חברות יצליחו בסוף, ונרצה להיות כמה שיותר צודקים כדי לתת לחברה שפונה אלינו את התשובה המדויקת ביותר בין אם היא תצליח או תיכשל. בנוסף גם מדד ה-AUC משמעותי עבורנו, כי הוא מתאר את איכות המסווג – ככל שהוא איכותי יותר, הערך גבוה יותר (מתקרב ל-1). כמובן שאת ההשוואה נבצע על ידי מבחן סטטיסטי t בהמשך.

לפני שלבי הלמידה הסתכלנו על הערכים שלנו וראינו כי הם אינם מאוזנים – ישנם הרבה יותר חברות הזנק שהן acquired מאשר closed. דבר זה יכול ליצור אצלנו הטיה בהחלטות המסווג, ולכן השתמשנו ב-SMOTE על מנת לבצע oversampling על אוכלוסיית ה-Closed ובכך להגיע לאיזון בין המחלקות השונות. אלגוריתם SMOTE מבצע oversampling על ידי אינטרפולציה לינארית בין נקודות קיימות, באזורים של מחלקת המיעוט. תחילה הרצנו אותו על הנתונים הנקיים, עם פרמטר `k_neighbors=8` כלומר האלגוריתם משתמש בסביבות של 8 שכנים על מנת לייצר את המידע הסינטטי.

לאחר הרצת המסווגים עם ובלי SMOTE, שמנו לב שאין שיפור בין ההרצות (ולעיתים אף גירעון בדיוק). לכן אימנו את המסווגים על הנתונים ללא ה-SMOTE, הלא מאוזנים אותו (ניתן לראות במחברת).

על מנת לבחור את המודל איתו נשתמש בסוף, בחנו ובדקנו חמישה סוגי מסווגים שונים (כנזכר לעיל).

את המסווג הסופי נבחר על פי התהליך הבא:

א. עבור כל אחד מהמסווגים הבאים [Decision Tree, Random Forest, Naïve Bayes, SVM, K-NN]
   1. חישוב מטריקות עבור מסווג הבסיס base (כלומר מסווג עם פרמטרים ברירת מחדל) באמצעות cross-validation עם חמישה folds (k=5). בשיטה זו אנו מחלקים את הנתונים ל-k חלקים שונים. בכל פעם אנו מבצעים אימון (train/fit) על k-1 חלקים ומבצעים הערכה (validation) על החלק הנותר. מתוך תהליך זה אנו מקבלים k מטריקות שונות, ובסוף נמצע אותן כדי להעריך את המטריקות עבור המסווג בכללותו.
   2. ריצה על פרמטר מסוים באופן ידני על מנת ללמוד את ההשפעה שלו על שאר, כאשר הפרמטר הזה משתנה מתוך פרמטרי ברירת המחדל.
   3. חיפוש פרמטרים על ידי random search
   4. חיפוש פרמטרים על ידי grid search
   5. השוואת הדיוק של המודלים שהתקבלו על ידי שני הסעיפים הקודמים (random & grid)
   6. השוואת ה-ROC curve של שלושת המודלים הנ"ל
   7. בחירת המודל הכי טוב מבין אלו שבדקנו על ידי ביצוע t-test על כל אחת מהמטריקות (עם התייחסות ל-accuracy כמטריקה הקובעת).

ב. השוואה בין חמשת המסווגים על ידי ביצוע t-test – בתחילת שלב זה יש לנו חמישה מסווגים מסוגים שונים, נבחר את המודל הכי טוב מבין כל המודלים הכי טובים שקיבלנו בשלבים קודמים על ידי ביצוע t-test באותן אופן כמו בשלב א'.

ג. בחירת המסווג היחיד הטוב ביותר – בעזרתו נחזה את ה-Target עבור נתוני הבוחן (test).

* עבור כל אחד מהמסווגים הוספנו dummies (עבור עמודות עם בין 3 ל-8 ערכים שונים – כלומר ערכים שעברו דיסקריטיזציה ואינם בינאריים), וראינו שהוספת ה-dummies אינה שיפרה לנו את הדיוק כלל (או אף אחת מהמטריקות האחרות), לכן בחרנו להמשיך בלעדיהם.

* ה-t-test שלנו מניח את הנחת האפס ששני המודלים שאין בניהם מובהקות סטטיסטית (עשינו מבחן דו-צדדי), ואם קיבלנו דחייה להנחה הזו, אז קבענו שמודל "מנצח" אם יש לו מטריקה טובה יותר.

* בחלק מהמסווגים הגדרנו טווח פרמטרים גדול יותר ל-random search מאשר ל-grid search, ולמעשה את טווח הפרמטרים ל-grid search הגדרנו על פי התוצאות של ה-random search. עשינו את זה כי ה-grid search בודק את כל האופציות האפשריות אבל זה לוקח הרבה זמן (ולעיתים גם הוא הגיע לתוצאה פחות טובה מאשר ה-random), לכן השתמשנו ב-random על מנת לצמצם ולדייק את החיפוש.

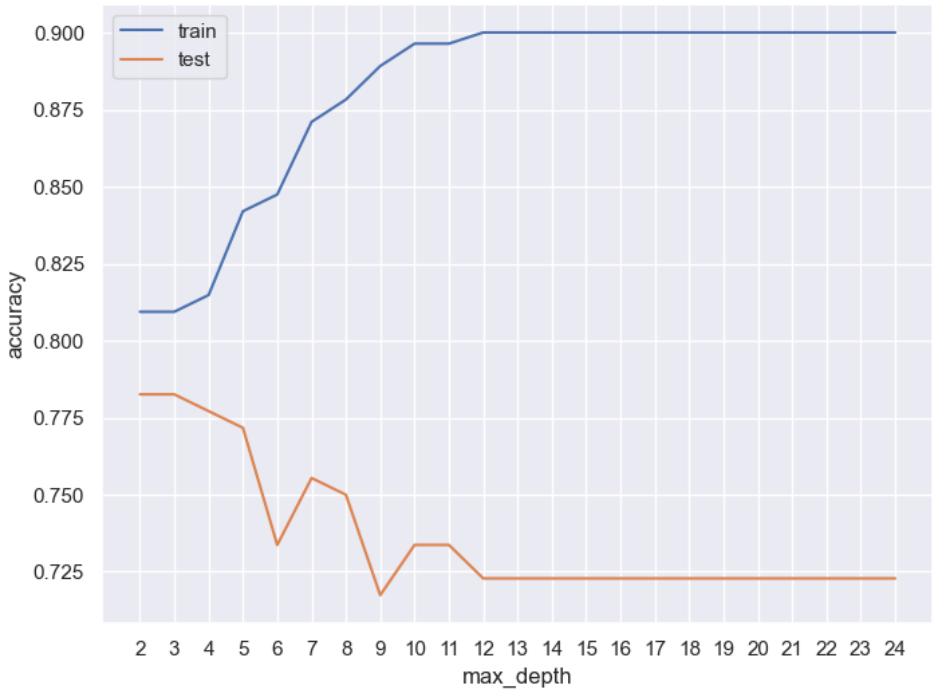
* כשהרצנו את המחברת מחדש הרקע של הגרפים השתנה, אבל אלו אותם גרפים כמו פה בדוח.

מודל ראשון - Decision Tree

במודל זה אנחנו בונים עץ כך שבכל הסתעפות אנחנו מסתעפים על פי ה-feature עם entropy/gini המקסימלי בנקודה זו בעץ.

כאשר הרצנו פעם ראשונה cross validation על מודל הבסיס, ראינו שקיבלנו overfitting על ה-train, ועל ה-test קיבלנו דיוק של כ-73%. דבר זה הגיוני, כי קיבלנו עץ מעומק מקסימלי אשר מותאם רק ל-train ולא לאף מידע אחר.

לכן, רצינו לראות את ההשפעה של הפרמטרים של העץ על הדיוק שלו. לעץ ישנם הרבה פרמטרים, ועל כן ראשית ניסינו "לשחק" עם עומק העץ, ולראות את ההשפעה של פרמטר זה על דיוק העץ שלנו. קיבלנו את הגרף הבא –



אנחנו יכולים לראות כי כמו שהבחנו במקרה של העץ ברירת המחדל – כאשר עומק העץ גדול יותר, כך הדיוק על ה-train מתקרב יותר ל-1, ובפרט נקבל overfitting.

על פי הגרף לעיל, ראינו כי עבור עומק 4 הדיוק על ה-test (שנוצר על ידי cross validation) היה מקסימלי. לא רצינו לבחור על פי הדיוק של ה-train כדי למנוע overfitting. כאשר חישבנו את המטריקות מחדש עבור עץ עם עומק 4 ראינו כי קיבלנו דיוק הרבה יותר טוב על ה-test, כ-79.6%.

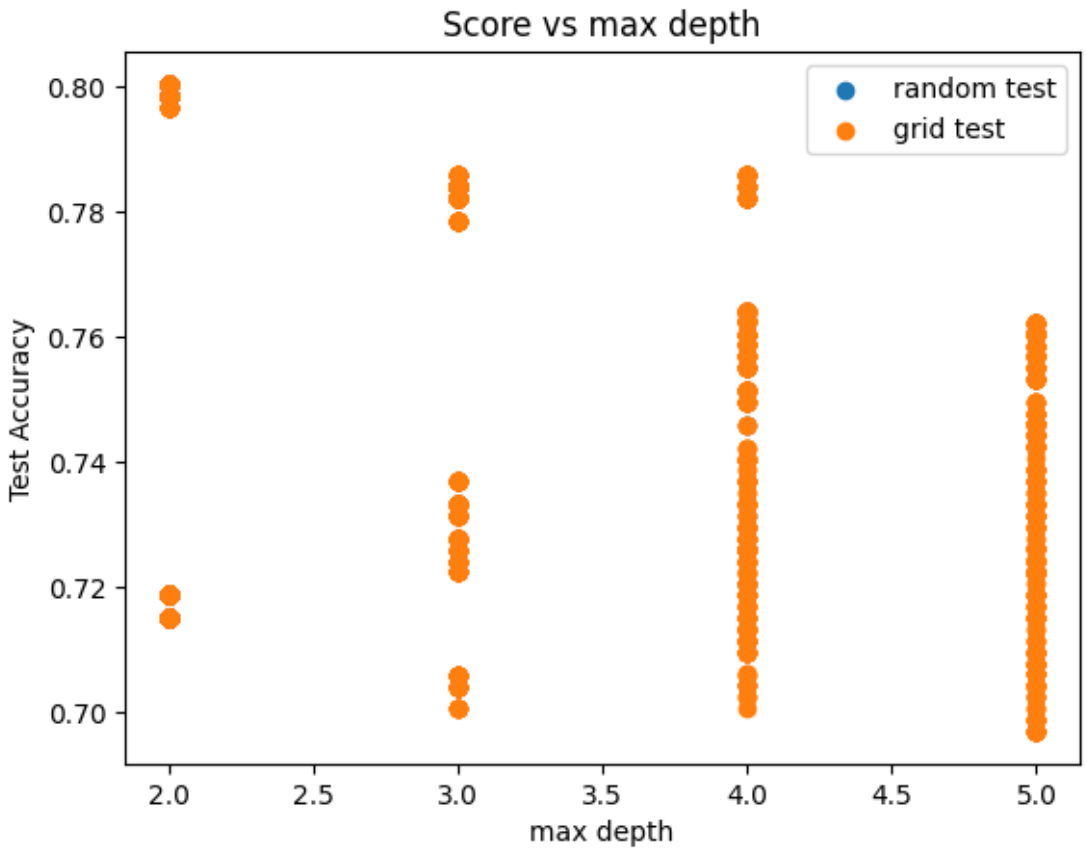
לאחר מכן, הרצנו random-search ו-grid-search על מנת למצוא שילוב פרמטרים אידיאליים למסווג שלנו (כי עומק מקסימלי אינו מספיק). הפרמטרים שעליהם הרצנו את החיפוש נקבעו על פי הגרף לעיל, על פי אינטואיציה שלנו בנוגע לעץ, ועל ידי ניסוי וטעיה. טווחי הפרמטרים נתונים בקובץ המחברת.

עבור שני החיפושים קיבלנו אומנם תוצאות שונות (מבחינת הפרמטרים), אך שתיהן נתנו לנו אותו דיוק (וגם אותו עץ, כפי שניתן לראות במחברת) –

```
[base] model accuracy = 71.20%
[random] model accuracy = 78.26%
Improvement of 9.92%
```

```
[base] model accuracy = 71.20%
[grid] model accuracy = 78.26%
Improvement of 9.92%
```

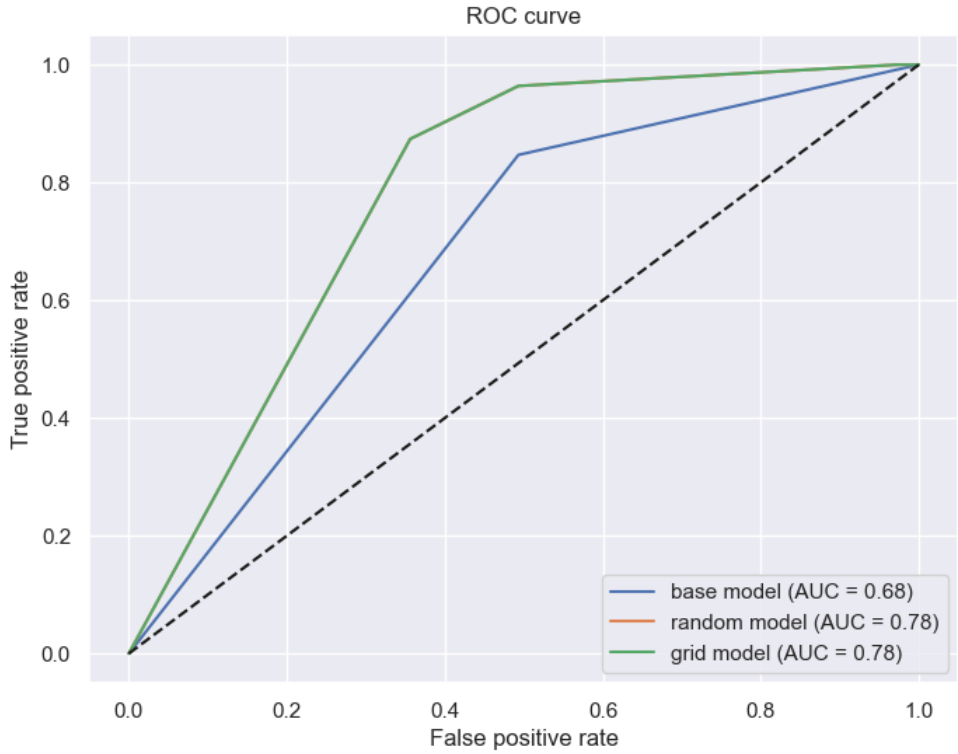
בנוסף, רצינו לראות את ההשפעה של הלמידה על הדיוק, ובחנו את זה על ידי השפעת הפרמטר של עומק העץ שלנו (כמו קודם), קיבלנו –



אפשר לראות כאן מגמה דומה לגרף הקודם – הדיוק המקסימלי מתקבל עבור ערכים נמוכים, וקטן ככל שהעץ מתעמק.

וכמו שאפשר לראות שניהם אכן שיפרו לנו את הדיוק מודל הבסיס, זה הגיוני כי אלו מודלים שעברו אופטימיזציה על הפרמטרים במטרה לקבל score כמה שיותר גבוה.

בנוסף, השוונו גם בין המודלים על ידי ROC curve ו-AUC וראינו כי גם כאן ה-random וה-grid נתנו תוצאות טובות יותר מה-base.



לבסוף, השוונו בין שלוש המודלים שלנו על ידי t-test וראינו כי כמו שחשבנו עד כה, המודלים שנוצרו על ידי ה-random-search וה-grid-search נתנו את התוצאות הכי טובות.

```
--------- metric for ttest: accuracy ---------
p-value: 0.03212201726016634
reject null hypothesis => random is BETTER THAN base
p-value: 0.03212201726016634
reject null hypothesis => grid is BETTER THAN base
random and grid have the same metric => No one is better than the other
```

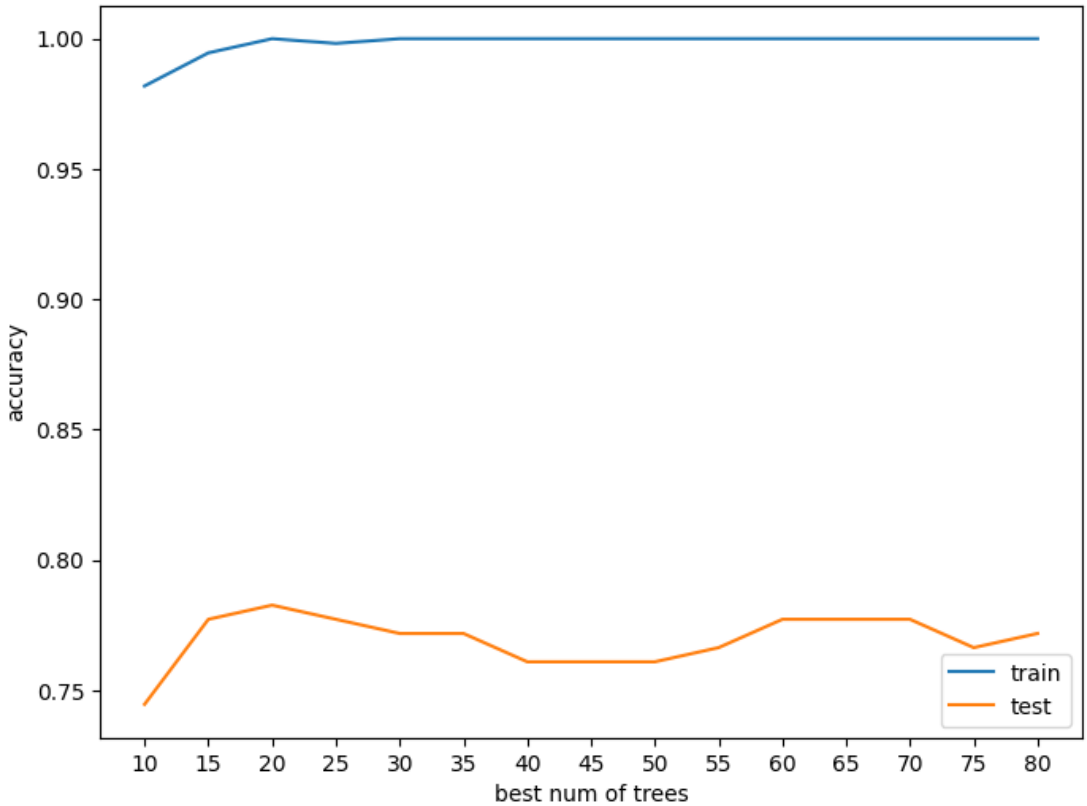
כמו שניתן לראות שני המודלים הללו נתנו אותה תוצאה עבור accuracy (ולמעשה גם לכל מטריקה אחרת), לכן רנדומית החלנו להמשיך עם ה-grid, ואחרי חישוב accuracy עם cross validation קיבלנו דיוק של 79.7%.

מודל שני - Random Forest

במודל זה אנחנו בונים יער עצים כך שכל עץ נבנה כמו Decision Tree – בכל הסתעפות אנחנו מסתעפים על פי ה-feature עם entropy/gini המקסימלי בנקודה זו בעץ.

כאשר הרצנו פעם ראשונה cross validation על מודל הבסיס, ראינו שקיבלנו overfitting על ה-train, ועל ה-test קיבלנו דיוק של כ-77.7%. דבר זה הגיוני, כי קיבלנו עץ מעומק מקסימלי אשר מותאם רק ל-train ולא לאף מידע אחר.

לכן, כמו עבור המסווג הקודם גם כאן רצינו לראות את השפעת הפרמטרים השונים של המסווג על המטריקות השונות. ואכן כמו לעץ, גם ליער העצים ישנם הרבה (ואף יותר) פרמטרים, ועל כן ראשית ניסינו "לשחק" עם מספר העצים ביער, ולראות את ההשפעה של פרמטר זה על דיוק המסווג שלנו. קיבלנו את הגרף הבא –



אנחנו יכולים לראות כי כמו שהבחנו במקרה של העץ ברירת המחדל – כאשר עומק העץ גדול יותר, כך הדיוק על ה-train מתקרב יותר ל-1, ובפרט נקבל overfitting. אנחנו יכולים לראות שהגענו מהר מאוד ל-overfit אפילו עבור שבעים עצים. דבר זה הגיוני כי נשים לב שיש לנו כ-730 sampling ב-dataset ולכן רק כ-550 משתמשים לבדיקת המודל, לכן אם יהיו לנו הרבה עצים המודל יהיה תואם אך ורק לערכים מהם הוא למד ולא לערכים חדשים.

על פי הגרף לעיל, ראינו כי עבור 20 עצים הדיוק על ה-test (שנוצר על ידי cross validation) היה מקסימלי. לא רצינו לבחור על פי הדיוק של ה-train כדי למנוע overfitting. כאשר חישבנו את המטריקות מחדש עבור יער עם 20 עצים ראינו כי קיבלנו דיוק יותר טוב על ה-test, כ-78%.

לאחר מכן, הרצנו random-search ו-grid-search על מנת למצוא שילוב פרמטרים אידיאליים למסווג שלנו (כי מספר עצים אינו מספיק). הפרמטרים שעליהם הרצנו את החיפוש נקבעו על פי

הגרף לעיל, על פי אינטואיציה שלנו בנוגע לעץ, ועל ידי ניסוי וטעיה. טווחי הפרמטרים נתונים בקובץ המחברת.

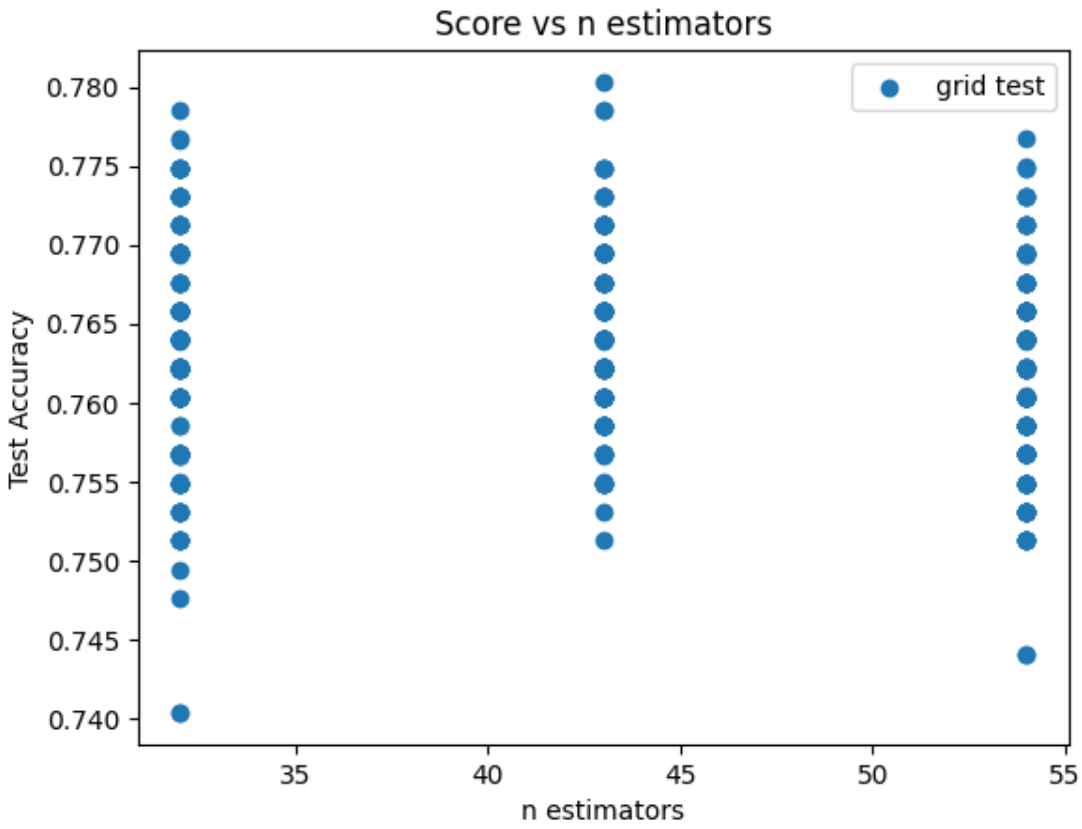
עבור שני החיפושים קיבלנו אומנם תוצאות שונות (מבחינת הפרמטרים), אך שתיהן נתנו לנו אותו דיוק (וגם אותו עץ, כפי שניתן לראות במחברת)

```
[base] model accuracy = 76.63%
[random] model accuracy = 76.63%
Improvement of 0.00%
```

```
[base] model accuracy = 76.63%
[grid] model accuracy = 77.72%
Improvement of 1.42%
```

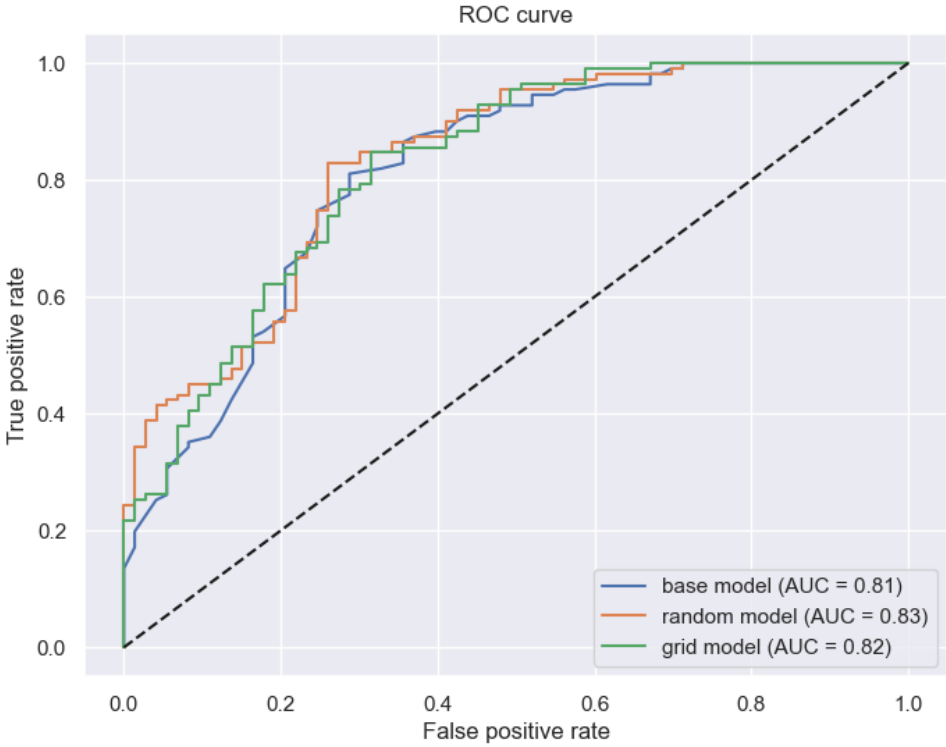
וכמו שאפשר לראות שניהם אכן שיפרו לנו את הדיוק מודל הבסיס, זה הגיוני כי אלו מודלים שעברו אופטימיזציה על הפרמטרים במטרה לקבל score כמה שיותר גבוה.

גם כאן רצינו לראות את ההשפעה של הלמידה על הדיוק, ובחנו את זה על ידי השפעת הפרמטר של מספר העצים (כמו קודם), קיבלנו –



אפשר לראות כאן מגמה דומה לגרף הקודם – ישנו "פיק" באזור של 43, והשאר מסביב לזה.

בנוסף, השוונו גם בין המודלים על ידי ROC curve ו-AUC וראינו כי גם כאן ה-random וה-grid נתנו תוצאות טובות יותר מה-base.



לבסוף, השוונו בין שלוש המודלים שלנו על ידי t-test וראינו כי כמו שחשבנו עד כה, המודלים שנוצרו על ידי ה-random-search וה-grid-search נתנו את התוצאות הכי טובות.

כמו שניתן לראות שני המודלים הללו נתנו אותה תוצאה עבור accuracy (ולמעשה גם לכל מטריקה אחרת), לכן רנדומית החלנו להמשיך עם ה-grid.

מודל שלישי - Naïve Bayes

במודל זה אנו נעזרים בנוסחת בייס ומניחים שאין תלות בין ה-features השונים בנתונים. כלומר בהינתן דוגמה X, נחפש מחלקה C_i עבורה ההסתברות $P(C_i|X)$ היא הגבוהה ביותר, כאשר נוכל להניח אי תלות בין ה-features נקבל:

$$P(C_i|X) = \frac{P(X|C_i) \cdot P(C_i)}{P(X)} = P(C_i) \cdot \frac{\prod P(x_k|C_i)}{P(X)}$$

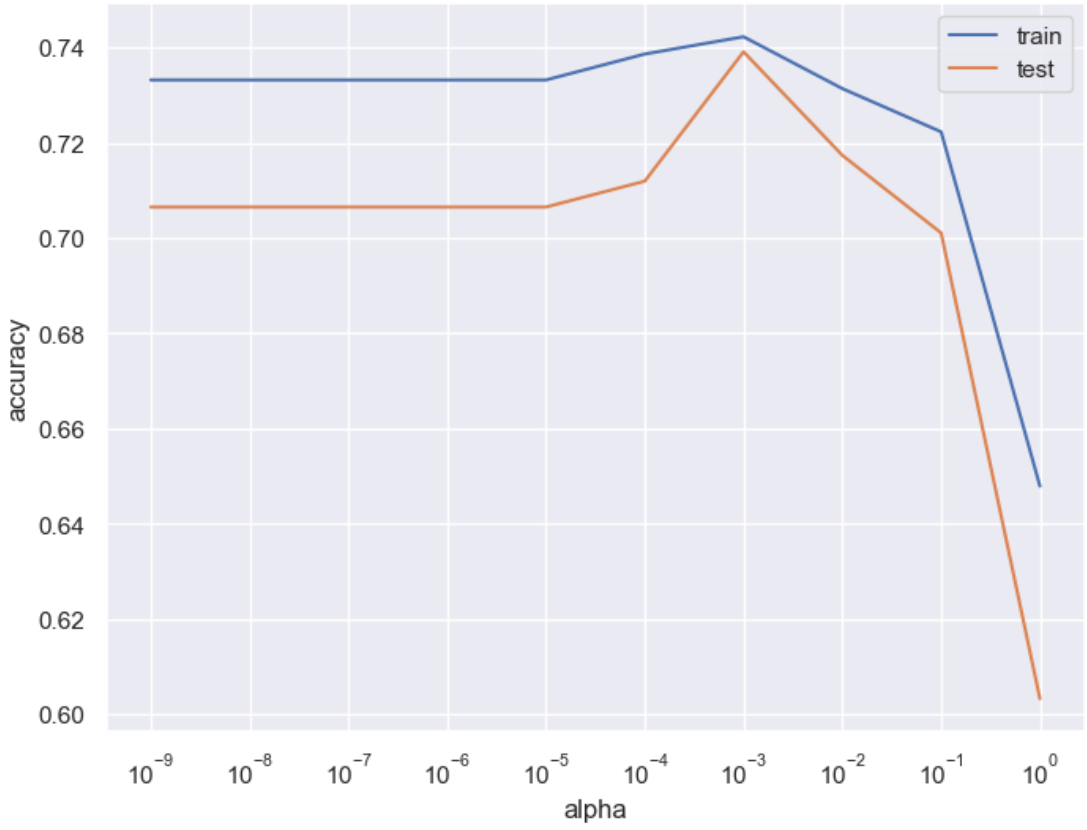
מודל זה קל ומהיר לשימוש וחישוב בשל הפשטות שלו.

השתמשנו במודל באסיאני שמשתמש בהתפלגות גאוסיאנית עבור *features* רציפים, על כן `Gaussian` בשם המסווג.

התחלנו בלהריץ את המסווג הבסיסי עבור הנתונים המקוריים ולחשב מטריקות שונות על מודל זה ע"י שימוש ב-*cross validation* וקיבלנו עבורו דיוק 71.9%.

ראינו שאין שיפור בהוספת ה-*dummies* (ניתן לראות גם במחברת הדיוק ירד דרסטית) לכן המשכנו שוב בלעדיהם.

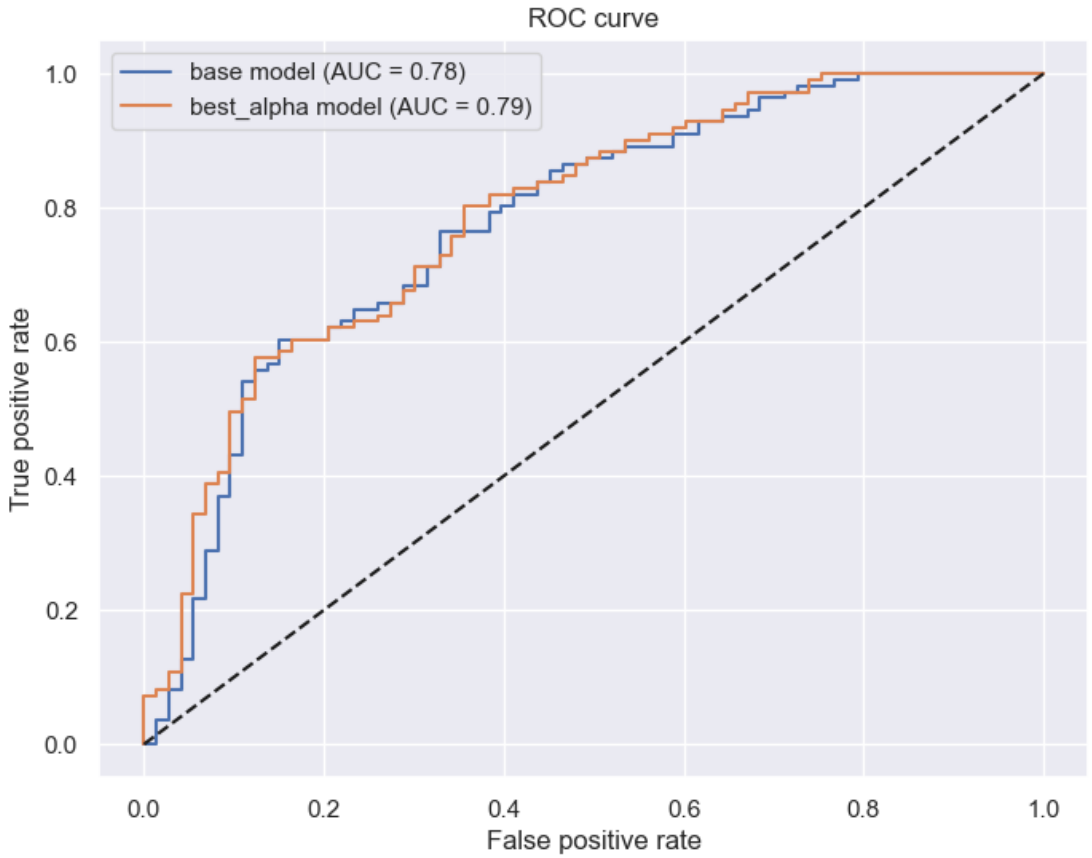
כעת ניסינו "לשחק" עם פרמטר ה-`var_smoothing` (המכונה `alpha` במחברת) אשר הינו היפר-פרמטר שנוסף לשם הוספת יציבות לחישוב. יש לשים לב שהציר האופקי הינו לוגריתמי –



בחרנו להמשיך עם `alpha=10^-3` , כי דיוק של המודל עבור הפרמטר הזה הינו 72.4%.

למעשה למודל זה ישנם רק שני פרמטרים, אחד מהם הוא ה-`var_smoothing` והשני מגדיר הסתברויות התחלה למחלקות – אם לא מכניסים הוא מגדיר את זה על פי הנתונים וכך העדפנו. כך למעשה סיימנו לחפש את הפרמטרים עבור מודל סיווג זה.

יצרנו דיאגרמות *ROC* עבור שני המסווגים הללו –



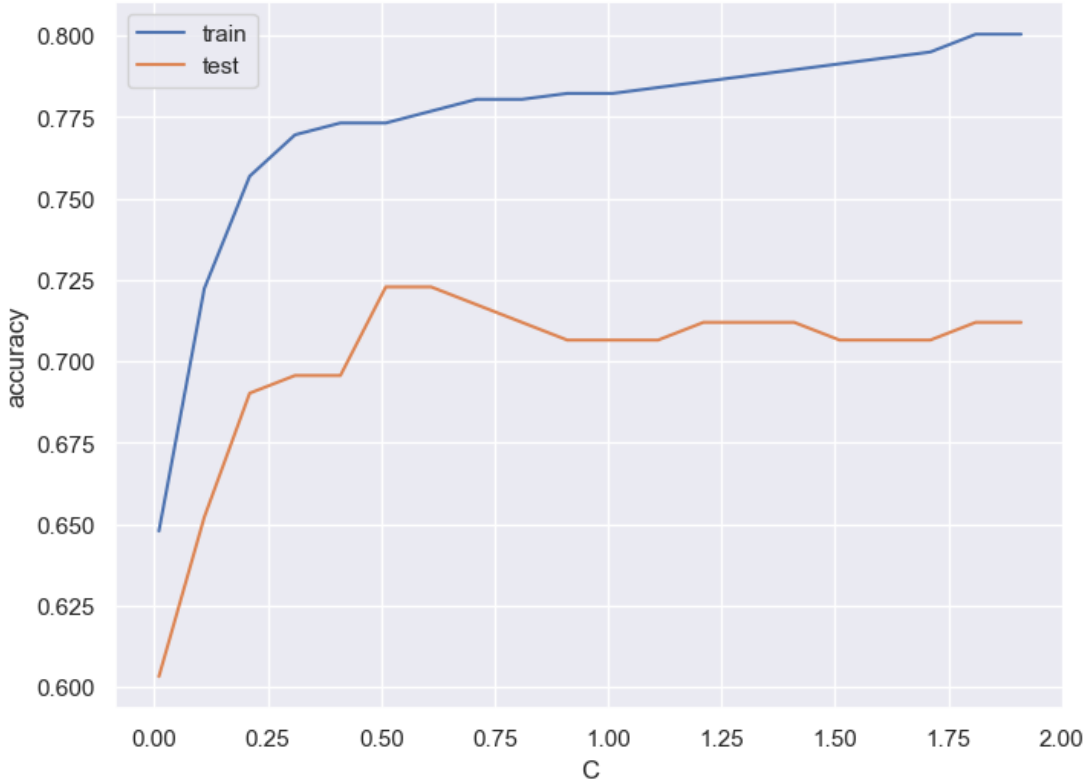
ביצענו מבחן *t* בין שני המסווגים ולא קיבלנו שיפור סטטיסטי מובהק, אך נמשיך עם המודל לאחר חיפוש הפרמטר משום שעליו קיבלנו דיוק יותר טוב ב-*test*, וגם ביוון שה-*AUC* שלו גדול יותר.

מודל רביעי – SVM

מודל זה משתמש במיפוי לא לינארי למימד גבוה יותר כדי להעביר מישור-על (קו) שיוצר הפרדה בין המחלקות במימד הגבוה יותר. ישנם גרעינים שונים שעוזרים לנו להטיל למימד הגבוה יותר, עליהם נעבור לדוגמה בחיפוש היפר-הפרמטרים.

הרצנו כמו בשלבים הקודמים את המודל הבסיסי וקיבלנו דיוק של 74.2%.

לאחר מכן "שיחקנו" עם הפרמטר C אשר הינו פרמטר רגוליזציה אשר "מעניש" את המודל על פי מטריקת L_2 –



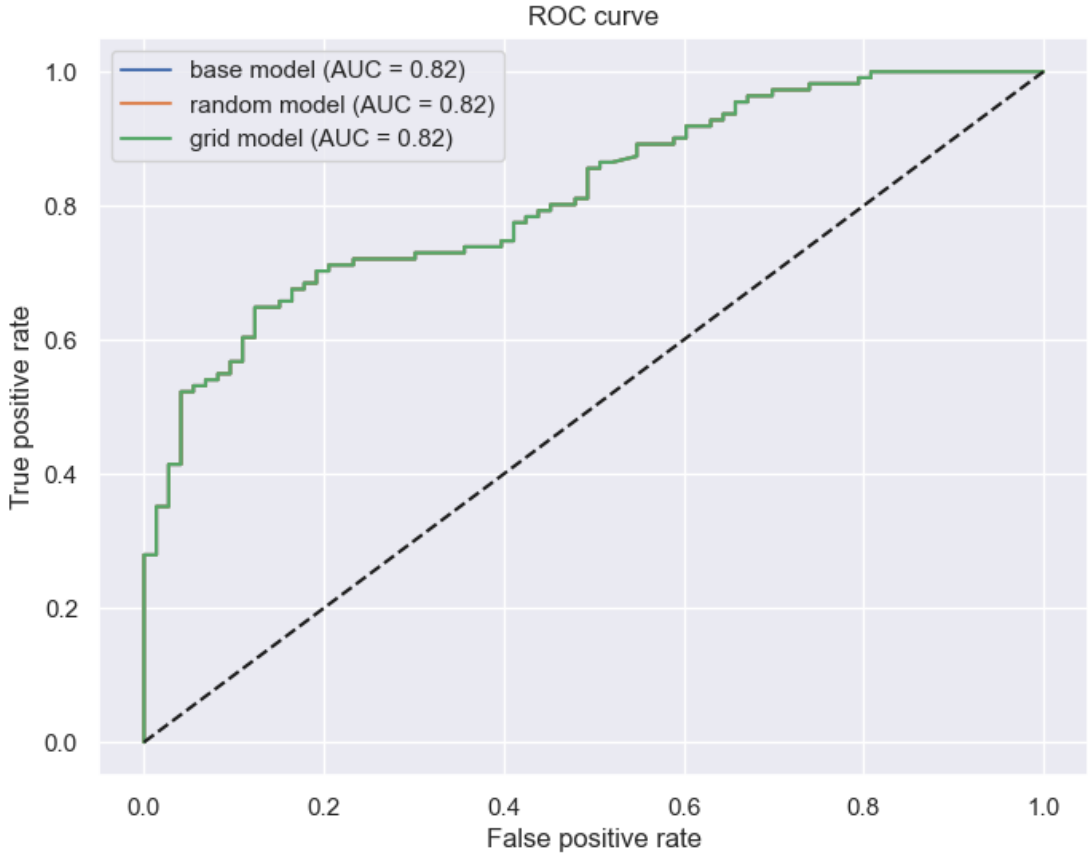
בחרנו להמשיך את חיפוש היפר-הפרמטרים סביב C הנבחר (על פי הגרף הנ"ל שמראה לנו את ההשפעה של C על הדיוק), ועל פרמטרים נוספים כגון סוג הגרעין. עבור פרמטר סוג הגרעין רצנו על: לינארי, פולינומי, סיגמואיד ו-מעריכי.

הרצת ה-random-search והרצת ה-grid-search נתנו מסווגים עם דיוק של –

```
[base] model accuracy = 70.65%
[random] model accuracy = 70.65%
Improvement of 0.00%
```

```
[base] model accuracy = 70.65%
[grid] model accuracy = 70.65%
Improvement of 0.00%
```

ראינו כי חיפוש הפרמטרים לא תרם לשיפור הדיוק, לכן נסתכל על ה-ROC curve של כל אחד מהמסווגים. להלן דיאגרמת ROC של שלושת המודלים –



כמו שניתן לראות ה-AUC של כל אחד מהמודלים זהה, לכן נבדוק את השוני בניהם על ידי ttest –

```
--------- metric for ttest: accuracy ---------
base and random have the same metric => No one is better than the other
base and grid have the same metric => No one is better than the other
random and grid have the same metric => No one is better than the other
```

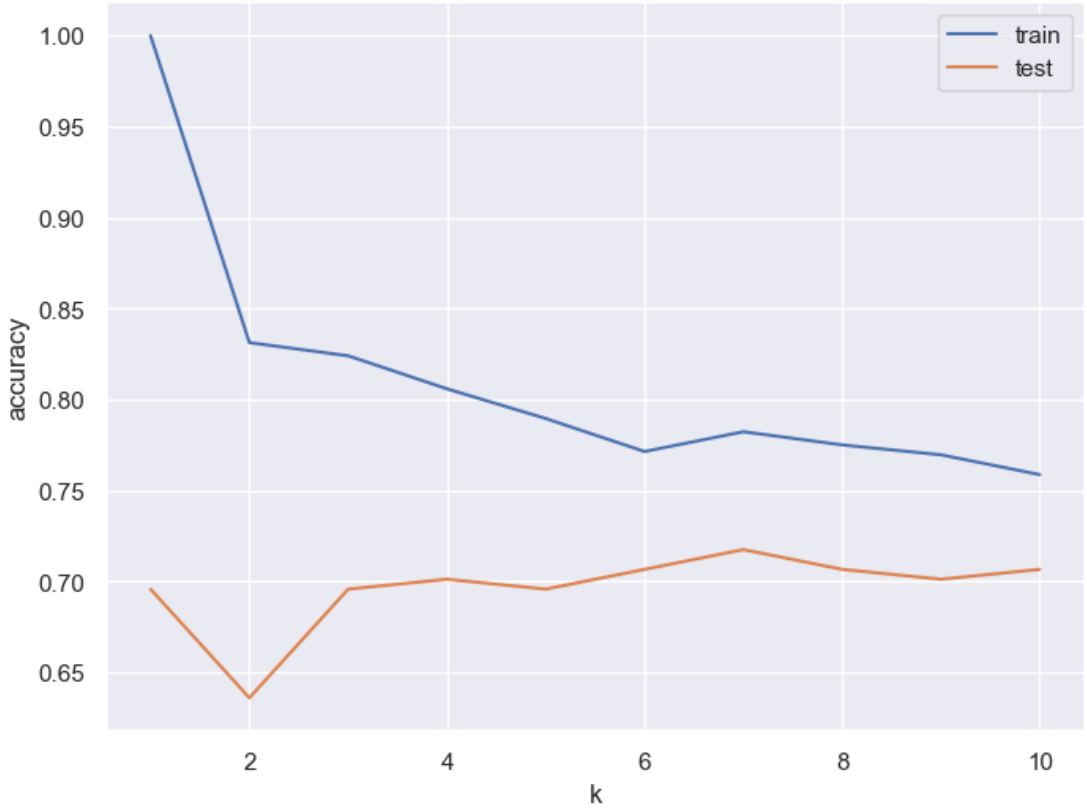
גם כאן אנחנו רואים שאין מובהקות סטטיסטית. לכן, מבין שלושת המסווגים האלו בחרנו את ה-grid באופן רנדומי.

מודל חמישי - K-NN

בהינתן נקודת בוחן, מודל זה מסווג אותה למחלקה על פי k השכנים הקרובים ביותר שלה (על פי מטריקות שונות). הוא מסתכל על k שכנים אלו, ומסווג על פי מחלקת הרוב שלהם.

כהרגלנו, הרצנו על המודל הבסיסי וקיבלנו את הביצועים הבאים – 70.6% דיוק.

כמו המסווגים הקודמים ננסה לשחק עם אחד הפרמטרים ולראות את ההשפעה שלו על הדיוק, בחרנו להסתכל על ההשפעה של מספר השכנים (אנחנו יכולים לנחש שעבור שכן אחד נקבל overfitting) –



אכן קיבלנו מה שציפינו לקבל – הגיוני כי עבור כל נקודה ב-train המודל מוצא את אותה הנקודה בתור השכן היחיד, לכן מצליח תמיד לסווג נכון. נשים לב גם לשיפור בדיוק.

כעת ננסה תשניות של המודל עם היפר-פרמטרים שונים. ראשית הסתכלנו על האלגוריתמים שבעזרתם תהליך מציאת השכנים מתבצע:

```
[ball_tree, kd_tree, brute]
```

תהליך מציאת השכנים עשוי להשפיע על דיוק פעולת המסווג.

פרמטר נוסף הינו מטריקת המרחק – מכיוון שמדברים על שכנים, צריך להגדיר את מטריקת חישוב המרחק בין שתי דוגמאות. השתמשנו במטריקות הבאות:

```
[euclidean, cosine ,haversine, infinity, Manhattan]
```

ובנוסף לעוד היפר פרמטרים, ניסינו גם לשחק עם מספר השכנים (k), כמו קודם.

הרצנו grid search וגם random search עם טווחי הפרמטרים, וקיבלנו את הביצועים הבאים:

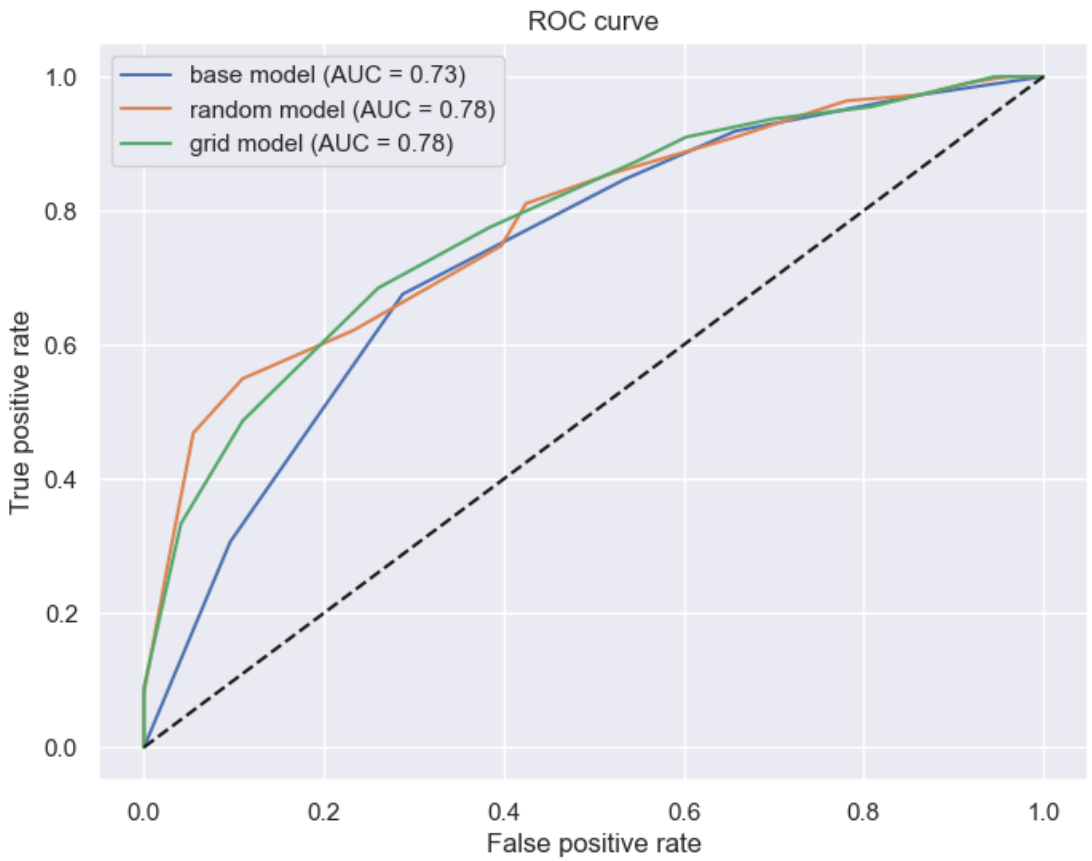
```
[base] model accuracy = 69.57%
[random] model accuracy = 70.65%
Improvement of 1.56%
```

```
[base] model accuracy = 69.57%
[grid] model accuracy = 70.65%
Improvement of 1.56%
```

פה ה-grid לתוצאה שהורידה את הדיוק, לכן עשינו חיפוש עם פרמטרי שיותר קרובה לתוצאות של ה-random. לבסוף קיבלנו את אותו הדיוק, וששניהם שיפרו את המודל הבסיסי.

** נשים לב שה-fit לא תמיד עובד על כל שילוב של ההיפר-פרמטרים, לכן קיבלנו אזהרה על כך שחלק מהחיפושים לא חוקיים – נדלג עליהם.*

הצגנו את דיאגרמת ה-ROC של שלושת המודלים:



גם כאן אנחנו יכולים לראות שאומנם קיבלנו שיפור על המודל הבסיסי אבל אין הבדל בניהם.

לכן הסתכלנו (כמו גם עבור המסווגים האחרים) על ttest כדי לראות האם ישנה מובהקות סטטיסטית בין המודלים –

```
--------- metric for ttest: accuracy ---------
p-value: 0.012330582492616189
reject null hypothesis => random is BETTER THAN base
p-value: 0.01725640248541328
reject null hypothesis => grid is BETTER THAN base
p-value: 0.7838844433882808
accept null hypothesis => No one of random and grid is better than the other
```

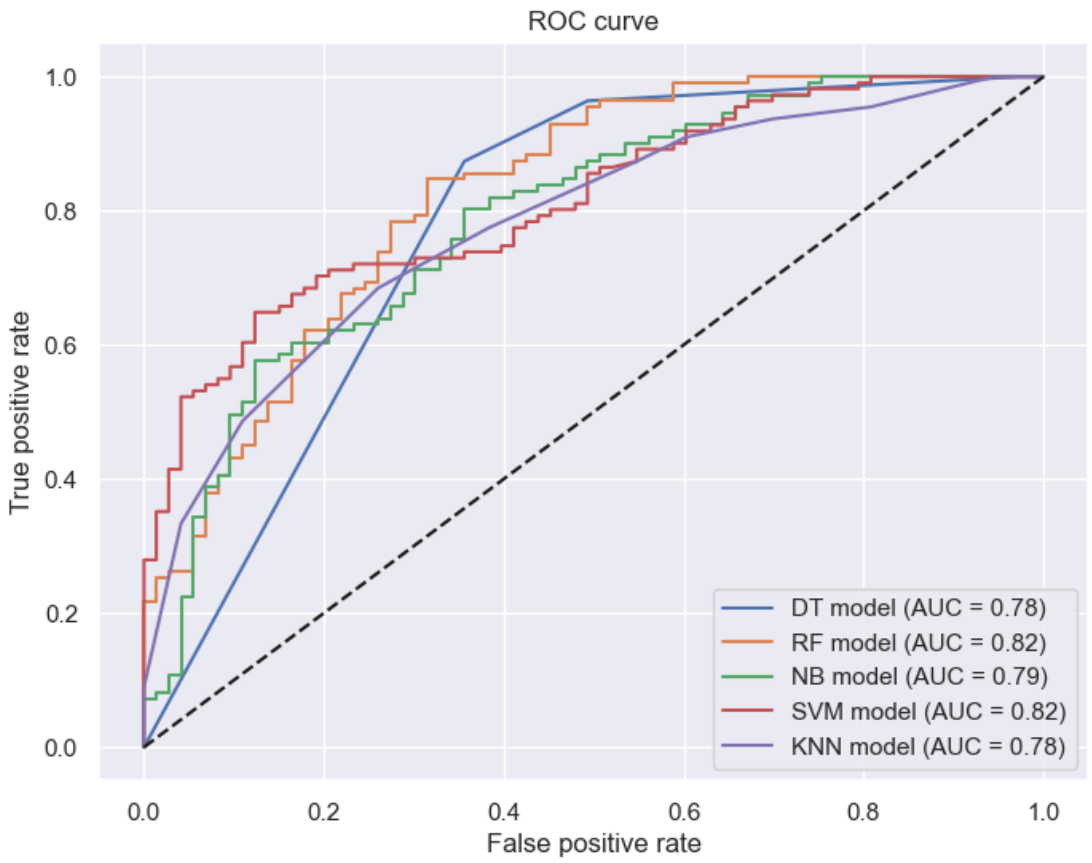
כמו שניתן לראות אין מובהקות סטטיסטית בין המודלים ולכן באופן רנדומי בחרנו להמשיך עם המודל שמתקבל מה-grid.

3) בחירת המסווג

לאחר שבחרנו מודל אופטימלי עבור כל מסווג רצינו לראות את ההבדלים בין המסווגים מבחינת כל אחת מהמטריקות השונות שעבדנו עימן.

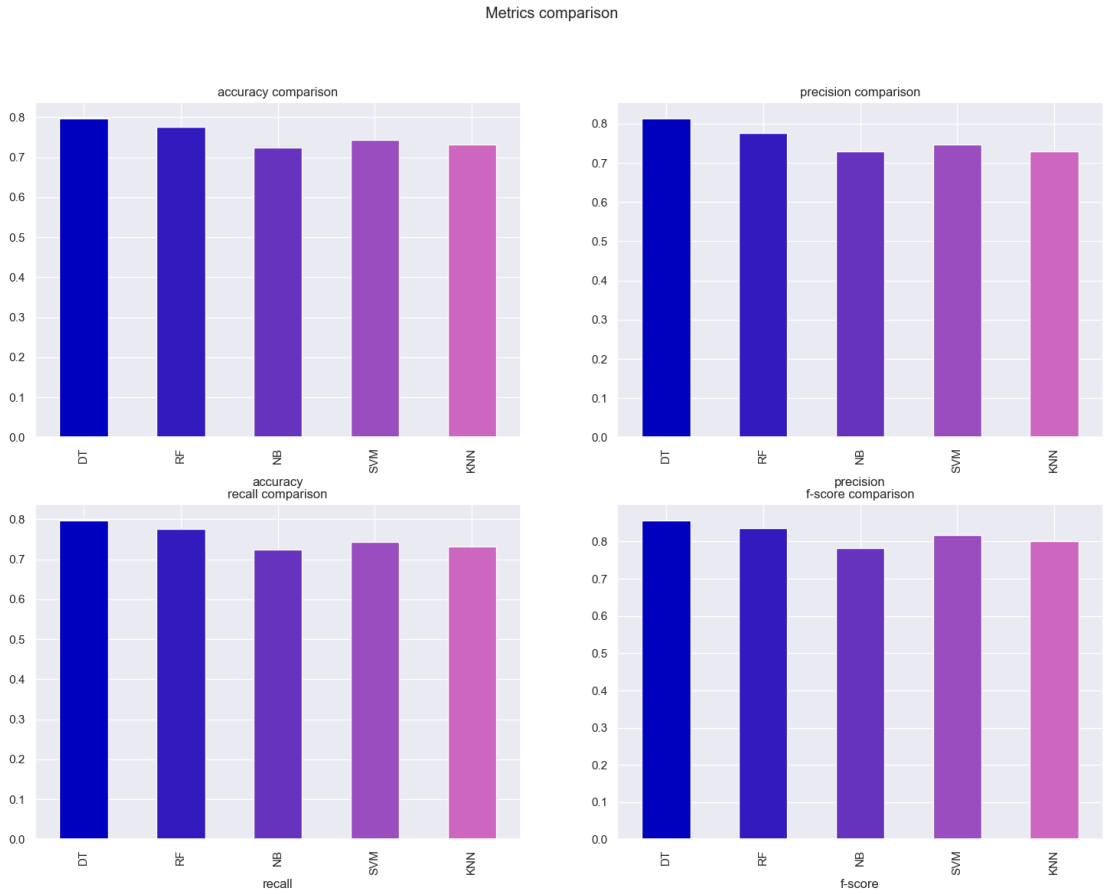
על מנת לעשות זאת הצגנו בגרפים את התוצאות עבור כל מטריקה –

התחלנו עם ה-ROC –



כפי שניתן לראות המסווגים RF וSVM קיבלו את התוצאות הטובות ביותר עבור ה-ROC.

לאחר מכן בדקנו עבור שאר המטריקות –



כפי שניתן לראות אומנם DecisionTree לא קיבל את ה-AUC הכי טוב, אבל הוא כן קיבל את התוצאות הכי טובות בשאר המטריקות.

כיוון שבדרישות התרגיל התבקשנו לבחור מודל עם דיוק מקסימלי, נבחר להמשיך איתו, אבל ביחס למטריקות שחשובות לנו (הדיוק וה-AUC), היינו מעדיפים לבחור את ה-RandomForest, כי הוא גם עם דיוק טוב (1% פחות מה-DT) אבל AUC הרבה יותר טוב.