הפקולטה להנדסה
ע"ש אלכסנדר קופקין
אוניברסיטת בר־אילן

כריית מידע - פרויקט חלק 2

שלב 1 - Split the data

ראשית, הוספנו בחלק א' כמה דברים קטנים:

- עשינו המרה של ערכים נומינלים לערכים של 0 ו1.
- יצרנו dateframe חדש שיכיל את כל הנתונים.
- עשינו export לקובץ אקסל על מנת לעבוד על הנתונים האלו ולא על הנתונים שקיבלנו אותם בהתחלה.

```
dist_col=['loan','mortgage','credit','positive']
for col in dist_col:
    df[col]=df[col].replace({True: 1,False: 0})
```

```
num_col_export = df.describe().columns # to get the numeric column
```

```
num_data_export = df[num_col_export] # numeric data
```

```
df['subscribed'] = df['subscribed'].replace({True: 1,False: 0})
```

```
num_data_export['subscribed'] = df['subscribed']
```

```
<ipython-input-95-f62a4cf2c92f>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/i
rsus-a-copy
  num_data_export['subscribed'] = df['subscribed']
```

```
num_data_export.to_excel("marketing_campaigns_train_after_pre_proc.xlsx", sheet_name='sheet1')
```

כעת, בתחילת חלק ב' ביצענו קריאה לקובץ הנתונים החדש שיצרנו ובנוסף ביצענו אתחולים שיעזרו לנו בהמשך כמו מחלקה להדפסה עם צבעים ופתיחת תיקיה לשמירת הנתונים.

```
# Class for printing with underline, colors and bold
class bcolors:
    HEADER = '\033[95m'
    OKCYAN = '\033[96m'
    ENDC = '\033[0m'
    BOLD = '\033[1m'
    UNDERLINE = '\033[4m'
```

```
# create folders
directory = "Decision Tree"
parent_dir = "./"
path = os.path.join(parent_dir, directory)
if not os.path.exists(directory):
    os.mkdir(path)

directory = "Best Params"
parent_dir = "./"
path = os.path.join(parent_dir, directory)
if not os.path.exists(directory):
    os.mkdir(path)
```

```
# import csv file
df = pd.read_csv("./marketing_campaigns_train_after_pre_proc.csv", index_col=0)
```

לאחר מכן הורדנו את הערכים שהיה חשוב לנו לשמור אך הם לא חלק מהפיצ'רים המתוקנים. בהתחלה היו לנו 20 פיצ'רים (לא ספרנו את subscribed) ולאחר הצמצום של הpre process נשארנו עם 17 פיצ'רים.

בנוסף, פיצלנו את כל הdata שלנו ל2 חלקים – train & test (75% train, 25% test).

```
df_classifier = df.copy()
l = ['age', 'account_balance', 'n_contact', 'p_days','l_call_duration','n_p_contact','p_days','status_cat','education_cat',
    'profession_cat', 'device_cat', 'month_l_date_cat', 'p_outcome_cat','age_bin','account_balance_bin',
    'education_Pre_Proc_cat','profession_Pre_Proc_cat','device_Pre_Proc_cat', 'p_outcome_Pre_Proc_cat']

df_classifier.drop(l, axis='columns', inplace=True)
```

```
# Split the data to features and target
X = df_classifier.drop('subscribed', axis=1)
y = df_classifier['subscribed']
```

```
# Split the data to train and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)
```

שלב 2 – Train classifiers & hyper parameters

ראשית בחרנו לבחון 8 מסווגים –

- Decision Tree
- Random Forest
- SVM
- K-Nearest Neighbors
- Naive Bayes classifier –
  - Gaussian
  - Multinomial
  - Complement
  - Bernoulli

מבין המסווגים הללו, חיפשנו את המסווג שישיג את ה accuracy הכי גבוה על ה train (לפי הנחיית התרגיל).
בנוסף, לדעתינו חשוב מאוד שה – sensitivity (recall) יהיה גבוה, משום שאנחנו מעוניינים למצוא כמות גבוהה של True Positive כי אנחנו רוצים למצוא את הלקוחות שבסוף נרשמו לביטוח.
אך הבנו שזה לא מספיק להשיג רק sensitivity גבוה כי המסווג יכל להגיד על כולם שהם ירשמו לביטוח ואז נקבל sensitivity = 100%, אך המסווג לא טוב כי נקבל False Positive גבוה.
אם נסתכל על מדד ה – precision:

$$precision = \frac{TP}{TP + FP}$$

ככל שנשאף להגדיל את ה precision אז אנו נקבל TP גבוה ובנוסף FP נמוך שזה אומר שלמסווג לא היה הרבה טעויות סיווג כאשר הוא סיווג על לקוח שהוא יירשם לביטוח (positive).
אנחנו חושבים שזהו המדד החישוב ביותר למסווג, משום שבסופו של דבר השיחות שיבוצעו הן שיחות אל הלקוחות אשר המסווג אמר עליהם שהם ירשמו לביטוח, לכן ככל

שה – False Positive יהיה נמוך וה – True Positive יהיה גבוה, רוב השיחות שיבוצעו ללקוחות יהיו שיחות עם פוטנציאל גבוה להרשמה לביטוח.

- הערה – בתרגיל הזה המדד העיקרי שנתנו לו יתחשבות למציאת המסווג היה ה – accuracy לאור הנחיית התרגיל, בהמשך הפרוייקט (חלק ג') נתחשב יותר במדד ה Precision במציאת המסווג הפוטנציאלי.

במסווגים – Decision Tree, Random Forest, SVM, KNN עבדנו לפי הסדר הבא:

1. קריאת קובץ ה hyper parameters הכי טובים שמצאנו עד עכשיו בשם Best_classifier.
2. ביצענו Randomized Search למציאת hyper parameters הכי טובים.
3. ביצענו Grid Search באזור של ה hyper parameters הכי טובים שנמצאו בשלב

- עבור ה – DT הפרמטרים שמדדנו היו:
  Criterion – האם עובדים לפי enthropy/gini.
  Max Depth – עומק העץ. ככל שהעץ עמוק יותר זה יכל לגרום ל overfitting.
  Min samples split – המספר המינימלי של דוגמאות הנדרש לפיצול.
  Min samples leaf – המספר המינימלי של דוגמאות בעלה.
- עבור ה RF:
  N_estimators – מספר העצים ביער העץ.
  Max_features – מספר התכונות שלוקחים בחשבון לפיצול העץ.
  ובנוסף הפרמטרים של העץ הרגיל –
  max_deph,min_samples_split, min_samples_leaf.
- SVM:
  C: מקדם שמשפיע על ה over או under fitting.
  בנוסף בדקנו גם את gamma,kernel ו degree .
- KNN:
  הפרמטר שבדקנו הוא K – כמות השכנים הכי קרובים שמשפיעים על הסיווג.

4. השוואה בין ה accuracy שמצאנו במסווג בבדיקה הנוכחית לבין ה accuracy הכי טוב שמצאנו בכלל הבדיקות (עבור כל חלוקה שעשינו ל data בכל הפעמים שהרצנו את הקוד).

5. אם המסווג שמצאנו בבדיקה הזו בעל דיוק יותר טוב מהמסווג השמור בקובץ Best_classifier אז עידכנו את הקובץ ב hyper parameters שמצאנו. אחרת, המשכנו עם ה hyper parameters שהיו שמורים בקובץ.
* בעצם החלטנו שאנחנו בוחרים את ההיפר פרמטרים על פי ההיפר פרמטרים שהשיגו תוצאת accuracy הכי גבוה עבור חלוקה מסויימת של ה data .

במסווגים מסוג – Naïve bayes classifier יצרנו 4 מסווגים:

```
GNB = GaussianNB()
MNB = MultinomialNB()
COPNB = ComplementNB()
BNB = BernoulliNB()
```

לאור הזמן הרב שלקח למחשב להריץ את החיפושים במסווגים הקודמים, ובנוסף ראינו שהמסווגים הללו לא מגיעים ל accuracy גובה, החלטנו שלמסווגים הללו לא נשקיע בחיפוש hyper parameters ולהמשיך לשלב הבא.

כמו כן בנוגע לכל המסווגים, כל הבדיקות שעשינו היו על ה X_train, y_train שיצרנו בחלוקה של ה data.

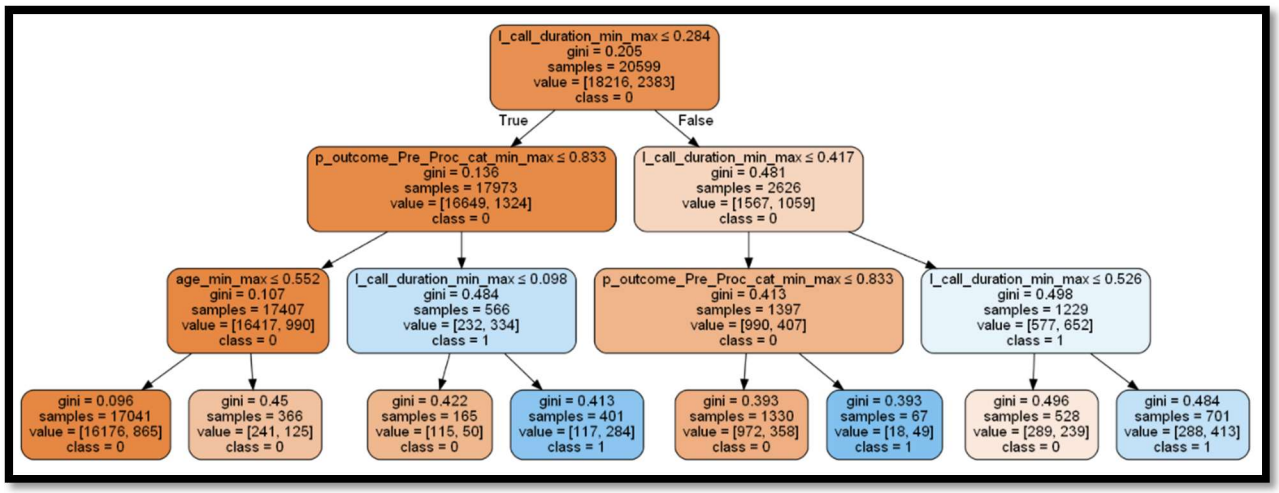
- הערה בקשר ל SVM – בגלל הקושי בזמן הריצה של חיפושי עבור ה - hyper parameters שלאחר בדיקה ראינו שהוא זמן ריצה אקספוננציאלי, ובנוסף לקח לנו הרבה מאוד זמן להריץ את ה – Random, grid search. השתמשנו במעטפת בשם – BaggingClassifier. מעטפת זו עזרה לנו להריץ את החיפושים בזמן ריצה פולינומיאלי וכך הצלחנו למצוא את ה hyper parameters של ה SVM.

הפקולטה להנדסה
ע"ש אלכסנדר קופקין
אוניברסיטת בר־אילן

Visualization

Decision Tree

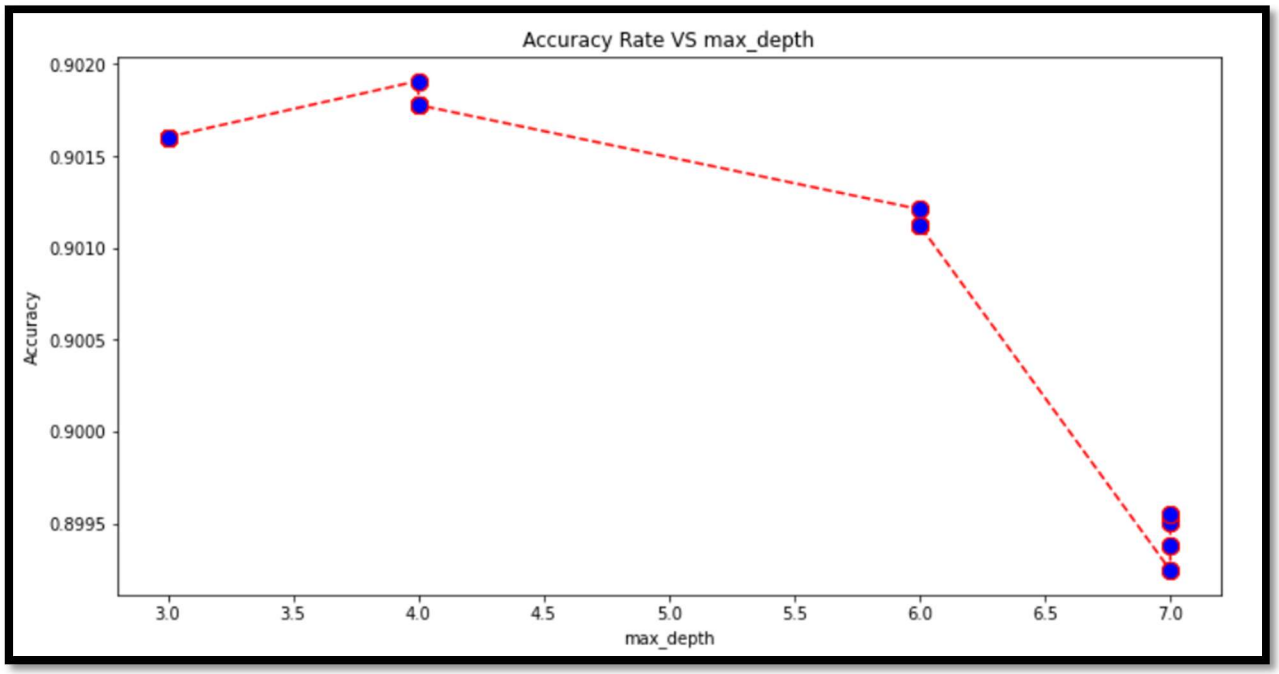
תמונה של המסווג הטוב ביותר מסוג Decision Tree :



ניתן לראות שרוב ההחלטות של המסווג הם על פי l_call_duration וככל הנראה זה בגלל שראינו בחלק 1 שלפיצ'ר זה היה קורלציה גבוהה עם subscribed (0.37 וזה היה גבוה ביחס לפיצ'רים אחרים).
ניתן לראות שרוב הסיווג הוא subscribed = 0 (צבוע בצבע כתום).

בנוסף ערכנו בדיקה של accuracy מול עומק העץ (ביצענו בדיקה אל מול כל תוצאות העצים שקיבלנו ב grid search).

גרף שמציג את תוצאות הבדיקה:

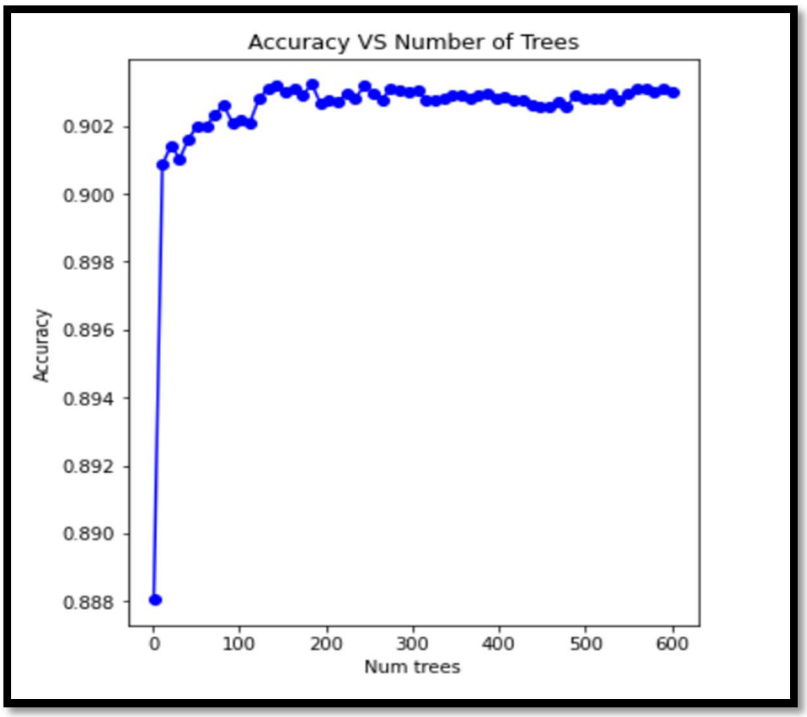


קיבלנו שעומק העץ שהשיג accuracy הכי גבוה אל מול החלוקה הזו של ה data הוא 4, והעץ עם הדיוק (accuracy) הגבוהה ביותר שהצלחנו למצוא מתוך כלל ההרצות של אלגוריתם חיפוש היפר פרמטרים הוא עץ בגובה 3, והדיוק שלו בגרף הוא כמעט הדיוק המקסימלי.
לכן ניתן לראות שהעץ החלטות הטוב ביותר שהצלחנו הוא מסווג טוב אל מול החלוקה הזו של ה data.

Random Forest

המסווג הטוב ביותר שקיבלנו עד כה (השיג accuracy הכי גבוה מכל הפעמים שביצענו הרצה לאלגוריתמי חיפוש) הוא עם n_estimators = 500 .
בכדי לבדוק עם מספר העצים שקיבלנו (500) הוא פרמטר טוב, ערכנו בדיקה של מספר העצים ביער כנגד ה – accuracy , מול החלוקה הנוכחית של ה X_train, y_train .

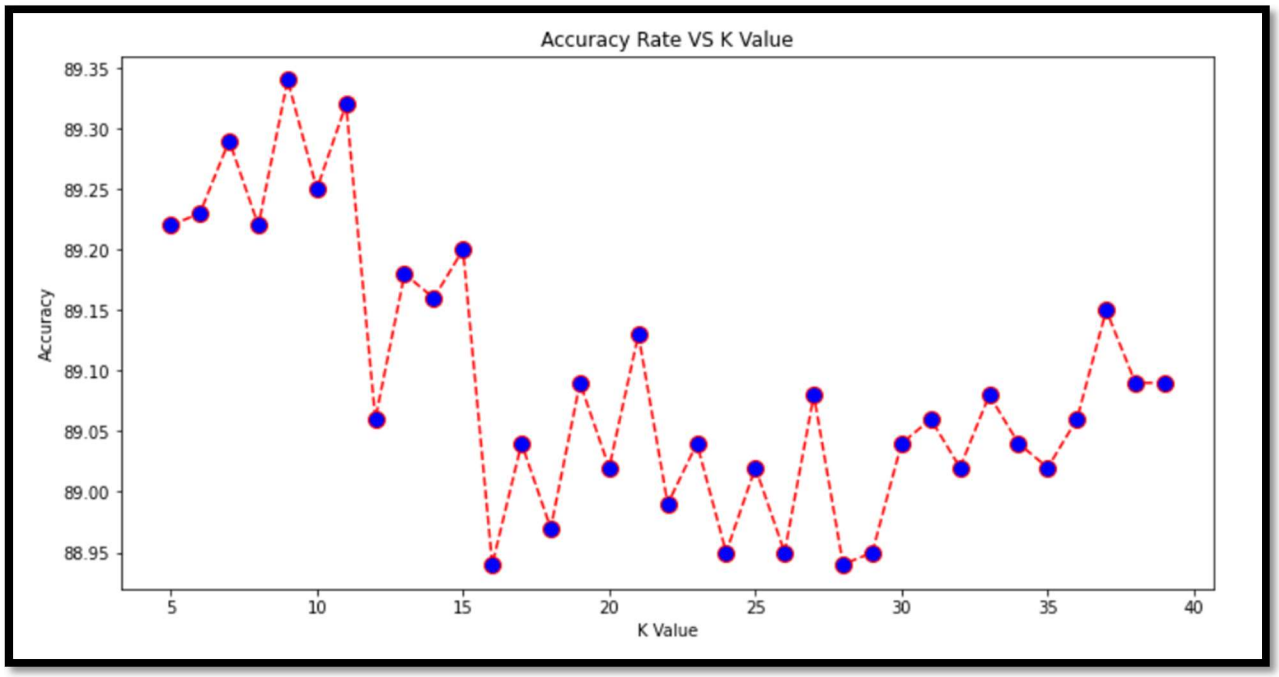
הצגנו את התוצאות בגרף הבא:



ניתן לראות מן הגרף שעבור החלוקה הנתונה n_estimators = 500 , X_train, y_train נותן אחוז דיוק גבוה. (בקירוב – דיוק מקסימלי).
לכן ניתן להסיק מן התוצאות שהמסווג שקיבלנו הוא אכן טוב אל מול החלוקה הזו של ה data .

K-Nearest Neighbors

ערכנו בדיקה של K value אל מול accuracy.
הצגנו את תוצאות הבדיקה בגרף:



קיבלנו שה – K value שהשיג accuracy גבוה עבור ה test הנוכחי הוא 9.
עבור כל הבדיקות שעשינו (מספר הרצות של הקוד וחלוקות שונות של test) ה K הטוב ביותר שמצאנו הוא – 17 כי הוא השיג עבור בדיקה מסויימת הוא הצליח להשיג accuracy של 89.57. וכאן בגרף הוא השיג אחוז של 89.05 לעומת ה K הטוב ביותר כאן שמשיג אחוז דיוק של 89.35.
החלטנו להישאר עם K=17 מכיוון שההפרש בין 89.05 ל 89.35 הוא אינו גבוה.

הפקולטה להנדסה
ע"ש אלכסנדר קופקין
אוניברסיטת בר־אילן

שלב 3 – Find the best classifier

לאחר שלב 2 מצאנו את הנציג הטוב ביותר עבור כל סוג מסווג.
בשלב זה בדקנו מיהו המסווג הכי טוב מבין כל המסווגים.

ראשית, חלקנו את X_train , y_train ל - $10\ Folds$ והרצנו כל מסווג על כל Fold.
שמרנו את התוצאות של המסווגים בתוך שני מילונים – cm_dict (מילון ששומר את ה confusion matrix של כל סיווג) וב – auc_dict (מילון ששומר את תוצאות ה accuracy של כל מסווג במהלך כל סיווג).

לאחר מכן, בדקנו את ממוצע התוצאות של כל מסווג, המסווג הנבחר הוא המסווג שהשיג ממוצע accuracy הכי גבוה.
המסווג הטוב ביותר שקיבלנו הוא Random Forest.
(כמו כן המשתנה - Best_clf התעדכן להיות rf -המסווג של random forest).

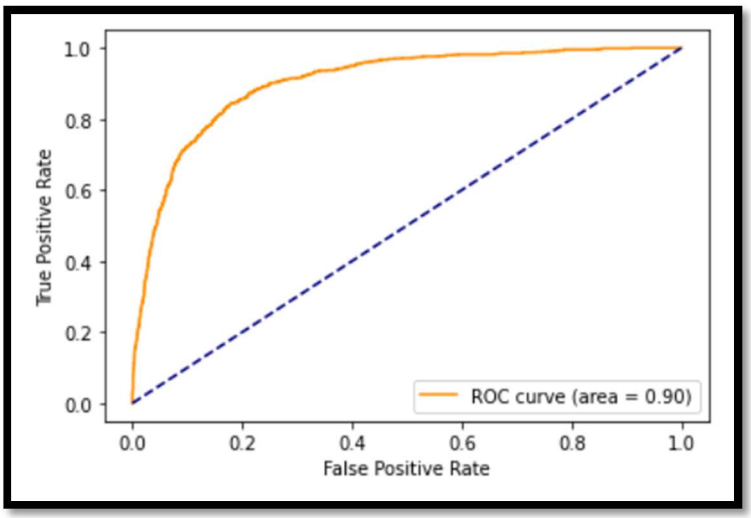
בנוסף ערכנו בדיקת ttest וקיבלנו את התוצאות הבאות:

```
Best clf VS DecisionTreeClassifier(max_depth=3, min_samples_leaf=4, random_state=42)
p-value: 0.09218053493399027
accept null hypothesis
Best clf VS SVC(C=0.1, degree=5, gamma=0.01, probability=True)
p-value: 1.0716066164746064e-06
reject null hypothesis
Best clf VS KNeighborsClassifier(n_neighbors=17)
p-value: 0.0012239598645094863
reject null hypothesis
Best clf VS GaussianNB()
p-value: 1.2521543907329877e-09
reject null hypothesis
Best clf VS MultinomialNB()
p-value: 1.4637678956321212e-06
reject null hypothesis
Best clf VS ComplementNB()
p-value: 1.7040726265557063e-13
reject null hypothesis
Best clf VS BernoulliNB()
p-value: 4.320650104793522e-06
reject null hypothesis
```

מן התוצאות ניתן לראות שמול 7 המסווגים קיבלנו 6 פעמים – reject null hypothesis, זאת אומרת שהמסווג יציב לעומת שאר המסווגים.
אל מול ה DT קיבלנו – accept null hypothesis, כנראה מכיוון שגם ה DT היה בעל accuracy גבוה והוא גם היה מסווג טוב ויציב, אך ה RF היה טוב יותר ביחס אליו.

שלב 4 – Test the best classifier

לאחר שמצאנו את המסווג שהשיג ממוצע accuracy הכי גבוה על ה train, סיווגנו באמצעותו את ה 25% ה data שהשארנו לצורך test.
בדקנו את ה accuracy שיצא לנו (יצא 90.22) וראינו שזה גבוה ותואם בקירוב לממוצע ה accuracy שהמסווג השיג עבור האימון על ה – train (90.45).
גרף ה – ROC :



ניתן לראות שה AUC גבוה – 0.90 וזה מעיד על כך שמסווג הטוב ביותר שמצאנו הוא מסווג טוב (על פי ההרצאה אם ה AUC גבוה מ 0.87 זה מעיד על מסווג טוב).
בנוסף לפני שעברנו לשלב 5 עשינו fit עבור כל ה DATA.

הפקולטה להנדסה
ע"ש אלכסנדר קופקין
אוניברסיטת בר־אילן

שלב 5 – Classify the test file

בשלב זה סיווגנו את הקובץ test שקיבלנו בתרגיל .
ראשית, הרצנו עבור הקובץ את תהליך ה pre processing שעשינו בחלק הראשון של הפרוייקט. קיבלנו קובץ -
marketing_campaigns_test_after_pre_proc.csv
(שמור בתקייה של קבצי CSV).
כעת, פתחנו 2 קבצים –
df_test: קובץ ה test המקורי.
df_test_after_PreProc: קובץ ה - TEST לאחר שעבר תהליך Pre Processing.

הרצנו את הסיווג על df_test_after_PreProc, קיבלנו פרדיקציה y_pred ושמרנו את התוצאות ב df_test.
לסיום, שמרנו את df_test בקובץ csv בשם –
marketing_campaigns_test_classifier_answers.csv
(הקובץ שמור בתקייה הראשית).