**מטלה מס 2: מגישים**

מנסור - 207641127

זוהר – 207010596

קישור ל COLAB - [HW2.ipynb - Colab](https://colab.research.google.com/drive/1NZIN7cOuQ5lf0GTM3045a4VFCIZZXS6t" \l "scrollTo=6fDj__pmVu35)

import pandas as pd # Import pandas library for table management

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split # to split the dataset for training and testing

from sklearn.tree import DecisionTreeClassifier # import Decision Tree Classifier model

# Include confusion matrix plot function to be used when we have defined a classifier

from sklearn.metrics import ConfusionMatrixDisplay, confusion\_matrix, roc\_auc\_score

import warnings # current version of seaborn generates a bunch of warnings that we'll ignore

warnings.filterwarnings("ignore")

##loading data from HW1 file

seker = pd.read\_csv("/content/sample\_data/NumericalDatasetHW1 (2).csv")

**הכנת נתונים לעץ החלטה**

*\*קובץ הנתונים כבר מסודר, נקי ומוכן לשימוש - ממטלה קודמת*

# חלוקת הנתונים לאימון ובדיקה באופן רנדומלי

# הפרמטר test\_size=0.2 מחלק את הנתונים ביחס 20-80 , אימון=80% ובדיקה=20%

train, test = train\_test\_split(seker, test\_size=0.2, random\_state=0,

                              stratify=seker['מחנות פוליטיים (יהודים בלבד)'])

# הכנת טבלאות המאפיינים והמטרה

# מכיוון ששמות העמודות אצלנו ארוכים ומבלבלים , אז עושים בשיטה זו:

# כל העמודות חוץ מהמשתנה המטרה הן המאפיינים המסבירים

feature\_columns = [col for col in train.columns if col != 'מחנות פוליטיים (יהודים בלבד)']

train\_X = train[feature\_columns]  # לקיחת המאפיינים המסבירים מנתוני האימון

train\_y = train['מחנות פוליטיים (יהודים בלבד)']  # המטרה של נתוני האימון

test\_X = test[feature\_columns]  # לקיחת המאפיינים המסבירים מנתוני הבדיקה

test\_y = test['מחנות פוליטיים (יהודים בלבד)']  # המטרה של נתוני הבדיקה

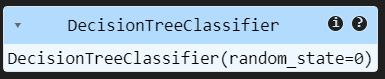
**בניית מודל עץ החלטה**

# בניית מסווג עץ החלטה עם פרמטרים ברירת מחדל

tree\_clf = DecisionTreeClassifier(random\_state=0)

# אימון המסווג באמצעות נתוני האימון

tree\_clf.fit(train\_X, train\_y)

****

**הצגת העץ המלא**

from sklearn.tree import export\_graphviz

import graphviz

import pydotplus

dot\_data = export\_graphviz(tree\_clf, out\_file=None,

                           feature\_names=train\_X.columns,

                           class\_names=[“מרכז/שמאל”, “ימין”],

                           rounded=True, filled=True, special\_characters=True)

graph1 = graphviz.Source(dot\_data)

# שמירת תמונה להצגה

pydot\_graph = pydotplus.graph\_from\_dot\_data(dot\_data)

pydot\_graph.write\_png(‘tree\_clf.png’)

# הצגת העץ

graph1

\*העץ גדול מדי\*

test\_X.head()

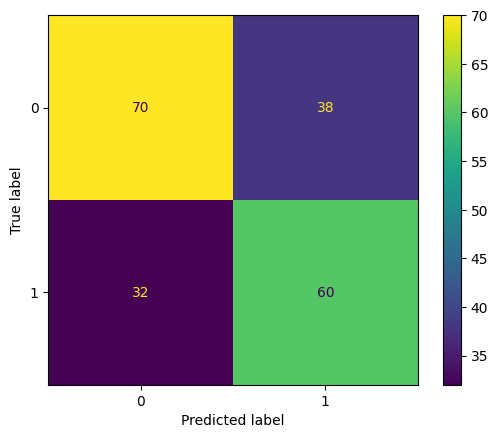
# חיזוי ערכים עבור נתוני הבדיקה באמצעות המסווג המאומן

predicted\_y = tree\_clf.predict(test\_X)

הערכת המודל - נתוני בדיקה – מטריצת בלבול

# הצגת מטריצת בלבול

ConfusionMatrixDisplay.from\_predictions(test\_y, predicted\_y)



# חישוב מטריקות

from sklearn.metrics import accuracy\_score, recall\_score, precision\_score

# מדידות עבור נתוני בדיקה

test\_accuracy = accuracy\_score(test\_y, predicted\_y)

test\_recall = recall\_score(test\_y, predicted\_y)

test\_precision = precision\_score(test\_y, predicted\_y)

print("מדידות עבור נתוני בדיקה:")

print("Accuracy: ", test\_accuracy)

print("Recall: ",test\_recall)

print("Precision: ", test\_precision)

מדידות עבור נתוני בדיקה:

Accuracy: 0.65

Recall: 0.6521739130434783

Precision: 0.6122448979591837

# ביצוע חיזויים עבור נתוני האימון באמצעות המודל המאומן

predicted\_train\_y = tree\_clf.predict(train\_X)

# הצגת מטריצת בלבול עבור נתוני האימון

ConfusionMatrixDisplay.from\_predictions(train\_y, predicted\_train\_y)

# חישוב מטריקות עבור נתוני האימון

train\_accuracy = accuracy\_score(train\_y, predicted\_train\_y)

train\_recall = recall\_score(train\_y, predicted\_train\_y)

train\_precision = precision\_score(train\_y, predicted\_train\_y)

print("מדידות עבור נתוני אימון:")

print("Accuracy: ", train\_accuracy)

print("Recall: ",train\_recall)

print("Precision: ", train\_precision)

מדידות עבור נתוני אימון:

Accuracy: 0.95625

Recall: 0.9182561307901907

Precision: 0.9853801169590644

A chart of a number and a number

AI-generated content may be incorrect.

* מטריקות נתוני האימון גבוהות יותר ממטריקות נתוני הבדיקה
* Accuracy TRAIN (0.956) > Accuracy TEST (0.650)
* יש הבדל של כ-30% בין הביצועים
* אם יש Overfitting, כנראה יש לנו אפשרויות טובות יותר לשפר את ביצועי המודל!

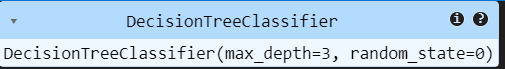
# טיפול ב-Overfitting

# בניית מסווג חדש עם הגבלת עומק העץ ל-3

max\_depth\_tree\_clf = DecisionTreeClassifier(random\_state=0, max\_depth=3)

# אימון המסווג החדש עם נתוני האימון

max\_depth\_tree\_clf.fit(train\_X, train\_y)



# ביצוע חיזויים באמצעות המודל החדש על נתוני הבדיקה

predicted\_y\_improved = max\_depth\_tree\_clf.predict(test\_X)

# הצגת מטריצת בלבול

ConfusionMatrixDisplay.from\_predictions(test\_y, predicted\_y\_improved)

# מדידת ביצועים באמצעות נתוני בדיקה

test\_accuracy\_improved = accuracy\_score(test\_y, predicted\_y\_improved)

test\_recall\_improved = recall\_score(test\_y, predicted\_y\_improved)

test\_precision\_improved = precision\_score(test\_y, predicted\_y\_improved)

print("Accuracy: ", test\_accuracy\_improved)

print("Recall: ", test\_recall\_improved)

print("Precision: ", test\_precision\_improved)

מדידות עבור נתוני בדיקה (מודל מוגבל):

Accuracy: 0.76

Recall: 0.7065217391304348

Precision: 0.7558139534883721

A chart of different colored squares

AI-generated content may be incorrect.

# ביצוע חיזויים באמצעות המודל החדש על נתוני האימון

predicted\_train\_y\_improved = max\_depth\_tree\_clf.predict(train\_X)

# הצגת מטריצת בלבול

ConfusionMatrixDisplay.from\_predictions(train\_y, predicted\_train\_y\_improved)

train\_accuracy\_improved = accuracy\_score(train\_y, predicted\_train\_y\_improved)

train\_recall\_improved = recall\_score(train\_y, predicted\_train\_y\_improved)

train\_precision\_improved = precision\_score(train\_y, predicted\_train\_y\_improved)

print("Accuracy: " ,train\_accuracy\_improved)

print("Recall: " ,train\_recall\_improved)

print("Precision: " ,train\_precision\_improved)

Accuracy: 0.76875

Recall: 0.6975476839237057

Precision: 0.7757575757575758

A chart of a number of colored squares

AI-generated content may be incorrect.

* **השוואת מטריקות המודל עם עומק מוגבל בין בדיקה לאימון**

Accuracy אימון דומה ל-Accuracy בדיקה

* **השוואת מטריקות בדיקה: עץ לא מוגבל לעומת עומק מוגבל**

נתוני בדיקה (עץ לא מוגבל, overfitted): Accuracy: 65.0%

נתוני בדיקה (לא overfitted, עומק מוגבל): Accuracy: 76.8%

* **תוצאה:**

Overfitting הופחת במודל עם עומק מוגבל (מטריקות אימון ובדיקה קרובות)

ביצועים השתפרו/נשארו דומים באמצעות מודל עם עומק מוגבל

הפחתת Overfitting משפרת יכולת הכללה על נתוני בדיקה

# ויזואליזציה של עץ החלטה עם עומק מוגבל

dot\_data = export\_graphviz(max\_depth\_tree\_clf, out\_file=None,

                           feature\_names=train\_X.columns,

                           class\_names=["מרכז/שמאל", "ימין"],

                           rounded=True, filled=True, special\_characters=True)

graph1 = graphviz.Source(dot\_data)

# שמירת תמונה להצגה

pydot\_graph = pydotplus.graph\_from\_dot\_data(dot\_data)

pydot\_graph.write\_png('tree\_clf.png')

# הצגת העץ

graph1

A diagram of a number of numbers

AI-generated content may be incorrect.

**בדיקת חשיבות מאפיינים**

import matplotlib.pyplot as plt

%matplotlib inline

import seaborn as sns

feature\_score\_DT = max\_depth\_tree\_clf.feature\_importances\_

df\_feature\_DT = pd.DataFrame(data=list(train\_X), columns=["feature"])

df\_feature\_DT["score"]= feature\_score\_DT

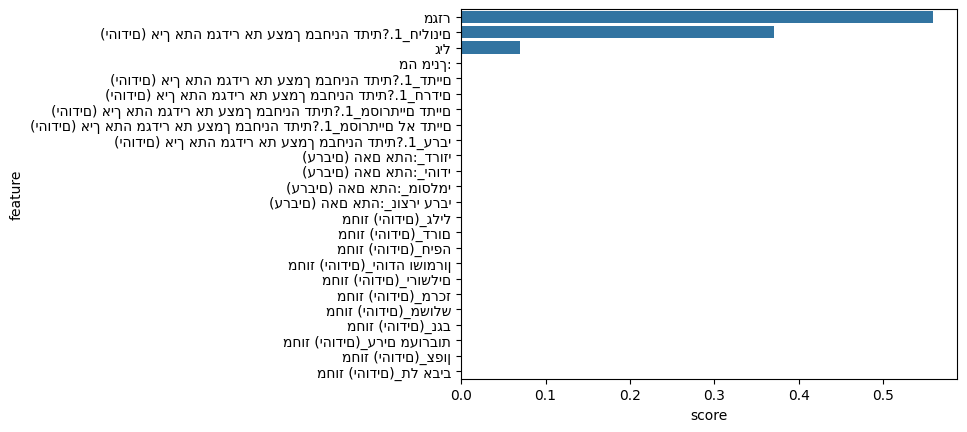
df\_feature\_DT = df\_feature\_DT.sort\_values(by=['score'], ascending=False)

sns.barplot(y="feature", x="score", data=df\_feature\_DT)

print('Decision Tree Feature Importance:')

plt.show()

Decision Tree Feature Importance:

****

**מסקנה**

המאפיינים החשובים ביותר למודל לחיזוי עמדה פוליטית הם:

**1. מגזר**

**2. איך אתה מגדיר את עצמך מבחינה דתית?\_חילונים**

**3. גיל**