תרגיל בית 2:

מגישים: איתי גיא (ת.ז. – 305104184), אורי בן-יצחק (ת.ז. – 066374737)

שאלה 1:

.*a*

```python
'''
. Computing regrssion model optimal weights.
. Return a [weights vector]
'''
def compute_reg_weigths(Y, b):
  YtY =  np.matmul(Y.T, Y)
  a = np.matmul(np.matmul(np.linalg.inv(YtY), Y.T), b)
  return a
```

```python
'''
. Computing predictions using MSE model.
. Return numpy array predictions
'''
def compute_reg_predict(Y, a):
  return np.matmul(Y, a)
```

```python
'''
. Computing regression errors.
. Return MSE, MAE
'''
def compute_reg_errors(ytrain_true, ytrain_pred, ytest_true, ytest_pred):
  mse_train = np.square(ytrain_true - ytrain_pred).mean(axis=0)
  mse_test = np.square(ytest_true - ytest_pred).mean(axis=0)
  return mse_train, mse_test
```

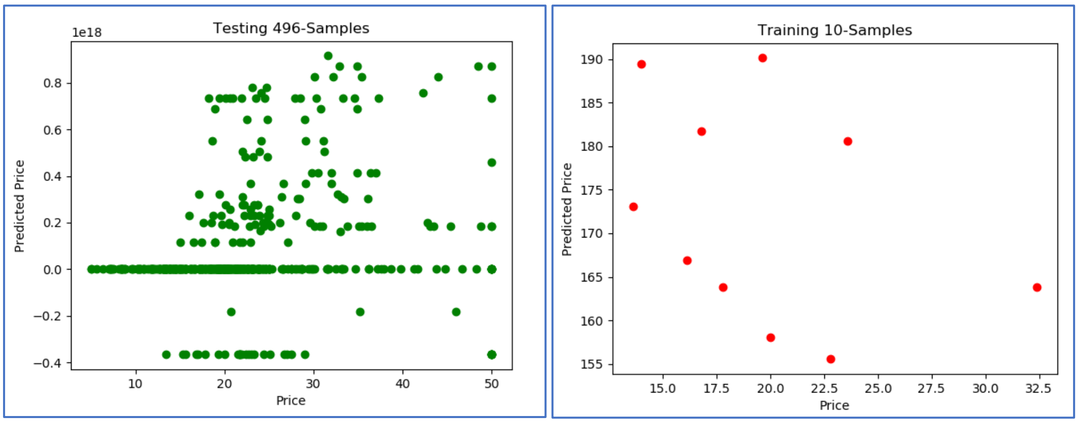
.*b*

```
=== Train-Size/Test-Size: 10/496 >>
 >> Train-MSE/Test-MSE: 23512.91170716526/6.325006803798419e+34
=== Train-Size/Test-Size: 50/456 >>
 >> Train-MSE/Test-MSE: 7.235137617297591/38.99288589668179
=== Train-Size/Test-Size: 100/406 >>
 >> Train-MSE/Test-MSE: 17.488980350095304/29.99543996872937
=== Train-Size/Test-Size: 200/306 >>
 >> Train-MSE/Test-MSE: 19.70210159269105/24.311335576692674
=== Train-Size/Test-Size: 300/206 >>
 >> Train-MSE/Test-MSE: 21.846956427517927/22.78653364618485
=== Train-Size/Test-Size: 400/106 >>
 >> Train-MSE/Test-MSE: 21.98323302632929/22.17579957531372
```

.*c* כפי שניתן להבחין בהדפסה הנ"ל, שגיאת האימון יורדת ושגיאת הבדיקה עולה כתוצאה מהרצאת *Regression* על הגדלים 10, 50, 100, 200, 300, 400.

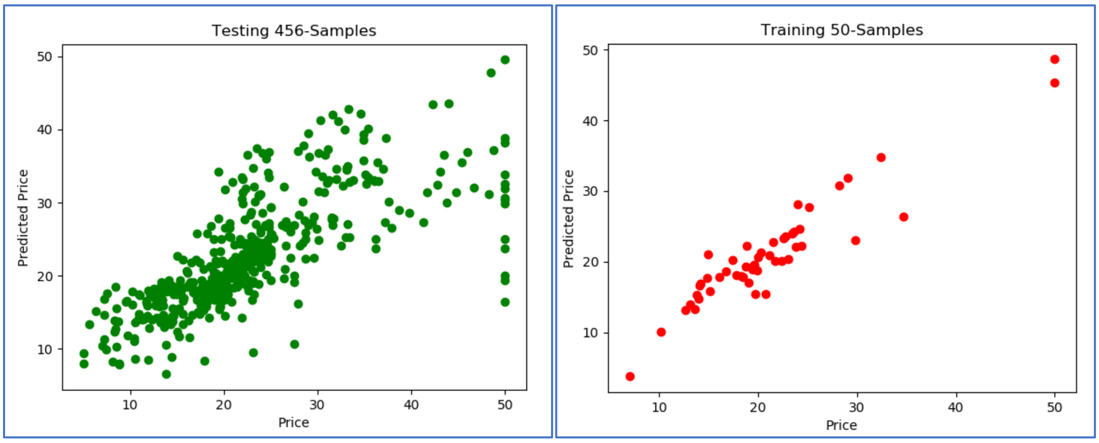
כאשר ישנם 10 דוגמאות אימון השגיאות מאוד גדולות מכיוון שדוגמאות האימון מעטות מדי ומפוזרות לכן לא ניתן למצוא מודל לינארי טוב שיכליל את דוגמאות הבדיקה (זה דומה לרעש). בנוסף לכך, ישנם הרבה דוגמאות בדיקה ביחס לדוגמאות אימון ולכן שגיאת האימון נמצאת קטנה יותר משגיאת הבדיקה.

ניתן לראות את דוגמאות האימון לעומת דוגמאות השגיאה:

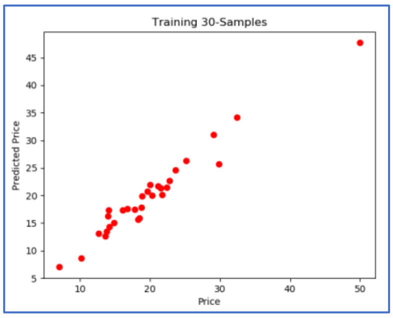


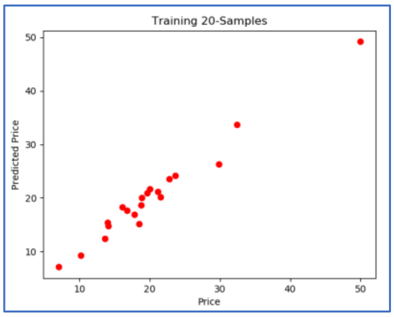
החל מ-50 דוגמאות אימון ניתן לראות שיש לדוגמאות מבנה לדוגמאות האימון והבדיקה וצופים שנמצא מודל שיוכל להקטין את השגיאות (המרחקים בין הנקודות הופכים קטנים יותר).

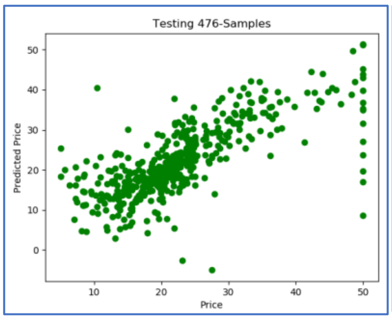
ניתן לראות את דוגמאות האימון לעומת דוגמאות הבדיקה:

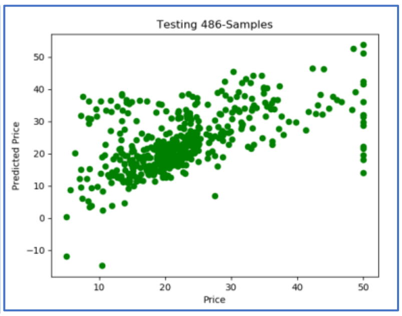


כך, ככל שהכמות של דוגמאות האימון גדלה ההכללה הופכת טובה יותר והשגיאה תרד. בנוסף, השגיאה יורד גם כי הכמות של דוגמאות הבדיקה יורדת ולכן כמות האיברים שמחשבים את שגיאת הבדיקה קטנה.









כפי שניתן לראות הקו הלינארי יבצע סיבוב בכדי למזער את השגיאה. הסיבוב הגדול ביותר יקרה מעל 10 דוגמאות אימון.

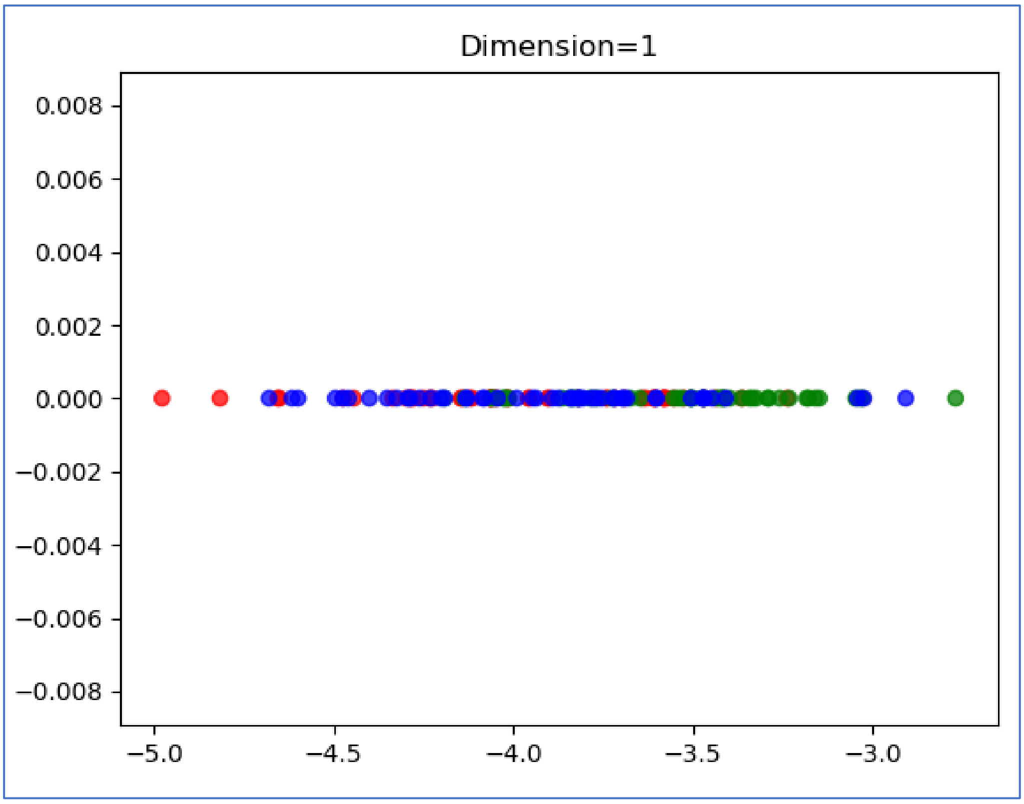
שאלה 2:

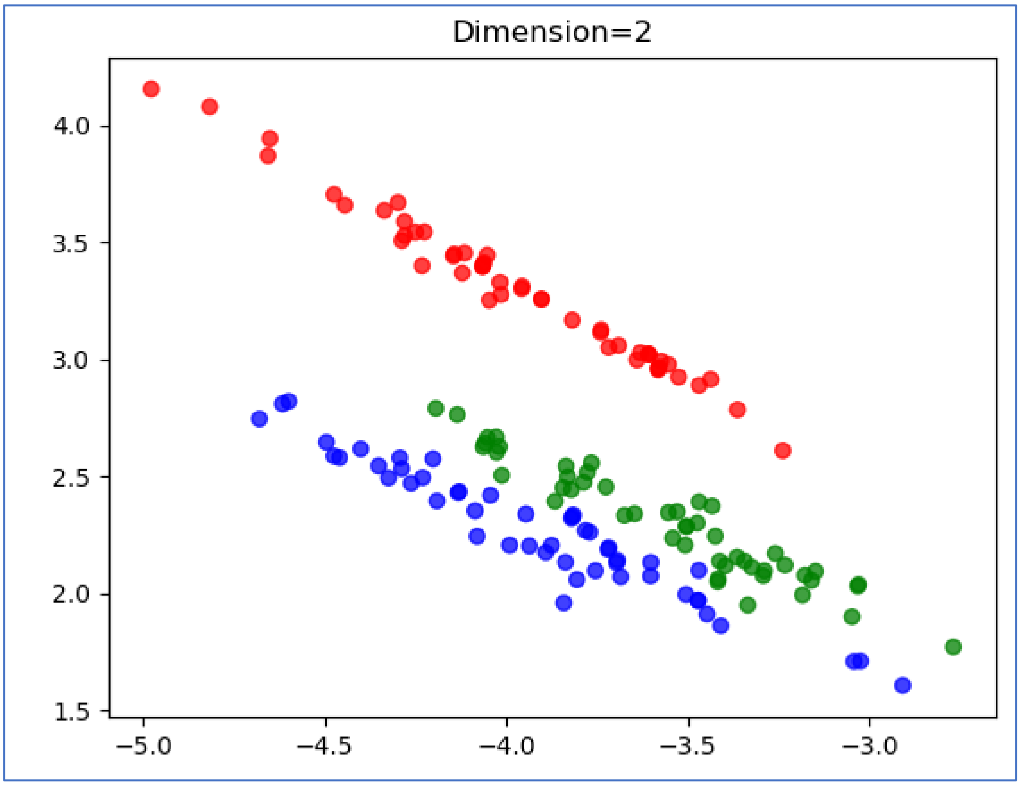
.a

```python
'''
. Computing LDA Projections.
. Return projections, projection matrix [numpy arrays]
'''
def lda(class1, class2, class3, dim):
  def _compute_scatter_within(classes):
    total_Sw = None
    for classk in classes:
      mean = np.mean(classk, keepdims=True, axis=0).tolist()[0]
      class_Sw = None
      for x in classk:
        x = np.reshape(x, (x.shape[0], 1))
        math_term = np.matmul(x - mean, x - mean)
        class_Sw = math_term if class_Sw is None else class_Sw + math_term

      total_Sw = class_Sw if total_Sw is None else total_Sw + class_Sw

    return total_Sw


  def _compute_scatter_between(classes):
    mean = np.asarray(np.mean(np.concatenate(tuple(classes), axis=0), keepdims=True, axis=0).tolist()[0])
    mean = np.reshape(mean, (mean.shape[0], 1))

    total_Sb = None
    for classk in classes:
      nk = classk.shape[0]
      meank = np.asarray(np.mean(classk, keepdims=True, axis=0).tolist()[0])
      meank = np.reshape(meank, (meank.shape[0], 1))
      math_term = nk * np.matmul(meank - mean, (meank - mean).T)
      total_Sb = math_term if total_Sb is None else total_Sb + math_term

    return total_Sb

  ## Function Entry-Point
  Sw = _compute_scatter_within([class1, class2, class3])
  Sb = _compute_scatter_between([class1, class2, class3])

  from scipy.linalg import eig
  w, v = eig(Sw, Sb)
  ev_indices = np.argsort(w)[::-1][:dim].tolist()
  V = v[ev_indices]
  data = np.concatenate(tuple([class1, class2, class3]), axis=0).T
  Y = np.matmul(V, data)
  return Y, V
```

.b

- דוגמאות עם מימד בודד לא ניתן להפרדה כמעט (יופרד עם המון שגיאות).
- דוגמאות עם 2 מימדים ניתנות להפרדה באופן כמעט מושלם.


Dimension=1
0.008
0.006
0.004
0.002
0.000
−0.002
−0.004
−0.006
−0.008
−5.0
−4.5
−4.0
−3.5
−3.0


Dimension=2
4.0
3.5
3.0
2.5
2.0
1.5
−5.0
−4.5
−4.0
−3.5
−3.0

.c

```
=== LDA Confusion Matrix By LOOCV With Dimension=1 >>
[[28. 17.  5.]
 [ 9. 33.  8.]
 [23. 16. 11.]]
 >> LDA 1-Dimensional Success-Rate 0.48 using diag covariance matrix
=== LDA Confusion Matrix By LOOCV With Dimension=2 >>
[[50.  0.  0.]
 [ 0. 47.  3.]
 [ 0.  0. 50.]]
 >> LDA 2-Dimensional Success-Rate 0.98 using same covariance matrix
```

.d

```
=== MSE Confusion Matrix By LOOCV >>
[[49.  1.  0.]
 [ 0. 33. 17.]
 [ 0.  9. 41.]]
 >> MSE Success-Rate: 0.82
```

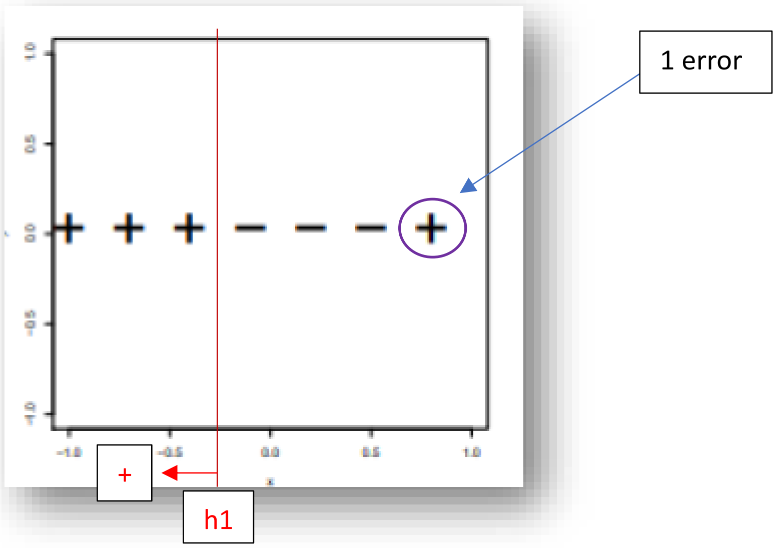
ההבדלים בין c ו-d הם בסוג המודל. בסעיף c השתמשנו במסווג בייסיאני שמאוד טבעי להשתמש כאשר הנתונים נראה כמו בסעיף b – גאוסיינים. לעומת זאת, בסעיף d השתמשנו במסווג לינארי שממזער את השגיאה הריבועית – הוא מצא פתרונות לא רעים אבל בגלל הקרבה בין מחלקות 2 ו-3 (ירוק/כחול) הוא שגה בהערכת הפתרון. ניתן לראות שאפילו שהמחלקה האדומה רחוקה מהמחלקות האחרות הוא שעה בדוגמא שקרובה למחלקה הירוקה (הקירוב הלינארי זז בכדי למזער את השגיאה הריבועית והנקודה זז למחלקה הירוקה).

- *הקוד המלא מצורף לתרגיל.*

שאלה 3:

a. כל עוד ניתן להשתמש רק במשפחת מסווגים ליניאריים התשובה היא **לא**. הסיבה לכך היא שהנתונים בנויים בצורה כזאת שבכל איטרציה האלגוריתם יבחר מסווג ליניארי **אחד** שיש לו טעות אחת לדוגמה החיובית וטעות אחת לדוגמה השלילית ומכיוון שמתחילים ממשקלים אחידים של $1/4$ נקבל ש-$\varepsilon_1 = 1/2$ ומכאן $\alpha_1 = 0$. מנוסחת עדכון המשקלים נקבל שהמשקלים של הטעויות וההצלחות ישארו זהים ולכן גם בכל האיטרציות $\alpha_2, \dots, \alpha_t$ נקבל שמשקלי המסווגים הם 0 ולכן לא נצליח לקבל $classification\ rate$ שטוב יותר ממסווג רנדומלי.

b.



c.

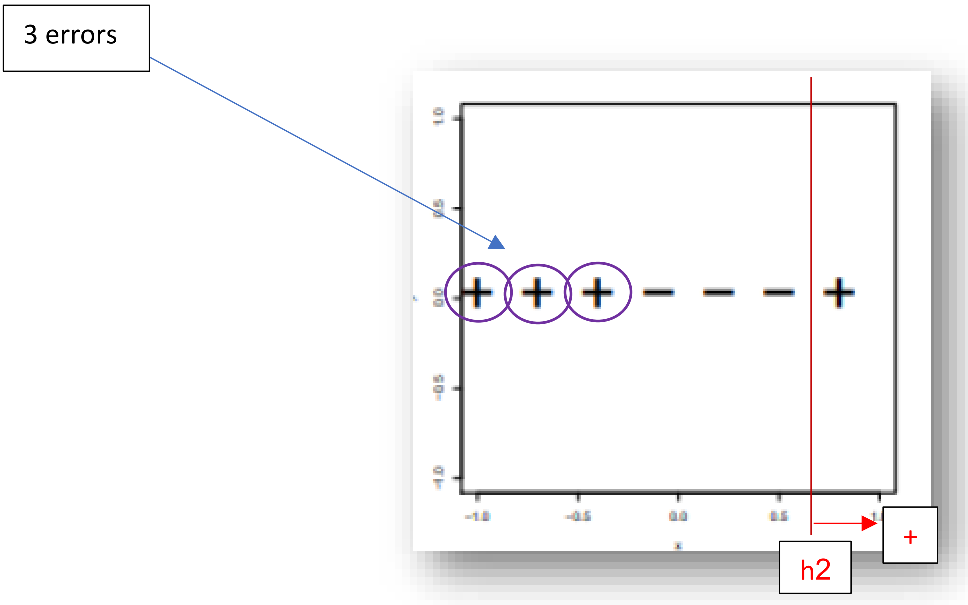
$$\boldsymbol{\varepsilon_1} = \frac{\mathbf{1}}{\mathbf{7}}, \boldsymbol{\alpha_1} = \frac{1}{2}\ln\left(\frac{1-\frac{1}{7}}{\frac{1}{7}}\right) = \frac{1}{2}\ln(6) = 0.5 * 1.791 = \mathbf{0.895}$$

ה-$classification\ accuracy$ יהיה $\frac{6}{7}$.

d.

- הדוגמאות בצד <u>הנכון</u>: $\boldsymbol{d_{right}} = \frac{1}{7} * \frac{1}{2} * \frac{1}{1-\frac{1}{7}} = \frac{\mathbf{1}}{\mathbf{12}}$
- הודמאות בצד <u>הלא נכון</u>: $\boldsymbol{d_{wrong}} = \frac{1}{7} * \frac{1}{2} * \frac{1}{\frac{1}{7}} = \frac{\mathbf{1}}{\mathbf{2}}$

$.e$



$.f$ בסיום ריצה זו הדוגמאות בעלות המשקלים הנמוכים ביותר יהיו כל הדוגמאות השליליות (אותם משקלים לכל הנקודות השליליות).

$.g$ ה-$classification\ rate$ **לא** השתפר מכיוון שבאיטרציה זו $\alpha_2 = \frac{1}{2}\ln(3) = 0.549$ ולכן עדיין לא יהיה ניתן לסווג את הדוגמא החיובית הקיצונית מימין.

כי נקבל: $classification\ rate \ \frac{6}{7} \leftarrow 0.895 * h_1 + 0.549 * h_2$.

<u>שאלה 4:</u>

$.a$ נחשב את האנטרופיה של כל אחת מהתכונות:

$$E_{price-low}[+2,-2] = 1, E_{price-med}[+2,-2] = 1, E_{price-high}[+2,-1] = 0.9182$$

$$\boldsymbol{E(price) = \frac{4}{11} * 1 + \frac{4}{11} * 1 + \frac{3}{11} * 0.9182 = 0.9776}$$

$$E_{maintenance-low}[+2,-0] = 0, E_{maintenance-med}[+2,-2] = 1, E_{maintenance-high}[+2,-3] = 0.9709$$

$$\boldsymbol{E(maintenance) = \frac{2}{11} * 0 + \frac{4}{11} * 1 + \frac{5}{11} * 0.9709 = 0.8049}$$

$$E_{capacity-2}[+1,-2] = 0.9182, E_{capacity-4}[+3,-3] = 1, E_{capacity-5}[+2,-0] = 0$$

$$\boldsymbol{E(capacity) = \frac{3}{11} * 0.9182 + \frac{6}{11} * 1 + \frac{2}{11} * 0 = 0.7958}$$

$$E_{airbag-yes}[+3,-3] = 1, E_{airbag-no}[+3,-2] = 0.9709$$

$$\boldsymbol{E(airbag) = \frac{6}{11} * 1 + \frac{5}{11} * 0.9709 = 0.9867}$$

מכיוון ש-$\boldsymbol{capacity}$ נותן לנו את האנטרופיה המינימאלית נבחר בו להיות ה-$root$ בעץ.

b. נחשב את האנטרופיה הכללית של הנתונים:

$$\boldsymbol{E}[+\mathbf{6},-\mathbf{5}] = -\frac{6}{11}\log2\left(\frac{6}{11}\right) - \frac{5}{11}\log2\left(\frac{5}{11}\right) = \mathbf{0.9940}$$

נחשב את ה-$Gain$ של כל אחת מהאפשרויות:

$$\boldsymbol{G}\left(\boldsymbol{Price}_{\{\boldsymbol{low},\boldsymbol{med}\}|\{\boldsymbol{high}\}}\right) = 0.9940 - 0.9776 = \mathbf{0.0164}$$

$$\boldsymbol{G}\left(\boldsymbol{Maintenance}_{\{\boldsymbol{high}\}|\{\boldsymbol{low},\boldsymbol{med}\}}\right) = 0.9940 - 0.9421 = \mathbf{0.0519}$$

$$\boldsymbol{G}\left(\boldsymbol{Maintenance}_{\{\boldsymbol{high},\boldsymbol{med}\}|\{\boldsymbol{low}\}}\right) = 0.9940 - 0.8108 = \mathbf{0.1832}$$

$$\boldsymbol{G}\left(\boldsymbol{Capacity}_{\{\mathbf{2}\}|\{\mathbf{4},\mathbf{5}\}}\right) = 0.9940 - 0.9445 = \mathbf{0.0495}$$

מכיוון שמדד ה-$Gain$ מחפש ערך מקסימלי נבחר במקרה זה את התכונה

$$\underline{\boldsymbol{c}.\,\boldsymbol{maintenance} - \{\boldsymbol{high},\boldsymbol{med}\}|\{\boldsymbol{low}\}}$$

c. כולם *(b, c, d)* **פרט ל-$\boldsymbol{a}.\,\boldsymbol{High\ Bias}$** מתארים את עץ ההחלטה.

עץ החלטה מבצע חיפוש היוריסטי במרחב הקלט ללא כמעט הנחות נוספת על הנתונים. סיבה זו מייצרת $\underline{low\ bias}$ בהגדרה אבל כתוצאה מ-$\underline{bias\ variance\ trade\ off}$ נקבל $\underline{(\text{b})\ high\ variance}$. יתר המאפיינים *(c, d)* נובעים ממבנה העץ והאלגוריתם (כפי שנילמד והוצגו דוגמאות בהרצאה):

c. $Lack\ Of\ Smoothness$: מאפיין את העץ מכיוון שהוא "חותך" את המרחב לפי בחירת המאפיינים בעץ וחיתוך כזה לפי הצירים הוא לא חלק.

d. $Unbounded\ parameter\ set$: מאפיין את העץ מכיוון שניתן לבצע $training$ על כמות תכונות ללא מגבלה ובכל צומת העץ יתקדם גם בלי מגבלה עד הגעתו לעלה (העץ יבחר את התכונות המסבירות ביותר עבור ההכללה).