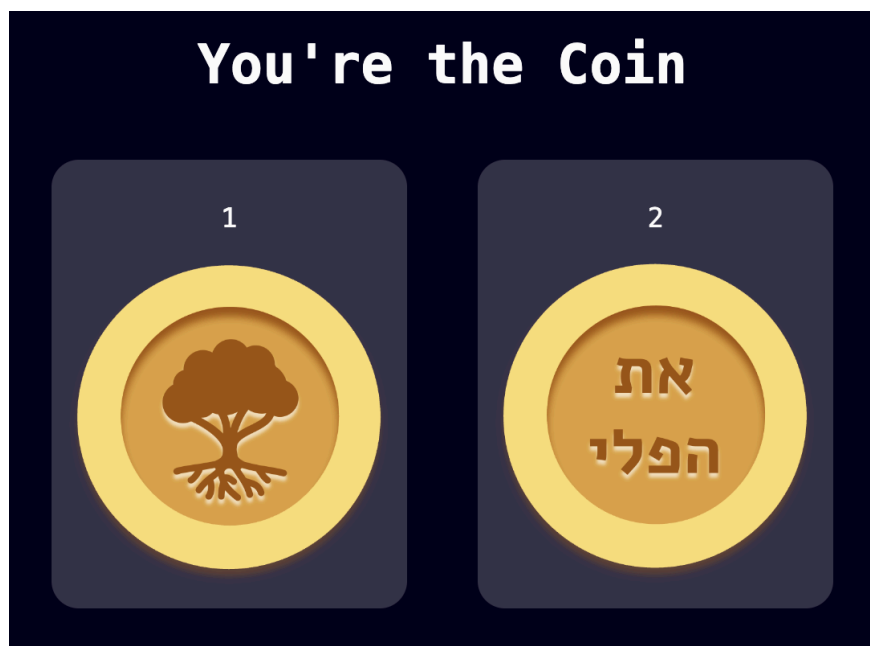




למידת מכונה על קצה המזלג



בתרגיל זה נממש את המוח מאחורי המשחק בעל העלילה הסוחפת **You're the Coin**, בו השחקן מדמיין שהוא מטבע ובודק האם הוא מסוגל לבחור בין עץ לפלי באופן אקראי, או שמא גם בלי כוונה יש דפוסים בבחירות שלו.

לאורך התרגיל נבדוק כמה אלגוריתמים שונים לחיזוי הבחירה הבאה של השחקן בהינתן היסטוריית הבחירות שלו, ונציג את מידת ההצלחה של כל אחד מהאלגוריתמים.

00 הכנות

1. הורידו למחשב את [קובץ ה-html הזה](#)
2. פתחו אותו בסביבת עבודה כלשהי וחפשו את הקטע שמתחיל ב - YOUR CODE GOES HERE
3. פתחו את הקובץ גם בדפדפן

01 שלום עולם

בתרגיל תחבשו שני כובעים:

- כובע השחקן, בו תבחרו כל פעם עץ או פלי (ע"י לחיצה על המטבעות או לחיצה על "1" ו-"2" במקלדת)
- כובע מממשי המשחק, בו תכתבו אלגוריתמים לחיזוי הבחירות של השחקן

בקובץ שקיבלתם יש כבר "אלגוריתם חיזוי" אחד ממומש - שמשמש ב - random כדי לחזות את פעולת השחקן, ולכן הוא לא מוצלח במיוחד. בכל זאת תוכלו להיעזר בו כדי להבין מה קורה פה.

בדפדפן, התחילו ללחוץ באקראי על "1" ו-"2" במקלדת (או על המטבעות), ותראו גרף שמתארך ככל שאתם משחקים יותר סיבובים. גרף זה מתאר את ניקוד האלגוריתם האקראי לחיזוי. האלגוריתם מקבל נקודה בכל פעם שהוא חוזה נכון את בחירת השחקן, ומאבד נקודה בכל פעם שהוא חוזה לא נכון.

בקוד, באזור בו עליכם לכתוב את הקוד שלכם, תראו את הפונקציה **predictRandom**. פונקציה זו היא המימוש של אלגוריתם החיזוי. כל פונקציה שתכתבו באזור זה, ואשר תתחיל במילה predict - תיכנס למשחק ותוכלו לראות את הניקוד שלה. הפונקציות בהן תממשו את האלגוריתמים מקבלות פרמטר אחד - מערך היסטוריית הבחירות של השחקן, כאשר כל בחירה היא או "H" המציינת עץ (Head) או "T" המציינת פלי (Tail). עליהן להחזיר מחרוזת פשוטה המתארת את החיזוי - כאמור, "H" עבור עץ, "T" עבור פלי.

הוסיפו שתי פונקציות חיזוי פשוטות:

1. פונקציית **predictHead**, אשר תמיד חוזה שהשחקן יבחר עץ.
2. פונקציית **predictTail**, אשר תמיד חוזה שהשחקן יבחר פלי.

שחקו שוב כעת, וודאו שהפונקציות שהוספתם מתנהגות כצפוי.

02 נקודות למחשבה

לפני שנעבור למימוש אלגוריתמים יותר מתוחכמים, נעצור לחשוב מה המטרה שלנו.

בעזרת הפונקציות שהוספנו בסעיף 01, קל לראות שכאשר השחקן בוחר תמיד אותו ערך, למשל עץ, פונקציית החיזוי **predictHead** טובה באופן משמעותי מהחיזוי האקראי. אולם אם תנסו באמת לבלבל את המחשב ולבחור באופן אקראי, כנראה שהפונקציות שחוזות ערכים קבועים לא יהיו טובות בהרבה מ- random (אלא אם יש לכם נטייה מובהקת להעדיף אחד מהמטבעות על השני באופן עקבי).

אנחנו רוצים לכתוב פונקציות חיזוי יותר חזקות, כך שגם כאשר ננסה לבלבל את המחשב, הוא יעלה על הדפוסים שלנו ויצליח לקבל ניקוד גבוה משמעותית מ- random (אלא אם נצליח להיות אקראיים באמת).

נקודה נוספת למחשבה: אם נכתוב אלגוריתם שנוטה לקבל ציונים דומים ל- random, זה אומר שהוא לא מצליח לנבא אותנו. אם נכתוב אלגוריתם שמקבל ציונים טובים יותר משמעותית מ- random, זה אומר שאנחנו יחסית צפויים והצלחנו לכתוב אלגוריתם שחוזה את הבחירה שלנו.

מה המשמעות של אלגוריתם שמקבל ציון **גרוע** משמעותית מ- random, באופן עקבי?

03 חימום

נניח בכל זאת שיש לשחקן העדפה מובהקת לאחד המטבעות, עץ או פלי, ונרצה פונקציית חיזוי שמצליחה למצוא את ההעדפה ולנבא לפיה. ממשו פונקציה כזו: מצאו את הבחירה המועדפת לפי היסטוריית הבחירות, והחזירו אותה.

לדוגמה: נניח שזו היסטוריית הבחירות של השחקן:

HTTHTTHTTHTTTHHTHT

השחקן בחר עץ 8 פעמים, ופלי 10 פעמים, ולכן פונקציית החיזוי תחזיר פלי. לאורך המשחק הבחירה שמתקבלת רוב הפעמים עשויה להשתנות (למשל אם השחקן יבחר עץ 3 פעמים או יותר מהמצב הנוכחי), והפונקציה שכתבתם תשקף את זה.

04 מפשילים שרוולים

הגיע הזמן לכתוב פונקציית חיזוי קצת יותר מתוחכמת, שבכל זאת קצת דומה לפונקציה הקודמת (ואף יכולה להיעזר בה). נניח שיש חוקיות כלשהי בבחירות השחקן - למשל, שהוא נוטה לבחור הרבה עץ ברצף, הרבה פלי ברצף, ומדי פעם מחליף ביניהם. או, שהוא מנסה להיות "לא עקבי" וכמעט תמיד מחליף (עץ-פלי-עץ-פלי). במקרים כאלה, הסתכלות על הבחירה האחרונה יכולה לעזור לנו בחיזוי הבחירה הבאה. כיצד נעשה זאת? נתבונן שוב בהיסטוריית הבחירות מהסעיף הקודם:

HTTHTTHTTHTTTHHTHT

במקום להסתכל על הבחירה הכי נפוצה באופן כללי, נשים לב שהבחירה האחרונה של השחקן היא פלי, ולכן נסתכל מה הבחירה הכי נפוצה **אחרי בחירת פלי**. נסמן בחירות אלו (ודאו שאתם מבינים את הסימון):

HTTHTTHTTHTTHHHTHT

במקרה זה, לאחר פלי נבחר עוד פלי 4 פעמים, ונבחר עץ 5 פעמים, ולכן נחזה עץ. (מה היינו חוזים אם הבחירה האחרונה של השחקן הייתה עץ?)

ממשו פונקציית חיזוי אשר משתמשת במנגנון שתואר כאן כדי לבצע את החיזוי.

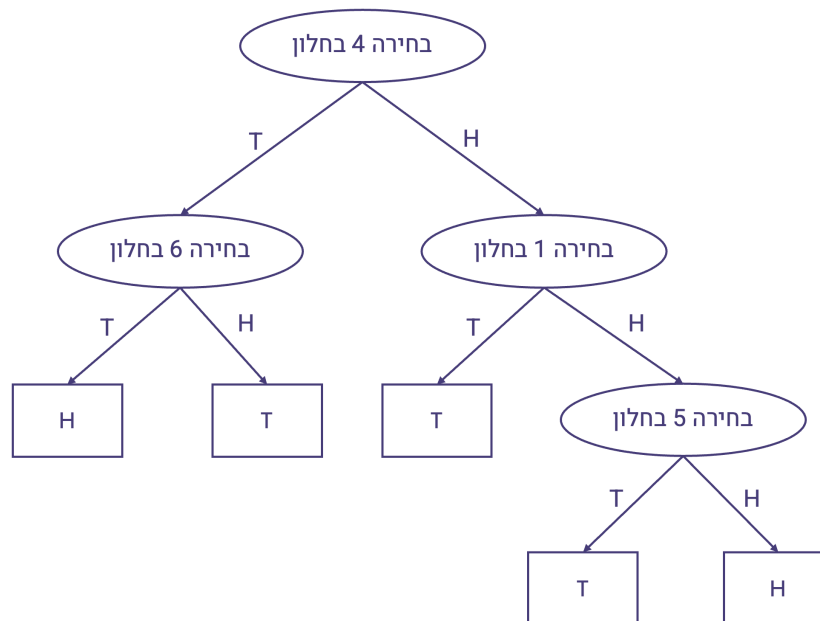
05 כשהגלים מתחזקים... (בנוס עסיסי)

מזל טוב, הגעתם לליגה של הגדולים!

בסעיף זה מתוארת גרסה בסיסית של אלגוריתם חיזוי אמיתי מתחום למידת המכונה - [עץ החלטה](#). המימוש שלו קצת יותר מורכב ודורש העמקה (מומלץ להתייחס אליו כמו פרויקט קטן), אז קחו נשימה ו... יאללה צוללים.

בסעיף האחרון הסתכלנו על הבחירה האחרונה של השחקן כדי לנסות לחזות את הבחירה הבאה, במטרה לזהות דפוסים. אם שיחקתם בעצמכם אתם בוודאי יודעים שיכולים להיות דפוסים של רצפים יותר ארוכים, והיינו רוצים לנסות ללמוד גם אותם. אפשר לנסות להשתמש באותה שיטה של הסעיף הקודם - להסתכל על 6 הבחירות האחרונות ולחזות את הבחירה הכי נפוצה אחרי כל הפעמים בהיסטוריית הבחירות בהן היו אותן 6 בחירות (לא חייב להיות 6, סתם כדוגמה לאורך חלון שאפשר להסתכל עליו). אבל זו אסטרטגיית חיזוי מאוד ספציפית - כדי שנקבל סטטיסטיקה משמעותית על רצפים באורך $7 = 1 + 6$ נצטרך לשחק המון, ואולי יש משהו יותר חכם שאפשר לעשות. אז מה נעשה?

עץ החלטה הוא מודל למידת מכונה קלאסי, בו חיזוי על רשומה מתבצע כרצף שאלות לגבי המאפיינים של הרשומה, כשכל שאלה מובילה לצומת אחר עד שנגיע לחיזוי. במקרה שלנו, אם מסתכלים על חלון של 6 בחירות אחרונות, עץ החלטה יכול להיראות כך:



הבהרה לגבי **חלון הבחירות האחרון**: נניח שהיסטוריית הבחירות של השחקן היא כזאת:

HTTHTTHTTHTTTHHTHT

זהו חלון הבחירה בגודל 6 האחרון:

HTTHTTHTTHTTHHHTHT

אם נסתכל על כל חלונות הבחירה בגודל 6 בהיסטוריית הבחירות, יחד עם הבחירה שהגיעה אחריהן:

בחירה אחרי החלון	חלון בחירה
H	HTTHTT
T	TTHTTH
T	THTTHT
H	HTTHTT
T	TTHTTH
...	...
T	TTHHTH

בחלון הראשון בדוגמה (HTTHTT), "בחירה 1 בחלון" היא H, "בחירה 3 בחלון" היא T, "בחירה 6 בחלון" היא גם T, וכן הלאה.

חזרה לעצי החלטה. נחלק את המימוש לשני שלבים - ייצוג עץ החלטה, ובניית עץ החלטה לפי היסטוריית הבחירות של השחקן.

השלב הראשון הוא יותר תכנותי - החליטו כיצד אתם הולכים לייצג עץ החלטה (בלי קשר לאלגוריתם שיבנה את עץ ההחלטה הספציפי בהתאם להיסטוריית השחקן), וממשו את הבסיס של ייצוג זה (אם נדרש). אפשר ללכת על כיוון OOP ולעשות מחלקה של **TreeNode** עם מצביעים לעוד שני **TreeNode**s, אפשר להשתמש במבני נתונים פשוטים (מערכים, אובייקטים) ופונקציות מתאימות שיודעות לעבוד איתם, או כל שיטה אחרת שתבחרו.

בשלב השני נעסוק בבניית עץ בהתאם להיסטוריית בחירות כלשהי. אנחנו רוצים לבנות עץ שמצליח כמה שיותר מהר (תוך כמה שפחות פיצולי H/T) להפריד בין חלונות שאחריהם נבחר עץ, לחלונות שאחריהם נבחר פלי. גם את התהליך הזה נעשה בשלבים.

בשלב הראשון, **ממשו פונקציית עזר** שאוספת, מתוך היסטוריית בחירות, את כל החלונות בגודל 6 (או כל גודל אחר שתבחרו), יחד עם הבחירה שהגיעה אחרי החלון. הפונקציה צריכה להחזיר משהו בסגנון הטבלה לעיל, בשיטת ייצוג לבחירתכם.

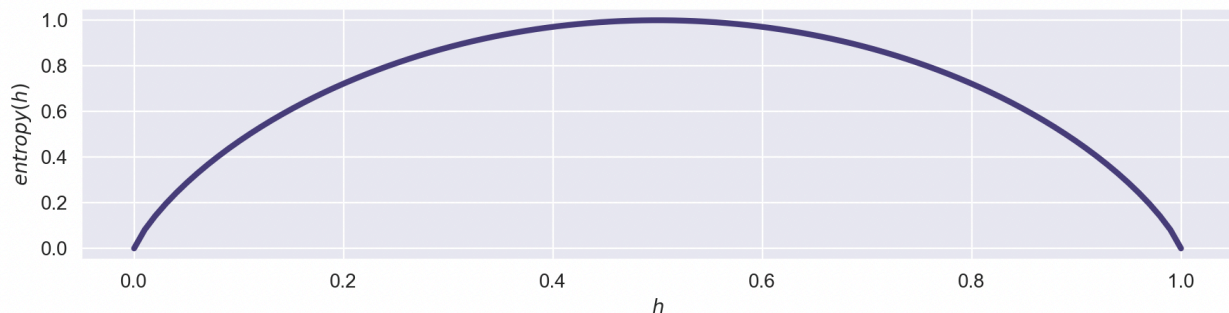
בשלב השני, **נגדיר מדד** להטרוגניות (ערבוב) של אוסף בחירות מסוים. אנחנו רוצים שהמדד יהיה גבוה כאשר באוסף יש הרבה בחירות משני הסוגים, ונמוך כאשר האוסף יחסית אחיד. המדד שנגדיר נקרא **אנטרופיה** (סטטיסטית, להבדיל מפיזיקלית), ומוגדר כך: נניח שבאוסף בחירות מסוים, h הוא שיעור עץ מתוך הבחירות (כלומר מספר בין 0 ל-1), ו- t הוא שיעור פלי מתוך הבחירות ($h + t = 1$). האנטרופיה של האוסף מוגדרת כ:

$$\text{entropy}(h, t) = -(h \cdot \log_2 h + t \cdot \log_2 t)$$

מכיוון ש- $h + t = 1$ אפשר גם להגדיר:

$$\text{entropy}(h) = -(h \cdot \log_2 h + (1 - h) \cdot \log_2 (1 - h))$$

הגרף הבא מראה את התנהגות ההגדרה שבחרנו, וכיצד האנטרופיה גבוהה כאשר יש ערבוב בין עץ לפלי באוסף הבחירות, ונמוכה כאשר האוסף יחסית מזוקק:



ממשו פונקציה המחשבת את האנטרופיה של אוסף בחירות נתון.

בשלב השלישי הגענו סוף-סוף למימוש האלגוריתם לבניית עץ החלטה. קודם נתאר את האלגוריתם ולאחר מכן נסביר.

נניח שהמשתנה X מכיל מערך עם כל החלונות בגודל 6 בהיסטוריית הבחירות של השחקן, והמשתנה y מכיל מערך עם כל הבחירות שהגיעו אחרי החלונות. X ו- y מתואמים, כלומר, האיבר ה-11 ב- y מתאר את הבחירה שהגיעה אחרי הרצף הנמצא במקום ה-11 ב- X .

בניית עץ החלטה (X, y) :

- אם יש ב- X פחות מ-5 חלונות:
 - נחזיר עץ החלטה עם צומת אחד שתמיד חוזה את הבחירה הנפוצה ב- y
 - נחשב את האנטרופיה של y
 - עבור כל פיצול אפשרי idx (בחירה 1 בחלון, בחירה 2 בחלון וכו'):
 - נשמור ב- X_H, y_h את כל החלונות / הבחירות שבהם החלון במקום idx הוא H
 - נשמור ב- X_T, y_t את כל החלונות / הבחירות שבהם החלון במקום idx הוא T
 - נחשב את האנטרופיה של y_h, y_t (בנפרד) ונשקלל כך לאנטרופיה חדשה כללית:

$$newEntropy = \frac{y_h.length \cdot entropy(y_h) + y_t.length \cdot entropy(y_t)}{y_h.length + y_t.length}$$
 - נחשב את ההפחתה באנטרופיה לפי הפיצול הנוכחי -
 - $improvement = entropy(y) - newEntropy$
 - נשמור את כל המשתנים שמצאנו ($X_H, X_T, y_h, y_t, improvement, idx$) איפשהו
- נמצא את הפיצול הכי טוב, כלומר עם ה- $improvement$ הכי גדול, מבין כל הפיצולים שבדקנו
- אם ה- $improvement$ הכי טוב קטן מ-0.1:
 - נחזיר עץ החלטה עם צומת אחד שתמיד חוזה את הבחירה הנפוצה ב- y
- אחרת, נשלוף את המשתנים idx, X_H, X_T, y_h, y_t של הפיצול הכי טוב
- ענף ימין = בניית עץ החלטה (X_H, y_h)
- ענף שמאל = בניית עץ החלטה (X_T, y_t)
- נחזיר עץ החלטה שבודק האם הבחירה ה- idx בחלון היא עץ או פלי. אם עץ, נעביר את החישוב לענף ימין; אחרת, נעביר את החישוב לענף שמאל.

הסבר: אלגוריתם אולי לא הכי פשוט, אבל בכל זאת, אלגוריתם שלומד לנבא את הבחירות שלנו לא הולך ברגל. אז מה קורה פה?

קודם כל בוודאי שמתם לב שמדובר ב**אלגוריתם רקורסיבי**. בכל שלב האלגוריתם מנסה למצוא את הפיצול שמפריד הכי טוב בין מקרים שבהם אנחנו צריכים לחזות עץ, למקרים שבהם אנחנו צריכים לחזות פלי.

תנאי העצירה של הרקורסיה הוא אחד משניים:

1. אין לנו מספיק חלונות / בחירות כדי ליצור עץ (הגדרתי 5 באופן יחסית שרירותי, מוזמנים לשחק עם זה)
2. לא מצאנו פיצול שמוסיף להפרדה מספיק כדי להצדיק הוספת רמה בעץ (גם פה בחרתי 0.1 באופן יחסית שרירותי)

אם עצרנו, נחזיר **עץ החלטה עם צומת אחד** שפשוט חוזה את הערך הנפוץ מבין y (כמו בסעיף 03).

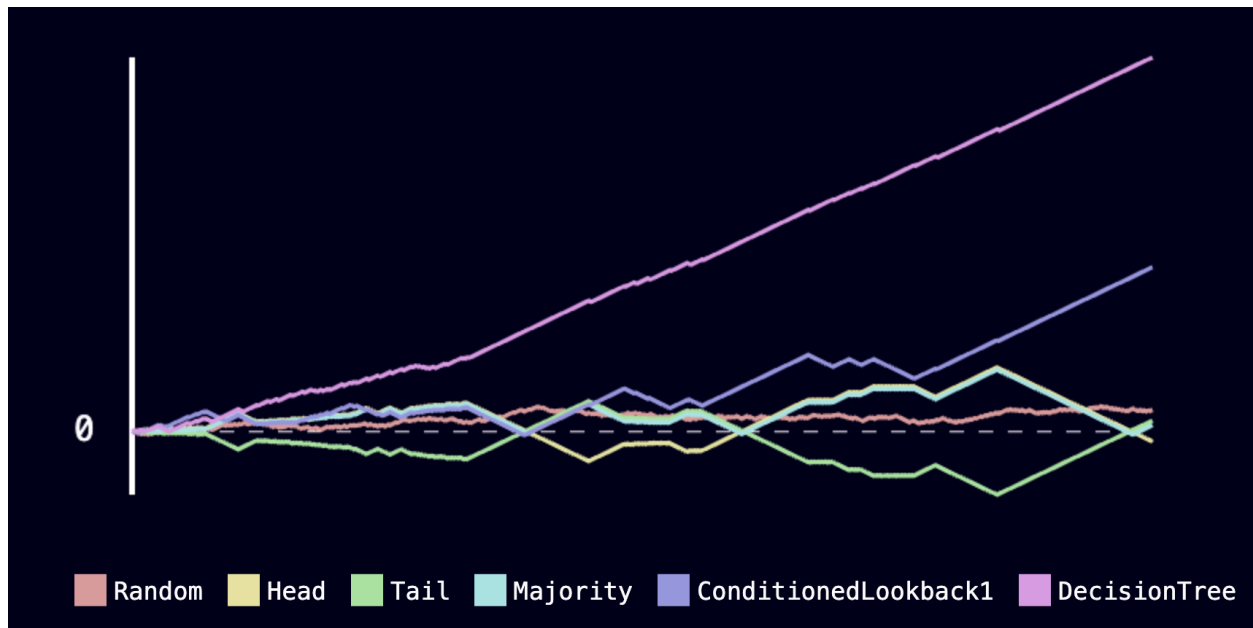
אם לא עצרנו, ז"א שמצאנו פיצול שאנחנו מעוניינים להפריד לפיו. במקרה זה נפצל את X ואת y לשני המקרים (עץ או פלי, במיקום בחלון שנקבע לפי הפיצול הכי טוב), וניצור עוד שני עצי החלטה לכל אחד מהמקרים האלו באמצעות **קריאה רקורסיבית מתאימה**. לאחר מכן נרכיב עץ אשר בודק את המיקום בחלון לפיו אנחנו מפצלים, ומעביר לענף המתאים.

איך מודדים את השיפור בהפרדה בפיצול? ניזכר באנטרופיה שהגדרנו - ככל שהיא יותר גבוהה, יש יותר ערבוב בין הבחירות. אנחנו יודעים את האנטרופיה של האוסף הכולל, ולכל פיצול נבדוק את **האנטרופיה הממוצעת** של כל אחד מהאוספים אחרי הפיצול. בשביל הממוצע אנחנו משתמשים ב**ממוצע משוקלל** כדי לשקף את העובדה שהפיצול לא חייב להיות שווה בגודל בין שני החלקים (אפשר לוודא שכאשר גודל שני החלקים אחרי הפיצול זהה, מדובר בממוצע פשוט). השיפור שנותן הפיצול הוא **ההפרש** בין האנטרופיה לפני הפיצול, לממוצע האנטרופיות אחרי.

[בקישור זה](#) תמצאו הסבר מעולה על עצי החלטה, עם אנימציות שמסבירות מה קורה בכל שלב.

06 מה כל זה אומר עלינו?

כל אחת ואחד מאיתנו שונים, אז אני לא יודע אילו תוצאות קיבלתם וכמה אתם צפויים.
אלה התוצאות שאני קיבלתי:



גם כשניסיתי לבלבל את המחשב, החיזוי שמסתכל על הבחירה הקודמת היה יחסית מוצלח, ועץ ההחלטה היה מאוד מוצלח. אם קיבלתם תוצאות דומות - משמע שאתם לא אקראיים כמו שאולי חשבתם):

דרך נחמדה לבדוק שהאלגוריתמים שלנו לא מרמים היא להזין באמת מספרים אקראיים (או לפחות [פסאודו-אקראיים](#)) ולוודא שאף אחד מהאלגוריתמים לא מצליח יותר טוב מחיזוי אקראי. כדי לנסות זאת, פתחו את ה - devTools והדביקו את הקוד הבא:

```
for (var i = 0; i < 200; i++) {
  document.querySelectorAll('.coin')[Math.round(Math.random())].click();
}
```