מבוא לבינה מלאכותית – תרגול 2

נושאים:

- **עקרון ERM, למידת PAC, ממד VC וניפוץ מרחבים**
- **סיווג בינארי באמצעות מפריד לינארי – Perceptron ו-SVM (hard margin)**

עקרון ERM:

ממשיכים בחלק התאורטי שלפני האלגוריתמים ללמידת מכונה. את התרגול של היום נתחיל בניסוח עקרון שדומה לרעיונות שכבר ראינו בתרגול הקודם, וגם בו אפשר להשתמש כעקרון מנחה ללמידת מכונה.

נתחיל מ-4 מרכיבים שמכילה כל מערכת לומדת (ולמעשה ראינו דוגמה שלהם כבר בתרגול הקודם):

1. **מרחב הקלט** – $\mathcal{X}$. המרחב יכול להיות מממד קבוע (סופי) או משתנה (למשל, אם כל דגימה היא משפט באנגלית, או קטע קול), ואוסף הדגימות שלנו הוא תת קבוצה של המרחב.
   כשממדי מרחב הקלט מתארים תכונות מסוימות של דגימה (למשל, כל דגימה מייצגת דירה מתוך אוסף דירות וכל ממד מייצג מדד עליה, כמו השטח שלה, כמה חדרים וכדומה), נקרא להם **פיצ'רים**.
2. **מרחב התיוגים** – $\mathcal{Y}$. בתרגול שעבר, ראינו משימה של רגרסיה, ולמעשה שם מרחב התיוגים היה $\mathcal{Y} = \mathbb{R}$. ניתן לבצע משימות תיוג אחרות, ובהן תיוג בינארי (בדרך כלל $\mathcal{Y} = \{0,1\}$ או $\mathcal{Y} = \{\pm 1\}$), תיוג מולטיקלאס ($\mathcal{Y} = \{1,2,\dots,k\}$, או $\mathcal{Y} = \{0,1,\dots,k-1\}$) ועוד.
   <u>הערה</u> – לא תמיד נעשה משימות של תיוג. יש משימות שבהן הדאטא לא בהכרח מתויג ועושים בהן דברים קצת שונים, נראה כמה מהן בהמשך הקורס.
3. **מחלקת ההשערות** (hypothesis set) – $\mathcal{H}$. זו משפחה של פונקציות $\{h_\theta : \mathcal{X} \to \mathcal{Y}\} \subseteq \mathcal{Y}^{\mathcal{X}}$ תלויות פרמטרים, שמתוכן נבחר את המסווג המועדף עלינו ביותר לצורך המשימה. החיזוי של כל תיוג של דגימה נתונה מתבצע על סמך הפונקציה הנבחרת - $\hat{y} = h_\theta(x)$.
4. **מדד לשגיאות (פונקציית loss)** - $l(y, \hat{y})$. זו פונקציה שמודדת את השגיאה שלנו בחיזוי לעומת הערך האמיתי. בדרך כלל מתקיים $l(y,y) = 0$.

עקרון ERM (Empirical Risk Minimization) הוא עקרון לפיו הולכים בחיפוש הפרמטרים הטובים ביותר על סמך מדגם נתון:
יהי מדגם $\{(x_i, y_i)\}_{i=1}^n$ כך ש-$(x_i, y_i) \sim \rho$. אזי, ההערכה שלנו לפרמטרים המועדפים ביותר θ^* תתבצע לפי $\theta^* = \arg\min_\theta \frac{1}{n}\sum_{i=1}^n l\big(y_i, h_\theta(x_i)\big)$.

<u>הערה</u> – מה שמכונה risk הוא הביטוי $\mathbb{E}_{(x,y)\sim\rho}[l(y,h_\theta(x))]$. הממוצע הוא הערכה אמפירית שלו, על סמך מדגם סופי.

<u>הערה</u> – יש עקרונות דומים נוספים שאפשר להשתמש בהם למציאת θ^* המועדף ביותר, ובהם כאלה שהפונקציה הממוזערת מערבת ביטויים נוספים שחלקם ראינו וחלקם נראה בתרגול הזה או בהמשך (רגולריזציה, שורש ממד VC של מחלקת ההשערות, dropout ועוד).

למידת PAC:

ההגדרת למחלקת השערות PAC-learnable היא ארוכה וכתובה מטה, בצורה שאני מקווה שמסודרת מספיק ברור לפי שורות. המשמעות של ראשי התיבות PAC היא Probably Approximately Correct.

מחלקת השערות $\mathcal{H}$ נקראת **PAC-learnable** אם
יש פונקציה $m_{\mathcal{H}}:(0,1)^2\to\mathbb{N}$ ואלגוריתם למידה,
כך שלכל $\epsilon\in(0,1)$, $\delta\in(0,1)$, התפלגות D על תחום $\mathcal{X}$ ופונקציה $f:\mathcal{X}\to\mathcal{Y}$,
קיים $m(\epsilon,\delta)$, כך שלכל מספר דגימות $m\geq m(\epsilon,\delta)$ שמתפלגות i.i.d. לפי D ומתויגות לפי f,
הרצה של האלגוריתם תחזיר השערה h
כך שבסיכוי של לפחות $1-\delta$, פונקציית ה-loss לפי h תהיה קטנה או שווה ϵ.

נשים לב שיש לנו בהגדרה שני פרמטרים: ϵ, המבטא כמה אנחנו מדויקים (מה שמתקשר ל-approximately correct בשם), ו-δ, המבטא כמה אנחנו בטוחים שהמודל שלנו צודק (מה שמתקשר ל-probably).

האינטואיציה מאחורי ההגדרה היא שמחלקת השערות עונה על ההגדרה אם אנחנו יכולים למצוא בה פונקציה (ללמוד את הפרמטרים שלה) באמצעות אלגוריתם למידה כלשהו, שיביא אותנו לשגיאה נמוכה (חסומה על ידי ϵ) בהסתברות גבוהה (לפחות $1-\delta$). ההסתברות נובעת מבחירת קבוצת האימון (המדגם שניקח לא תמיד מייצג היטב את ההתפלגות המקורית, ולכן חוסר הוודאות). היכולת תלויה גם בבחירת גודל המדגם, כי נוכל להכליל טוב ככל שיהיה לנו מדגם גדול יותר מהתפלגות הקלט.

הוכחה שמחלקות הן PAC-learnable על פי הגדרה היא לפעמים ארוכה ומסובכת. לכן, פותחה תיאוריה שמפשטת יותר את הדרך למצוא האם ממחלקה היא PAC-learnable.

ממד VC וניפוץ מרחבים:

נגדיר 3 הגדרות:

<u>הגדרה</u> – תהי מחלקת השערות $\mathcal{H}$. תהי קבוצה $C = \{c_1, c_2, \dots, c_n\} \subseteq \mathcal{X}$ סופית. **הגבלה (restriction)** של $\mathcal{H}$ ל-C היא קבוצת הפונקציות ב-$\mathcal{H}$ שמצומצמות ל-C. נסמן: $\mathcal{H}_C = \{(h(c_1), \dots, h(c_n)) : h \in \mathcal{H}\}$ (אוסף של וקטורים n-ממדיים).

<u>הגדרה</u> – מחלקת השערות **מנפצת (shatters) קבוצה סופית $C \subset \mathcal{X}$** אם כל הפונקציות האפשריות מ-C למרחב התיוגים נמצאות ב-$\mathcal{H}_C$. למשל, אם מרחב התיוגים הוא $\{0,1\}$, אז ההגבלה של מחלקת ההשערות צריכה להיות בגודל $2^{|C|}$.
משמעות ההגדרה היא שעבור הקבוצה C, לא משנה איזה תיוג ניתן לכל אחת מהנקודות, ניתן יהיה למצוא פונקציה במחלקת ההשערות שמתאימה לקומבינציית התיוגים הזו.

<u>הגדרה</u> – **ממד VC (Vapnik-Chervonenkis dimension)** של מחלקת השערות $\mathcal{H}$ הוא הגודל המקסימלי של קבוצה $C \subset \mathcal{X}$ שהמחלקה יכולה לנפץ. אם לכל גודל של קבוצה, $\mathcal{H}$ מצליחה לנפץ קבוצה בגודל הזה, אז ממד VC של המחלקה יהיה אינסוף. נסמן: $VCdim(\mathcal{H})$.

דוגמה:

מצאו את ממד VC של המחלקה הבאה, כאשר $\mathcal{X} = \mathbb{R}^2$, $\mathcal{Y} = \{0,1\}$:

$$\mathcal{H} = \left\{ h_{(a_1,a_2,b_1,b_2)}(x) = \begin{cases} 1, & x \in (a_1,a_2) \times (b_1,b_2) \\ 0, & otherwise \end{cases} \mid a_1 \le a_2, b_1 \le b_2 \right\}$$

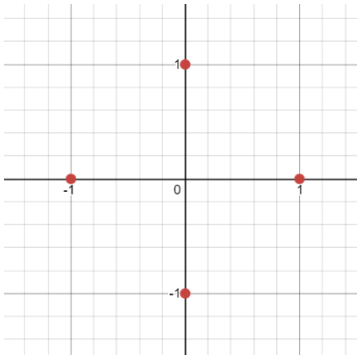
פתרון:

נשים לב שהמחלקה היא מחלקה של מלבנים. לפי כל פונקציה מהמחלקה, אם נקודה במרחב נמצאת בתוך המלבן שמוגדר על ידי הפונקציה, הפונקציה תחזה שתתויג 1, ואחרת תחזה תיוג 0.

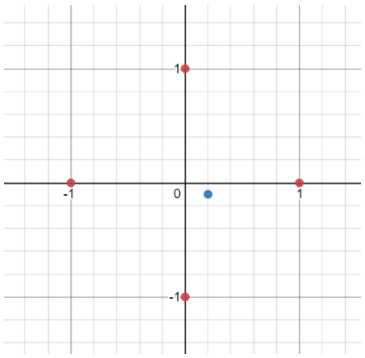
כעת, כדי למצוא שממד VC של מחלקה הוא d, צריך להראות:
א. שיש קבוצה בגודל d שהמחלקה מנפצת.
ב. שלכל קבוצה בגודל $d+1$, המחלקה לא מנפצת את הקבוצה.

נוכיח שהממד הוא 4:

א. קל למצוא קבוצה בגודל 4 שהמחלקה מנפצת (למשל, צורת יהלום כמוצג למטה. בדקו שעבור כל קומבינציה של אחדות ואפסים כתיוגים של כל אחת מארבע הנקודות, אפשר למצוא מלבן שיכיל אך ורק את הנקודות בעלות התיוג 1).



ב. לכל קבוצת נקודות בגודל 5 ב-$\mathbb{R}^2$, נסמן (בלי הגבלת הכלליות) ב-c_1 נקודה שהיא "אמצעית", במובן כזה שהקואורדינטות שלה הן לא הקיצוניות ביותר בשני הממדים (ראו מטה). נקודה זו תמיד קיימת, אבל לא ניתן למצוא פונקציה שתתאים לתיוגים $(0,1,1,1,1)$.



בין ממד VC ובין PAC-learning ישנו קישור חשוב מאוד: המשפט היסודי של הלמידה החישובית קובע, בין היתר, **שמחלקת השערות היא PAC-learnable אם ורק אם מימד VC שלה סופי**.

מסווגים לינאריים – Perceptron, SVM:

כיסינו מספיק תאוריה, לפחות לעכשיו. כעת, נתחיל ללמוד על אלגוריתמים קונקרטיים ללמידה.
הסוג הראשון של אלגוריתמים שנלמד הוא מסווגים לינאריים בינאריים.
אלה למעשה על-מישורים, שהחיזוי עבור נקודות שנמצאות מעליהם הוא 1 ומתחתיהם 0 (או -1, תלוי איך בוחרים את מרחב התיוגים).

נתאר כל פונקציה כזו באמצעות הפרמטרים שלה (שמאוחר יותר נלמד): $(\vec{w}, b)$, כך שכל נקודה במרחב $\overrightarrow{x_i}$ תהיה מעל המישור אם $\vec{w} \cdot \overrightarrow{x_i} + b > 0$ ומתחתיו אם $\vec{w} \cdot \overrightarrow{x_i} + b < 0$. לכן, לשם נוחות, החיזויים על סמך המסווגים יהיו $\hat{y_i} = sign(\vec{w} \cdot \overrightarrow{x_i} + b)$.
בדרך אחרת ושקולה, נחליף את $(\vec{w}, b)$ ב-$\vec{w}$ בלבד, וכל נקודה $\vec{x}$ נכתוב כ-$(\vec{x}, 1)$. בנוסף, הדרישה לכל נקודה ניתנת להחלפה בדרישה ש- $y_i\vec{w} \cdot \overrightarrow{x_i} > 0$.

הערה – ניתן להוכיח כי מימד VC של מסווגים לינאריים ב-$\mathbb{R}^d$ הוא $d + 1$, לכן מפרידים כאלה הם PAC-learnable.

Perceptron:

זהו אחד האלגוריתמים שבאמצעותם ניתן ללמוד משקלים $\vec{w}$ להפרדה של הדאטא. באלגוריתם הזה, המשקלים מתעדכנים online, כלומר בכל פעם בוחנים נקודה אחת מתוך הדאטא. אם צדקנו בסיווג עבורה, המודל לא יתעדכן. אחרת, המשקלים $\vec{w}$ יעודכנו בהתאם לטעות, במטרה להקטין אותה.
נראה זאת בצורת אלגוריתם:

<u>קלט:</u> סט האימון, $(\overrightarrow{x_1}, y_1), \dots, (\overrightarrow{x_n}, y_n)$.
נאתחל את המשקולות $\vec{w}^1$.
לכל איטרציה t:

נעבור על סט האימון (בין אם בסדר הנתון או לא).
אם קיים i כך ש-$y_i\vec{w}^t \cdot \overrightarrow{x_i} \leq 0$,
נעדכן את המשקולות: $\vec{w}^{t+1} = \vec{w}^t + y_i\overrightarrow{x_i}$.

<u>פלט:</u> וקטור המשקולות הסופי $\vec{w}^{final}$.

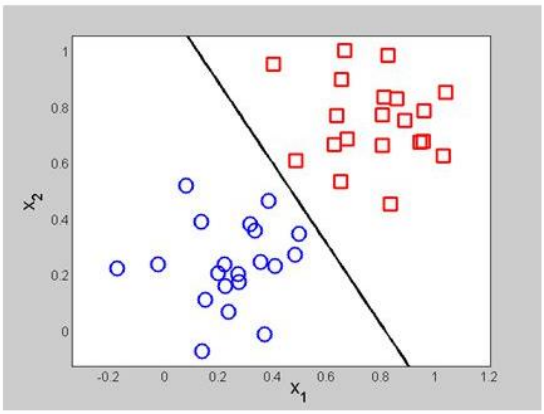
<u>הערות:</u>

- לפי משפט, האלגוריתם ייעצר אחרי מספר איטרציות סופי (בהנחה ויש הפרדה). אולם, מספר זה יכול להיות גדול מאוד.
- בצורה מעט חכמה יותר, צעד העדכון של המודל הוא $\vec{w}^{t+1} = \vec{w}^t + \eta_t y_i \vec{x_i}$, כאשר η_t, שנקרא "קצב הלימוד" (learning rate), קובע כמה קיצוני יהיה צעד העדכון שמבצעים (כמו שניתן לראות, הוא יכול להיות קבוע או תלוי איטרציה).
- להבנה, היעזרו ב-GIF בקישור הבא. שימו לב כי המסווג (שהניחו מראש שנלמד ללא bias) מתואר בעזרת הווקטור הירוק. בכל פעם מוצאים נקודה שגויה שתסומן באדום, משתמשים בווקטור הכתום ($\vec{x_i}$ המתאים) לבניית צעד העדכון המופיע באדום ($\eta y_i \vec{x_i}$).

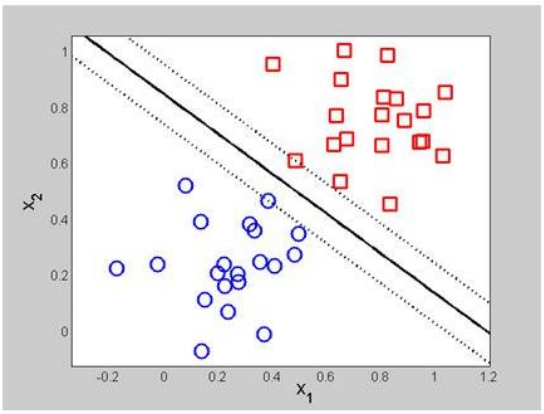
SVM (Support Vector Machine):

זהו אלגוריתם אחר למציאת מסווג לינארי לדאטא. באלגוריתם זה, מחפשים מישור מפריד כזה שיהיה רחוק ככל האפשר מהנקודות בדאטא, כלומר ישאיר שוליים (margin) רחבים.
שלא כמו פרספטרון, שבו כל פתרון שבו המסווג מפריד בין הדגימות החיוביות לשליליות בדאטא לגיטימי עבורנו, ב-SVM דואגים גם לכך שיהיה בעל יכולת הכללה גבוהה ככל האפשר.
כלומר, המפריד ב-SVM נלמד ככה שנוכל להיות הכי בטוחים שאפשר בכך שאם נדגום נקודה חדשה מהדאטא היא תסווג נכון.

Perceptron:



SVM:



את המשימה הזאת ניתן לבצע בשני אופנים – על כל הדאטא יחד או online כמו קודם. בתרגול הזה נלמד את הדרך הראשונה, המכונה "hard margin" (בה אנחנו מניחים שהדאטא ניתן להפרדה על ידי מסווג לינארי כלשהו).

כאשר לומדים על כל הדאטא, בעיית חיפוש הפרמטרים היא בעיית האופטימיזציה הבאה:

$$(w^*, b^*) = \arg \max_{(w,b):\|w\|_2=1} \min_{i\in\{1,\dots,n\}} |w \cdot x_i + b|$$

$$s.t. \qquad y_j\big(w \cdot x_j + b\big) > 0 \; \forall j \in \{1, \dots, n\}$$

שימו לב שכאשר $\|w\|_2 = 1$, הביטוי $|w \cdot x_i + b|$ משמעותו המרחק בין הנקודה x_i למישור המפריד, ולכן מה שהבעיה מנסה למצוא הוא את המשקלים שעבורם הנקודה הכי קרובה מסט האימון תהיה רחוקה ככל האפשר, תחת האילוץ שלא תהיה טעות בסיווג סט האימון.
ניתן לנסח את הבעיה הזו בצורה שקולה כך:

$$(w^*, b^*) = \arg \min_{w,b} \|w\|_2^2$$

$$s.t. \qquad y_j\big(w \cdot x_j + b\big) \geq 1 \; \forall j \in \{1, \dots, n\}$$

וזו בעיית אופטימיזציה ריבועית שניתן לפתור.