מבוא ללמידה חישובית (20942) | סיכום הרצאה 10

מנחה: ד"ר שי מימון
סמסטר: 2022א'
נכתב על ידי: מתן כהן

1 ולידציה - Validation

- ולידציה היא עוד דרך התמודדות עם התאמת יתר (overfitting)
- גם רגולריזציה וגם ולידציה הן שיטות אשר מנסות להביא למינימום את ה-E_{out} בנוסף להקטנת E_{in}

הגדרה. קבוצת המבחן - Test set
קבוצת המבחן (Test set) היא תת-קבוצה של $\mathcal{D}$ (סט הדוגמאות הכולל שלנו) **שלא נכללת בתהליך הלמידה** ומשמשת על אותנו בהערכת ההיפותזה הסופית שלנו $g \in \mathcal{H}$.
במילים אחרות - קבוצת המבחן משמשת אותנו להערכת ביצועי המודל.

הערה. תוצאות השגיאה E_{test} על קבוצת המבחן הן שיערוך **בלתי מוטה** (unbiased) ל- E_{out}

הגדרה. קבוצת ולידציה - Validation set

- קבוצת הולידציה היא תת-קבוצה של דוגמאות **שלא נכללת בתהליך הלמידה** ומשמשת לאמידת ה-$\boldsymbol{E_{out}}$
- אף על פי שקבוצת הולידציה לא נכללת בצורה ישירה בתהליך אימון המודל, היא כן עוזרת **בקבלת החלטות** בזמן תהליך הלמידה
- קבוצת הולידציה משמשת אותנו **בעת תהליך הלמידה** והיא **קטנה עד כדי** שהשערון שלה ל- E_{out} לא אופטימיסטי.

הערה. **ברגע** שקבוצת דוגמאות **משפיעה** בצורה מסוימת על תהליך הלמידה - היא לא חלק מ**קבוצת המבחן** ולכן קבוצת הולידציה היא לא חלק מקבוצת המבחן.

1.1 K מתוך N

בהינתן סט דומאות $\mathcal{D} = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ נחלק את הסט ל 2 קבוצות, $\mathcal{D}_{train}$ ו- $\mathcal{D}_{val}$ **בצורה שלא מבוססת על ערכי הדוגמאות** כך ש:

- בקבוצה $\mathcal{D}_{val}$ יש K דוגמאות
- בקבוצה $\mathcal{D}_{train}$ יש $N-K$ דוגמאות

כעת, החזאי האופטימלי שיתבסס על דוגמאות מתוך $\mathcal{D}_{train}$ נסמנו ב- g^- (כיוון שהוא לא כולל את כל הדוגמאות).
כמו-כן נסמן את השגיאה בין ערך החזאי האופטימלי של g^- על נקודה $\mathbf{x}_n$ לבין הערך האמיתי של הנקודה y_n בתור $e\left(g^-(\mathbf{x}_n), y_n\right)$ ונבחין כי **תוחלת השגיאה היא ה-** $\boldsymbol{E_{out}(g^-)}$:

$$\mathbb{E}_{\mathbf{x}_n}\left[e\left(g^-(\mathbf{x}_n), y_n\right)\right] = E_{out}(g^-) \tag{1.1}$$

על מנת לחשב את ה- E_{val} על g^- נחשב:

$$E_{val}(g^-) = \frac{1}{K} \sum_{\mathbf{x}_n \in \mathcal{D}_{val}} e\left(g^-(\mathbf{x}_n), y_n\right)$$

כעת, נרצה לחשב את התוחלת על השגיאה E_{val} על פי g^- כיוון שהאיבר $e\left(g^-(\mathbf{x}_n), y_n\right)$ הוא בעצם משתנה אקראי.
כמו-כן מההנחה שלנו שכלל הדוגמאות בלתי תלויות סטטיסטית נוכל להשתמש בתכונות התוחלת ובמה שהראינו ב- 1.1 ולקבל:

$$\begin{aligned}\mathbb{E}_{\mathcal{D}_{val}}\left[E_{val}(g^-)\right] &= \frac{1}{K}\sum_{\mathbf{x}_n\in\mathcal{D}_{val}}\mathbb{E}_{\mathcal{D}_{val}}\left[e\left(g^-(\mathbf{x}_n), y_n\right)\right] \\ &\overset{1.1}{=} \frac{1}{K}\sum_{\mathbf{x}_n\in\mathcal{D}_{val}} E_{out}(g^-)\end{aligned}$$

ולכן כיוון שסוכמים K קבועים (כי בקבוצה $\mathcal{D}_{val}$ ישנן K דוגמאות) נקבל:

$$\mathbb{E}_{\mathcal{D}_{val}}\left[E_{val}(g^-)\right] = E_{out}(g^-)$$

בצורה דומה נגדיר את **שונות השגיאה**:

$$\text{Var}_{\mathbf{x}}\left[e\left(g^-(\mathbf{x}_n), y_n\right)\right] = \sigma^2(g^-)$$

ומכך **שכלל הדוגמאות בלתי תלויות סטטיסטית** נוכל להשתמש בהגדרת השונות ובעובדה שלכל משתנה אקראי X וקבוע α:

$$\text{Var}(\alpha\cdot X) = \alpha^2\text{Var}(X)$$

ולכל שני משתנים אקראיים בת"ס (או **חסרי קורלציה**) X_1, X_2:

$$\text{Var}(X_1 + X_2) = \text{Var}(X_1) + \text{Var}(X_2)$$

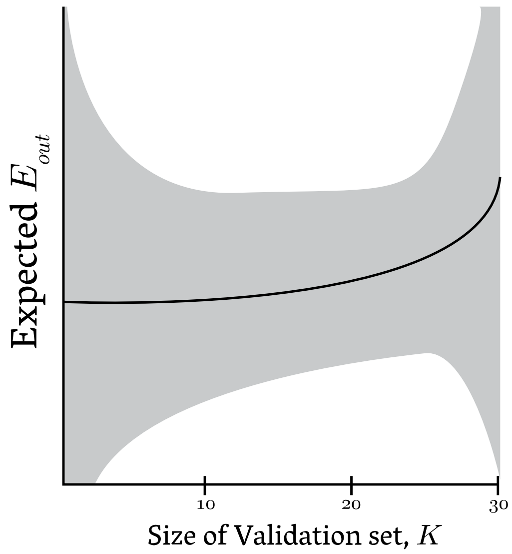
ובאופן דומה לחלוטין למה שנעשה מעלה:

$$\text{Var}_{\mathcal{D}_{val}}\left[E_{val}(g^-)\right] = \frac{1}{K^2}\sum_{\mathbf{x}_n\in\mathcal{D}_{val}}\text{Var}\left[e\left(g^-(\mathbf{x}_n), y_n\right)\right] = \frac{1}{K}\sigma^2$$

מכך נסיק:

במידה ונרצה שהשונות תהיה כמה שיותר קטנה (**ואנחנו רוצים**) הרי שהיא תלויה ב-K ולכן נסיק שככל שתהיינה לנו **יותר דגימות** בולידציה כך **הדיוק בשיערוך שלנו** E_{out} **ישתפר!**

דוגמה. בדומה לדוגמה מהשיעור הקודם, גם כאן נתבונן בהיפותזה מתוך $\mathcal{H}_2$ (פולינומים מדרגה 2), עבור $Q_f = 10$ (דרגת פולינום על פי נוצרו הדוגמאות), עבור $N = 40$ דוגמאות ודרגת רעש 0.4.



$$\mathcal{H}_2,\ \ Q_f = 10,\ \ N = 40,\ \ \text{noise} = 0.4$$

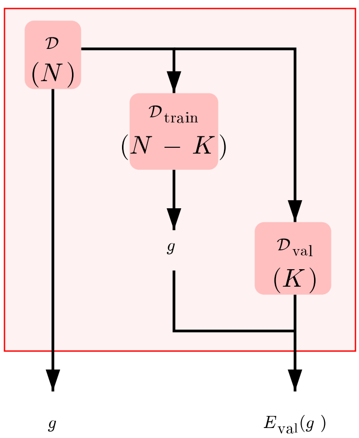
- על ציר ה-y נמצאת תוחלת $\mathbb{E}_{\mathcal{D}_{val}}\left[E_{val}(g^-)\right] = E_{out}(g^-)$
- על ציר ה-x נמצא את מספר הדוגמאות בסט הולידציה
- הקו השחור הוא אותו יחס בין מספר הדוגמאות בסט הולידציה לבין התוחלת
- השטח האפור מצייג את **השונות/סטיית התקן סביב הממוצע** - $\sqrt{\frac{\sigma^2}{K}}$

נקודות חשובות על סמך הדוגמה:

- בנקודה מסוימת נראה שתוחלת השגיאות עולה - זו כיוון שבנקודה זו K גדול מידי וסט האימון שכולל $N - K$ דוגמאות דליל מאוד - דבר שפוגע בתהליך האימון
- השונות (השטח האפור) הולכת וקטנה עד נקודה מסוימת כאשר מגדילים את K
  - הנקודה בה השונות חוזרת לגדול היא אותה נקודה בה השגיאה עולה - משמע הגענו למצב בו K גדול מידי ואין לנו מספיק דוגמאות להתאמן עליהן
- כלל אצבע הוא הקצאה של $K = \frac{N}{5}$ עבור סט הולידציה

מסקנה. K צריך להיות מצד אחד **גדול** כדי שהדיווח שלנו על טיב המודל יהיה אמין אך מצד שני הוא צריך להיות מספיק **קטן** על מנת שנוכל לקבל חזאי טוב.

בסופו של דבר נוכל לאמן את החזאי שלנו על סט מצומצם עם $N - K$ דוגמאות ולקבל חזאי g^- ובנוסף לאמן מודל נוסף על כל N הדוגמאות ולקבל חזאי g שבו נשתמש.



1.2 בחירת מודל - Model Selection

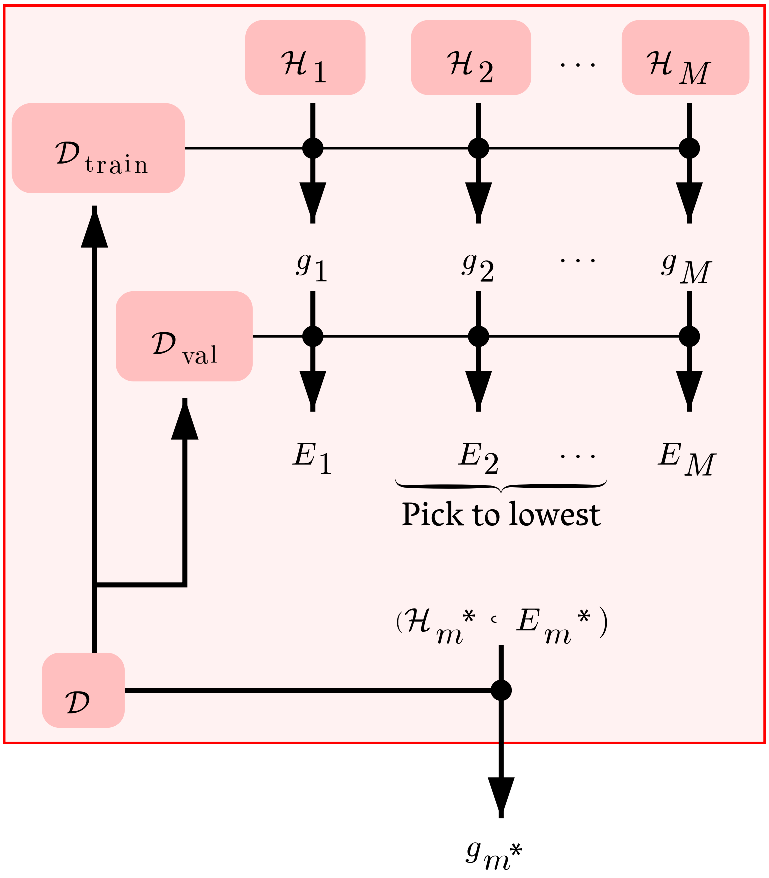
אחד השימושים הכי משמעותיים של הולידציה זה **להדריך אותנו** בעת הלמידה שלנו - וזהו הדבר המבדיל בין Test set ל-Validation set.

בפרט, הולידציה עוזרת לנו **לבחור מודל** מבין כלל המודלים שהתקבלו מתהליך הלמידה.

- בחירה בין מודל **לינארי** ל**ללא לינארי**
- בחירת **דרגת** פולינום במודל
- בחירת ערכים **לפרמטרי רגולריזציה**
- או כל פרמטר אחר **שמשפיע** על תהליך הלמידה שלנו

מסקנה. כל היפר-פרמטר שקיים בתהליך הלמידה ניתן לקביעה על פי הולידציה ובעזרת עובדה זו נוכל לאמן מודלים שונים עם היפר-פרמטרים שונים על מנת למצוא מודל אופטימלי.

נעשה זאת על ידי קביעת **סט דיסקרטי** של פרמטרים מראש (ניתן לבחור פילוג בצורה אחידה, לוגריתמית וכו') ונאמן חזאים שונים על פי אותם פרמטרים. לאחר מכן נוכל לבחור את המודל שנתן את הביצועים הכי טובים ביחד עם הפרמטרים שלו.



הסבר:

(1) מכלל המודלים וההיפותזות $\mathcal{H}_1, \ldots, \mathcal{H}_M$ נקבל חזאי אופטימלי מתאים $g_1^-, \ldots, g_M^-$ על פי סט האימון $\mathcal{D}_{train}$

(2) נבחן את ה- E_{out} על סט הולידציה $\mathcal{D}_{val}$

(3) נבחר את המינימלי מבין $E_1, \ldots, E_M$ בתקווה שייצג את הביצועים הכי טובים על Out-of-sample data

(4) על פי אותו E_m שבחרנו נקבל היפר-פרמטרים אופטימליים ונפעיל את אותו חזאי עם הפרמטרים על $\mathcal{D}$ (כל הדוגמאות)

(5) נקבל g_m^* שהוא החזאי הרצוי

1.3 Cross Validation

כפי שאמרנו, ככל ש- K קטן , השגיאה של g מתקרבת אל השגיאה של g^- כיוון שאנחנו מאמנים על "כמעט" אותו מספר של נקודות.

כמו-כן, ככל ש- K גדל, השגיאה של g^- מתקרבת לשגיאה על סט הולידציה של g^- ובאופן יותר ברור:

$$\boxed{E_{out}\left(g\right)\underbrace{\approx}_{\text{small K}}E_{out}\left(g^{-}\right)\underbrace{\approx}_{\text{large K}}E_{val}\left(g^{-}\right)}$$

אז מצד אחד אנחנו רוצים K **קטן** אך מצד שני אנחנו רוצים K **גדול**, מה עושים?!

1.3.1 Leave one out cross validation

בשיטה זו נפתור את הדילמה לעיל.

אלגוריתם Leave one out

(1) לכל n מ-1 ועד N:

(א) הגדר את סט האימון:

$$\mathcal{D}_n=\left(\mathbf{x}_1,y_n\right),\ldots,\left(\mathbf{x}_{n-1},y_{n-1}\right),\cancel{\left(\mathbf{x}_n,y_n\right)},\left(\mathbf{x}_{n+1},y_{n+1}\right),\ldots,\left(\mathbf{x}_N,y_N\right)$$

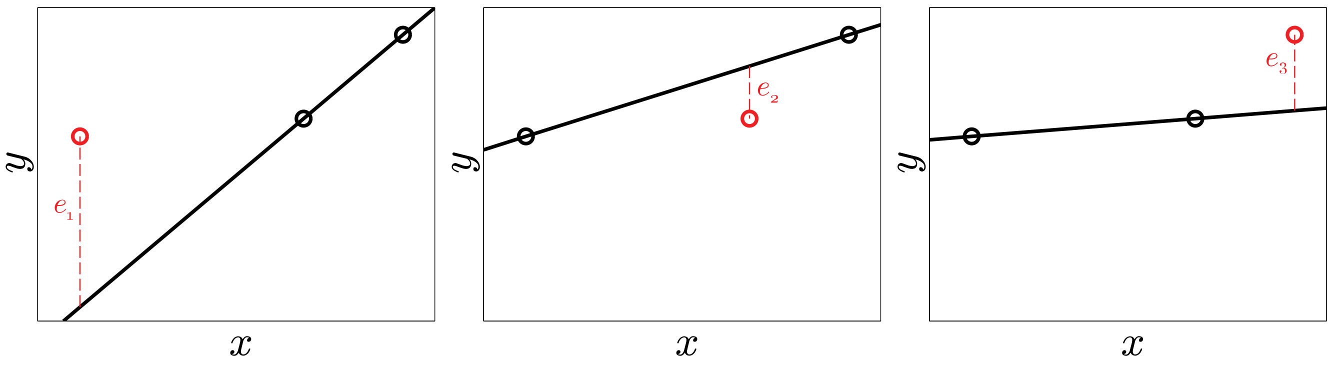
(ב) סמן את ההיפותזה הסופית שנלמדה מ- $\mathcal{D}_n$ ב- g_n^-

(ג) סמן את השגיאה של ההיפותזה שהתקבלה על ידי $E_{val}\left(g_n^-\right)=e\left(g_n^-\left(\mathbf{x}_n\right),y_n\right)$ ב- e_n

(2) הגדר את שגיאת ה - Cross validation error ב:

$$E_{cv}=\frac{1}{N}\sum_{n=1}^{N}e_n$$

דוגמה. ל- Leave one out Cross Validation



$$E_{cv}=\frac{1}{3}\left(e_1+e_2+e_3\right)$$

הסבר:

ישנן 3 נקודות, לכן ביצענו 3 איטרציות כאשר בכל אחת מהאיטרציות השארנו נקודה אחת בחוץ לבדיקת השגיאה. לבסוף מצאנו את E_{cv} בתור ממוצע השגיאות.

1.3.2 K-fold cross validation

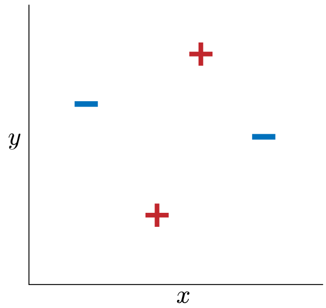
זוהי דרך נוספת להתמודדות עם הבעיה, היא זהה ל- Leave one out מבחינת המתודולוגיה אך כאן נרצה להשאיר קבוצה יותר גדולה של דוגמאות בחוץ בגודל K, ישנם $\frac{N}{K}$ מודלים שנאמן בצורה זו.

2 רשתות נוירונים - Neural Networks

- הכללה של הפרספטרון
- יכולות למדל עם פונקציות מטרה מורכבות()

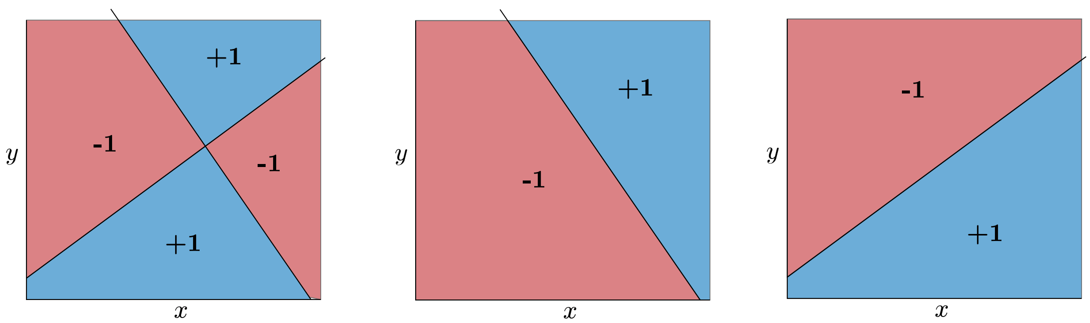
2.1 Multi-layer Perceptron

ניזכר כי מודל הפרספטרון לא הצליח להתמודד עם בעית ה-XOR שכן אף קו לינארי לא יכול להפריד את הדוגמה הבאה:



אך אם נוכל להשתמש ב**שני חזאים? האם נוכל לפתור את הבעיה?**

התשובה היא - כן, אם נגדיר שאם שני החזאים מסכימים על הסימן שלהם, נחזיר -1 ואם לא מסכימים נחזיר $+1$.



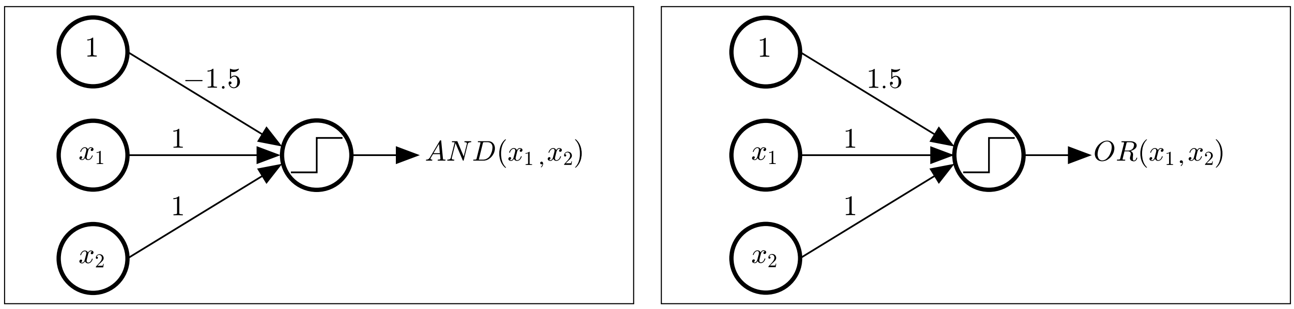
$h_1(\mathbf{x})=sign(\mathbf{w}_1^T\mathbf{x})$ $h_2(\mathbf{x})=sign(\mathbf{w}_2^T\mathbf{x})$

נבחין כי זו פעולת XOR ולכן נרצה להביע את הפעולה בעזרת:

$$h = \bar{h_1} \cdot h_2 + h_1\bar{h_2}$$

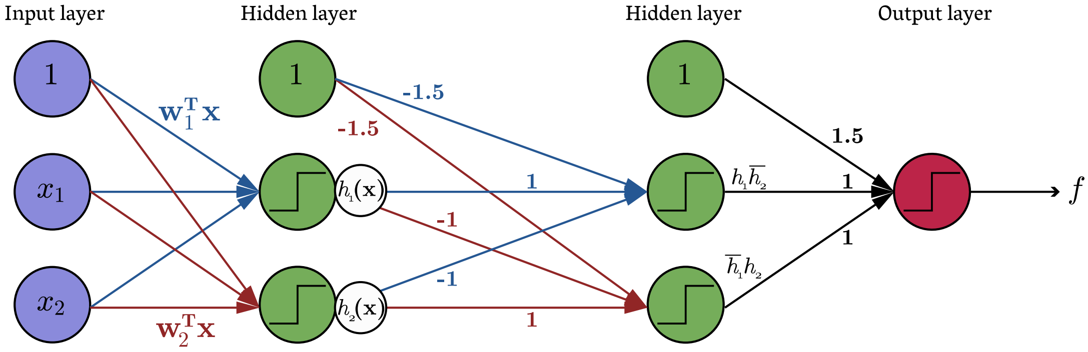
ולכן נרצה לממש שערי AND, OR ו- NOT.

2.1.1 מימוש שערי AND, OR ו- NOT:



על מנת לממש שער NOT כל שצריך הוא להפוך את סימני המשקולות של הקלט.

כעת נוכל לייצר את שער ה-XOR בעזרת שערי AND, OR והפיכת סימני המשקולות (NOT):



2.2 Sigmoidal Neural Networks

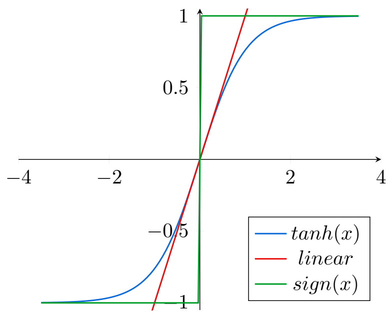
ראינו כי השימוש בפונקצית $sign$ הוא בעייתי כיוון שזוהי אינה פונקציה "חלקה" מה שגורם לבעיה הקומבינטורית להיות אפילו קשה יותר.
היינו רוצים פונקציה חלקה וגזירה שדומה לפונקצית ה- $sign$ שתאפשר לנו להשתמש בכלים אנליטיים כמו גרדיאנט על מנת למצוא משקלים אופטימליים.

דוגמה. נוכל להשתמש ב tanh:

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$\lim_{x\to\infty} \tanh = 1, \quad \lim_{x\to-\infty} \tanh = -1$$

ואז בעזרת המקדם של x נוכל לשלוט על מהירות ההתכנסות של tanh ל $+1$ ו- -1.
באופן כללי נוכל לראות את השוני בין הפונקציות:



בעזרת שימוש בפונקציות הללו נוכל לבנות רשתות נוירונים.

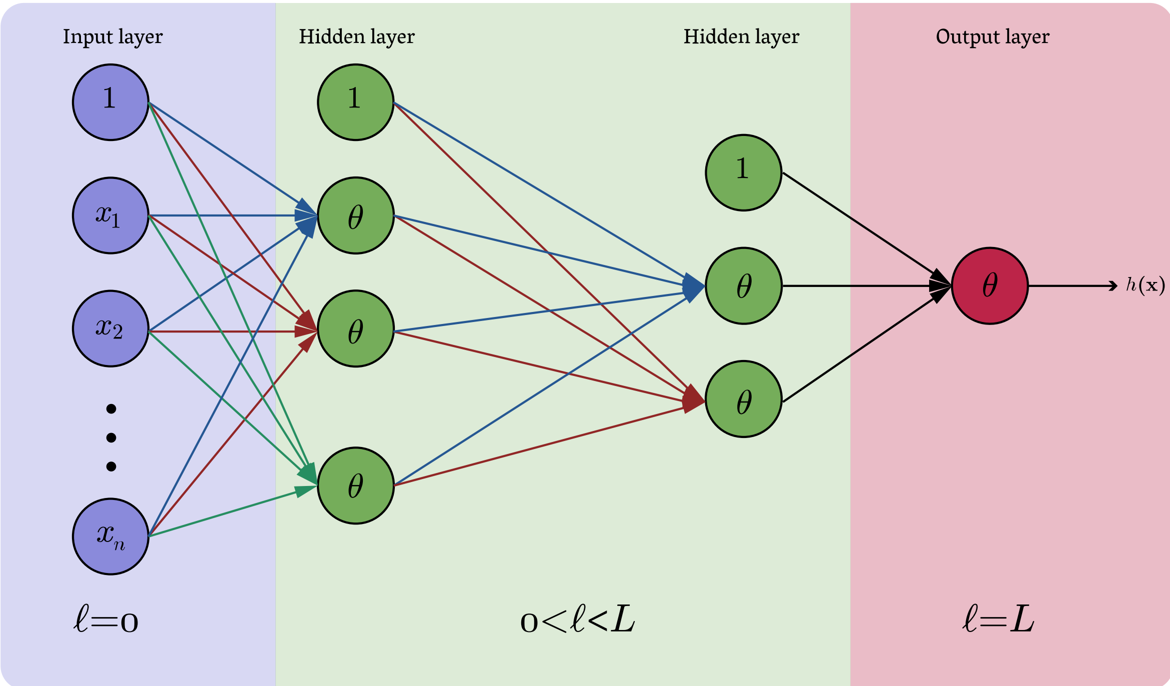
2.3 Neural Networks - 'softened' MLP

2.3.1 ארכיטקטורת הרשת:

- Input layer - שכבת הכניסה (קלט): השכבה הראשונה ברשת, **לא נספרת מבין השכבות** - $\ell = 0$
- Hidden layers - שכבות חבויות: כלל השכבות שנמצאות בין שכבת הכניסה לשכבת המוצא - $0 < \ell < L$
- Output layer - שכבת המוצא: ממנה יוצא החזאי - $\ell = L$
- בנוסף לפי'צרים מוסיפים את האיבר **ההטיה** (1), אין לו כניסות והוא מהווה כניסה לפונקצית האקטיבציה שלנו

2.3.2 שיטת הפעולה

- כל node בכל שכבה מוציא ערך
- כל קשת בין node ל- node מכילה משקל שמוכפל בערך היוצא מ-node המקור אל יעדו
- ערך הכניסה ל-node הוא חיבור תוצאות המכפלה שנוצרה במשקולות
- מפעילים את פונקצית האקטיבציה (כמו tanh) על התוצאה שהתקבלה מהחיבור
- נקבל במוצא פונקציה (לרוב לא לינארית) מורכבת של x ושל w



2.3.3 שימוש ברשתות נוירונים לבעיות למידה שונות

נוכל להשתמש ברשתות נוירונים עבור בעיות למידה שונות

- **רגרסיה** - נחליף את θ בשכבת המוצא בפונקצית הזהות
  - **הסתברות/רגרסיה לוגיסטית** - נחליף את θ בשכבת המוצא ב $sigmoid$
- **סיווג** - נחליף את הסיגמויד θ בשכבת המוצא בפונקצית $sign$

2.3.4 אופטימיזצית הרשת

ברגע שקבענו את הארכיטקטורה (מספר השכבות ומספר ה-nodes בכל שכבה), נרצה לחפש פרמטרים אופטימליים שיתאימו את המודל לנתונים בצורה הטובה ביותר.

- **רגרסיה** - שימוש בשגיאה ריבועית: $e(\mathbf{x}) = (h(\mathbf{x}) - y)^2$ + מיצוע ביחס לכלל הדוגמאות
- **סיווג** - שימוש ב cross entropy (ידובר בהמשך) או במדדי שגיאה אחרים

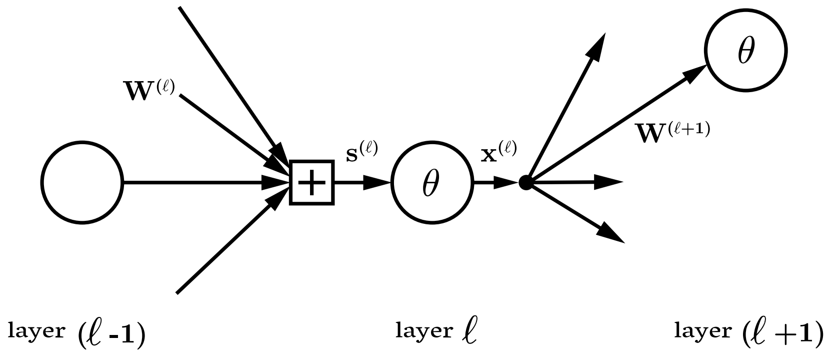
2.4 ארכיטקטורת הרשת

כפי שנאמר ארכיטקטורת הרשת מוגדרת על ידי:

- מספר השכבות L
- ממד כל שכבה (מספר הנוירונים/nodes)

ההיפותזה מוגדרת על ידי בחירה של משקלים ברשת - משמע **כל** בחירה של משקלים נותנת **היפותזה אחרת**.

2.4.1 סימונים



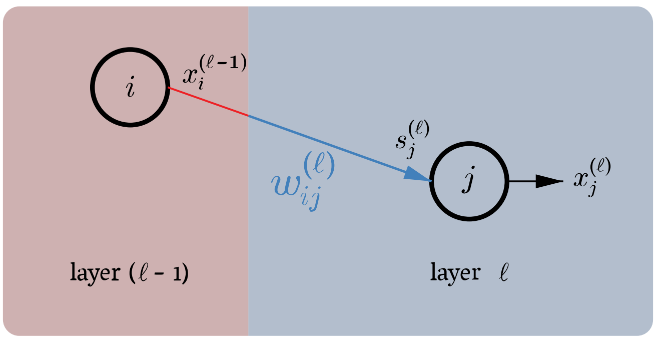
ממד	סימון	שם	משמעות
$d^{(\ell)}$	$\mathbf{s}^{(\ell)}$	signals in	כניסה לאקטיבציה בשכבה ℓ (לאחר סכימה)
$d^{(\ell+1)}$	$\mathbf{x}^{(\ell)}$	outputs	יציאה מפונקצית האקטיבציה בשכבה ℓ
$\left(d^{(\ell-1)}+1\right) \times d^{(\ell)}$	$W^{(\ell)}$	weights in	מטריצה של משקולות הנכנסות לשכבה ℓ
$\left(d^{(\ell)}+1\right) \times d^{(\ell+1)}$	$W^{(\ell+1)}$	weights out	מטריצה של משקולות שיוצאות משכבה ℓ

- אין ערכים שנכנסים לשכבת הכניסה ולכן אין $W^{(0)}$ או $\mathbf{s}^{(0)}$
- אברי המטריצה $W^{(\ell)}$ הם מהצורה: $W_{ij}^{(\ell)}$ כך ש:

כל שורה i שייכת ל-node בשכבה $\ell - 1$ וכל עמודה j שייכת ל-node בשכבה ℓ

- התוספת של $+1$ לממדים היא כתוצאה מהוספת ההטיה (bias)

zoom-in למעבר בין נוירונים i ו- j:



- מ-i יוצא המוצא $x_i^{(\ell-1)}$
- הקשת בין i ל-j בעלת המשקולה $w_{ij}^{(\ell)}$
- המכפלה בין $w_{ij}^{(\ell)}$ ל- $x_i^{(\ell-1)}$ מסומנת ב- $s_j^{(\ell)}$

2.5 Forward Propagation

כפי שהסברנו עד כה, אנחנו "מעבירים קדימה" את המכפלות שלנו בצורה הבאה:

- בכל שכבה מכפילים את $\mathbf{x}^{(\ell)}$ ב- $W^{(\ell+1)}$ ומקבלים $\mathbf{s}^{(\ell+1)}$:

$$\mathbf{x} = \mathbf{x}^{(0)} \xrightarrow{W^{(1)}} \mathbf{s}^{(1)} \xrightarrow{\theta} \mathbf{x}^{(1)} \xrightarrow{W^{(2)}} \mathbf{s}^{(2)} \xrightarrow{\theta} \mathbf{x}^{(2)} \ldots \to \mathbf{s}^{(L)} \xrightarrow{\theta} \mathbf{x}^{(L)} = h(\mathbf{x})$$

- האלגוריתם:

אלגוריתם Forward Propagation

---

(1) **התחל** את $\mathbf{x}^{(0)}$ בערך $\mathbf{x}$

(2) **לכל** $\ell = 1$ ועד L **בצע**:

(א) הגדר את $\mathbf{s}^{(\ell)}$ בתור:

$$\mathbf{s}^{(\ell)} \leftarrow \left(W^{(\ell)}\right)^T \mathbf{x}^{(\ell-1)}$$

(ב) הגדר את $\mathbf{x}^{(\ell)}$ בתור:

$$\mathbf{x}^{(\ell)} \leftarrow \begin{bmatrix} 1 \\ \theta\left(\mathbf{s}^{(\ell)}\right) \end{bmatrix}$$

(3) **החזר** $\mathbf{x}^{(L)} = h(\mathbf{x})$

- מדוע $\mathbf{s}^{(\ell)}$ מוגדר כך?
  - ⋆ כיוון שאנחנו מחשבים את הסכום הממושקל שכולל **פלטים** מהשכבה הקודמת **ומשקולות** בין השכבה הקודמת לנוכחית

$$s_j^{(\ell)} = \sum_{i=0}^{d^{(\ell-1)}} w_{ij}^{(\ell)} x_i^{(\ell-1)}$$

  - · זוהי בעצם מכפלה סקלרית בין העמודה ה-j-ית של W לבין הוקטור $\mathbf{x}$!
  - ⋆ לפיכך נוכל פשוט לכתוב בכתיב וקטורי:

$$s^{(\ell)} = \left(W^{(\ell)}\right)^T \mathbf{x}^{(\ell-1)}$$

- לבסוף נקבל את ההיפותזה h ונבדוק את השגיאה:

$$\begin{aligned} E_{in}(\mathbf{w}) &= \frac{1}{N}\sum_{n=1}^{N}\left(h(\mathbf{x}_n;\mathbf{w}) - y_n\right)^2 \\ &\stackrel{3}{=} \frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{x}_n^{(L)} - y_n\right)^2 \end{aligned}$$

- נבחין כי נצטרך לאפטם את השגיאה על פי $\mathbf{w}$ וזוהי משימה לא פשוטה, לשם כך נשתמש ב Back propagation

2.6 Backpropagation Algorithm

אלגוריתם ה- Backpropagation עוזר לנו לאפטם את פונקצית השגיאה שראינו ממקודם.

כיוון ש $h(\mathbf{x})$ היא פונקציה "חלקה" (גזירה), נוכל להפעיל **Gradient Descent** על פונקצית השגיאה.

נתחיל במוטיב:

דוגמה. The sigmoidal perceptron

נתון פרספטרון שבמקום פונקצית ה- $sign$ נתונה פונקצית ה-tanh:

- $h(\mathbf{x}) = \tanh(\mathbf{w}^T\mathbf{x})$
- $E_{in}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\left(\tanh\left(\mathbf{w}^T\mathbf{x}\right) - y_n\right)^2$

נבדוק את הגרדיאנט:

- תחילה נבדוק את נגזרת tanh ונראה כי ישנו פתרון סגור:

$$\frac{\partial \tanh}{\partial s} = \frac{\partial}{\partial s}\frac{\sinh(s)}{\cosh(s)} = \frac{\cosh^2(s) - \sinh^2(s)}{\cosh^2(s)} = \boxed{1 - \tanh^2(s)}$$

$$\nabla E_{in}(\mathbf{w}) = 2\cdot\frac{1}{N}\sum_{n=1}^{N}\left(\tanh\left(\mathbf{w}^T\mathbf{x}\right) - yn\right)\cdot\left(1 - \tanh^2\left(\mathbf{w}^T\mathbf{x}\right)\right)\cdot\mathbf{x}_n$$

- במידה ונשתמש ב MLP (רשת נוירונים) לא יהיה פתרון פשוט וסגור, לשם כך נשתמש באלגוריתם ה- Backpropagation
- אלגוריתם זה משתמש בשיטת ה Gradient Descent שנמצאת בסיכום שיעור 4, אך להלן חזרה קצרה:
  - זוהי טכניקה למציאת מינימום של פונקציה גזירה
  - בהתחשב במשקולות ההתחלתיות, הנתיב שהאלגוריתם יניב יביא אותנו למינימום מקומי
  - נרצה תמיד לקחת צעד כנגד כיוון הרדיאנט
  - נהוג להתחל עם משקולות קטנות בצורה אקראית
  - נהוג לסיים על פי:
    - ⋆ מספר צעדים/איטרציות
    - ⋆ בחינת הנורמה והשגיאה
    - ⋆ חסם עליון לשגיאה (נרצה שיגיע מתחת לערך מסוים)
  - כיוון שיש מס' נקודות אופטימום, נהוג להתחיל ממספר נקודות שונות ואז לבחור את הנקודה שהניבה את הביצועים הטובים ביותר

אלגוריתם Gradient Descent

(1) **התחל** את המשקולות של צעד $t = 0$ ל $\mathbf{w}(0)$, ואת גודל הצעד η

(2) **לכל** $t = 0, 1, 2, ...$ **בצע**:

(א) **חשב** את הגרדיאנט: $\mathbf{g}_t = \nabla E_{in}(\mathbf{w}(t))$

(ב) **קבע** את כיוון התזוזה: $\mathbf{v}_t = -\mathbf{g}_t$

(ג) **עדכן** את המשקולות: $\mathbf{w}(t+1) = \mathbf{w}(t) + \eta\mathbf{v}_t$

(ד) **המשך** לצעד הבא אם לא הגענו לתנאי עצירה

(3) **החזר** את סט המשקולות הסופי

מה שנרצה לעשות באלגוריתם ה-Backpropagation הוא דבר פשוט:

- נרצה להסתכל על מיצוע השגיאות $E_{in}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} e_n$
- ולמצוא מינימום בהתאם למטריצת המשקולות:

$$\frac{\partial E_{in}}{\partial W^{(\ell)}} = \frac{1}{N}\sum_{n=1}^{N}\frac{\partial e_n}{\partial W^{(\ell)}}$$

- האלגוריתם מתבסס על שימוש חוזר **בכלל השרשרת** במהלך מעבר מסוף רשת הנוירונים להתחלתה.