מבוא למערכות לומדות - הרצאה 8 - Boosting

29 ביוני 2015

"לומד חלש" הוא אלגוריתם למידה נאיבי הלומד היפותזות מאד פשוטות (מעין כללי אצבע). ההרצאה היום תעסוק באלגוריתמים אשר לוקחים לומד חדש, ובונים ממנו "לומד חזק". כלומר, אלגוריתם למידה המסוגל ללמוד היפותזות מורכבות. אלגוריתמים (לומדים חזקים) המתאימים לתיאור הנ"ל נקראים **אלגוריתמי האצה (Boosting)**.

ההרצאה כולה תעסוק בקלסיפיקציה בינארית. כלומר, נניח ש־$Y = \{\pm 1\}$ ונרצה למזער את $L_{\mathcal{D}}^{0-1}$.

1 לומדים חלשים

לפני שנציג את האלגוריתמים, נביט במספר דוגמאות לאלגוריתמים המתאימים לתיאור של לומד חלש.

סיפים על הישר

כאן $X = \mathbb{R}$. **סף על הישר** הוא היפותזה מהצורה

$$h_\theta(x) \mapsto \mathrm{sign}(x - \theta)$$

עבור $\theta \in \mathbb{R}$. דוגמא ללומד חלש הוא אלגוריתם הממשש את כלל ה־ERM ביחס למחלקה $\mathcal{H} = \{h_\theta \mid \theta \in \mathbb{R}\}$. נעיר שבתרגול 3 ראיתם שניתן לעשות זאת ביעילות.

גדמי החלטה (Decision Stumps)

גדמי החלטה מכלילים סיפים על הישר. כאן, $X = \mathbb{R}^n$. **גדם החלטה** הוא היפותזה מהצורה

$$h_{\theta,i}(x) \mapsto \mathrm{sign}(x_i - \theta)$$

עבור $\theta \in \mathbb{R}$ ו־$i \in [n]$. דוגמא ללומד חלש הוא אלגוריתם הממשש את כלל ה־ERM ביחס למחלקה $\mathcal{H} = \{h_{\theta,i} \mid \theta \in \mathbb{R}, i \in [n]\}$. בדומה לסיפים על הישר גם עבור גדמי החלטה ניתן לממש את כלל ה־ERM ביעילות.

באופן כללי, ניתן לקחת כל אלגוריתם למידה שלמדנו, ולהפעיל אותו על תת קבוצה קטנה של הקואורדינטות, או לחלופין, לדרוש ממנו להחזיר היפותזה מאד פשוטה (עץ החלטה עם מספר קטן של עלים, וקטור מנורמה קטנה, ...)

2 יערות אקראיים (Random Forests)

הדרך אולי הנאיבית ביותר לבנות לומד חזק מלומד חלש, היא פשוט להריץ את הלומד החלש בכמה אופנים שונים, לקבל היפותזות

$$h_1, \ldots, h_k : X \to \{\pm 1\}$$

ולהחזיר פשוט את הכרעת הרוב. כלומר, את ההיפותזה

$$h(x) = \mathrm{Majority}(h_1(x), \ldots, h_k(x))$$

אלגוריתם פופולארי הממש את הרעיון הנ"ל הוא אלגוריתם היערות האקראיים:

יערות אקראיים

פרמטרים: אלגוריתם ("חלש") W הלומד עצים, מספר עצים k, מספר קורדינטות d
קלט: $S \in (\{\pm 1\}^n \times \{\pm 1\})^m$
1. עבור $i = 1, 2 \ldots, k$:
1.1. בחר באקראי תת קבוצה $C \subset [n]$ בגודל d
1.2 הרץ את W ביחס לקואורדינטות ב־C, וקבל היפותזה h_i
2. החזר את ההיפותזה $h(x) = \mathrm{Majority}(h_1(x), \ldots, h_k(x))$

3 האצה אדפטיבית (AdaBoost)

אלגוריתם ה־AdaBoost משתמש בלומד החלש על תתי מדגמים של המדגם הנתון

$$S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \subset X \times \{\pm 1\}$$

כלומר, הלומד החלש מקבל כל פעם תת־מדגם $S' \subset S$, ועליו למצוא היפותזה עם ביצועים לא טריוויאלים (שגיאה קטנה מחצי) על S'. AdaBoost משקלל את ההיפותזות שהחזיר הלומד החלש, כך שבמידה והלומד החלש הצליח במשימתו, להיפותזה המשוקללת תהיה שגיאה קטנה.

יהיה נוח יותר לעבוד עם **משקולים** של המדגם במקום תתי מדגמים. AdaBoost יחזיק משקל $D_i \geq 0$ לכל דוגמא. המשקלות הללו יגדירו התפלגות

$$\left(\bar{D}_1, \ldots, \bar{D}_m\right) := \frac{1}{\sum_{i=1}^n D_i} \left(D_1, \ldots, D_m\right)$$

על הדוגמאות. בכל שלב הלומד החלש ידרש למצוא היפותזה $h : X \to \{\pm 1\}$ הצודקת על רוב הדוגמאות ביחס להתפלגות הנ"ל. כלומר, היפותזה עם שגיאת 1־0 ממושקלת

$$L_{S,D}^{0-1}(h) := \sum_{i=1}^{m} \bar{D}_i l_{0-1}(h(x_i), y_i)$$

לא טריוויאלית. נעיר שאת רב אלגוריתמי הלמידה פשוט מאד להרחיב לאלגוריתמים העובדים גם עם מדגמים ממושקלים. שכן, רוב האלגוריתמים ממזערים, בין אם בצורה יוריסטית ומקומית, ובין אם בצורה ריגורוזית את $L_S(h)$ עבור איזשהו לוס. לכן, ניתן להכליל אותם למדגמים ממושקלים, ע"י מעבר מ־$L_S(h)$ ל־$L_{S,D}(h)$.

בכל שלב AdaBoost יפעיל את הלומד החלש ביחס למשקלות הנוכחיים. במהלך הריצה, AdaBoost ירכז את המשקלות ב־"דוגמאות הקשות" - אלו עליהן "רוב" ההיפותזות שהוחזרו ע"י הלומד החלש נכשלות.

קונקרטית, נסמן ב־

$$D^{(t)} = (D_1^{(t)}, \ldots, D_m^{(t)})$$

את המשקלות בתחילת השלב ה־t. בשלב הראשון המשקולות יהיו אחידים. כלומר יתקיים $\forall i, \quad D_i^{(1)} = 1$. בשלב ה־$t$, AdaBoost יריץ את הלומד החלש ביחס למשקלות $D^{(t)}$ ויקבל היפותזה $h_t : X \to \{\pm 1\}$. נסמן ב־

$$\epsilon_t := L_{S,D^{(t)}}^{0-1}(h_t)$$

את השגיאה הממושקלת של h_t. AdaBoost יגדיר מקדם $w_t = \frac{1}{2}\log\left(\frac{1}{\epsilon_t} - 1\right)$ ל־h_t. נשים לב ש־w_t נע בין 0 (עבור $\epsilon_t = \frac{1}{2}$) ל־∞, ויגדל ככל ש־ϵ_t יקטן. המקדם הנ"ל יקבע כמה חשיבות ייחס האלגוריתם ל־h_t. החשיבות הנ"ל תבוא לידי ביטוי בדומיננטיות של h_t בהיפותזה הסופית, ובגודל השינוי במעבר מ־$D^{(t)}$ ל־$D^{(t+1)}$. קונקרטית, בסוף השלב ה־t AdaBoost יעדכן את המשקלות כך שמשקל הדוגמאות הדוגמאות עליהן h_t שגה יגדל פי e^{w_t}, בעוד המשקל על הדוגמאות עליהן h_t צדק יקטן פי e^{w_t}. בסוף הריצה, לאחר T שלבים, תוחזר ההיפותזה

$$H_T = \sum_{t=1}^{T} w_t h_t$$

נסכם:

AdaBoost

פרמטרים: אלגוריתם ("חלש") W
קלט: $S \in (X \times \{\pm 1\})^m$, מספר איטרציות T
1. אתחל $D^{(1)} = (1, \ldots, 1)$
2. עבור $t = 1, 2 \ldots, T$
2.1. הרץ את W ביחס למשקלות $D^{(t)}$ וקבל היפותזה $h_t : X \to \{\pm 1\}$
2.2. הגדר $\epsilon_t := L^{0-1}_{S,D}(h)$
2.3. הגדר $w_t = \frac{1}{2} \log\left(\frac{1}{\epsilon_t} - 1\right)$
2.4 עדכן $D^{(t+1)}_i = D^{(t)}_i e^{-w_t h_t(x_i) y_i}$
3. החזר את ההיפותזה $H_T = \sum_{t=1}^{T} w_t h_t$

3.1 השגיאת האמפירית

נניח שהאלגוריתם החלש הצליח במשימתו, והשגיאות ϵ_t היו כולן קטנות מ־$\frac{1}{2}$. מה תהיה השגיאה האמפירית של ההיפותזה הסופית h? המשפט הבא חוסם את $L^{0-1}_S(h)$ ביחס לשגיאות ϵ_t.

משפט 3.1 נניח שלכל t, $\epsilon_t = \frac{1}{2} - \gamma_t$. אזי,

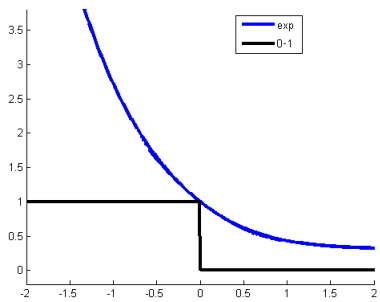
$$L^{0-1}_S(h) \leq \exp\left(-2 \sum_{t=1}^{T} \gamma_t^2\right)$$

למשל, אם בכל שלב השגיאה הייתה קטנה מ־0.4, נקבל ש־$L^{0-1}_S(h) \leq \exp(-0.02T)$. כלומר, השגיאה קטנה במהירות אקספוננציאלית.

ניגש להוכחת המשפט. ההוכחה מראה שבכל שלב t, AdaBoost מקטין פי $e^{-2\gamma_t^2}$ את השגיאה של תחליף (קמור) מסויים ל־l_{0-1}. נגדיר את **ההפסד המעריכי (Exponential Loss)** להיות הפונקציה $l_{\exp} : \mathbb{R} \times \{\pm 1\} \to \mathbb{R}_+$ הבאה:

$$l_{\exp}(\hat{y}, y) = \exp(-\hat{y} y)$$

נשים לב שבדומה ל־l_{0-1}, $l_{\exp}$ הוא מהצורה $f(\hat{y}y)$ עבור $f : \mathbb{R} \to \mathbb{R}_+$. בציור הבא מופיעות הפונקציות f המתאימות לשני ההפסדים:



הוכחת הלמה הבאה פשוטה ומושארת כתרגיל

למה 3.2 לכל $h : X \to \mathbb{R}$, $L_S^{0-1}(h) \leq L_S^{\exp}(h)$

לב הוכחת המשפט הוא הלמה הבאה:

למה 3.3 נסמן $H_t = \sum_{t'=1}^{t} w_{t'} h_{t'}$ (עבור $t = 0$, $H_t := 0$). אזי לכל $t \geq 1$ מתקיים

$$L_S^{\exp}(H_t) \leq \exp(-2\gamma_t^2) L_S^{\exp}(H_{t-1})$$

הוכחה: ראשית נשים לב שמתקיים

$$\begin{aligned}
D_i^{(t)} &= e^{-w_{t-1}h_{t-1}(x_i)y_i} D_i^{(t-1)} \\
&= e^{-w_{t-1}h_{t-1}(x_i)y_i} e^{-w_{t-2}h_{t-2}(x_i)y_i} D_i^{(t-2)} \\
&= e^{-[w_{t-1}h_{t-1}(x_i)y_i + w_{t-2}h_{t-2}(x_i)y_i]} D_i^{(t-2)} \\
&\vdots \\
&= e^{-[w_{t-1}h_{t-1}(x_i)y_i + w_{t-2}h_{t-2}(x_i)y_i + \ldots + w_1 h_1(x_i)y_i]} D_i^{(1)} = e^{-H_{t-1}(x_i)y_i}
\end{aligned}$$

כעת, מתקיים

$$\begin{aligned}
\frac{L_S^{\exp}(H_t)}{L_S^{\exp}(H_{t-1})} &= \frac{\sum_{i=1}^m \exp(-H_t(x_i)y_i)}{\sum_{j=1}^m \exp(-H_{t-1}(x_j)y_j)} \\
&= \sum_{i=1}^m \frac{\exp(-H_{t-1}(x_i)y_i)}{\sum_{j=1}^m \exp(-H_{t-1}(x_j)y_j)} \exp(-w_t h_t(x_i)y_i) \\
&= \frac{1}{\sum_{i=1}^m D_i^{(t)}} \sum_{i=1}^m D_i^{(t)} \exp(-w_t h_t(x_i)y_i)
\end{aligned}$$

נשים לב שהביטוי האחרון הוא $L_{S,D^{(t)}}^{\exp}(w_t h_t)$. מהי, אם כן, השגיאה של $w_t h_t$ ביחס ל־$l^{\exp}$? ובכן, מכיוון ש־$L_{S,D^{(t)}}^{0-1}(h_t) = \epsilon_t$, הדוגמאות עבורן $h(x_i)y_i = -1$ מהוות ϵ_t

אחוז (לפי $D^{(t)}$) מכלל הדוגמאות. הדוגמאות הללו תורמות $\epsilon_t e^{-h(x_i)y_i w_t} = \epsilon_t e^{w_t}$ ל־ $L^{\exp}_{S,D^{(t)}}(w_t h_t)$. יתר הדוגמאות מהוות $1-\epsilon_t$ אחוז מכלל הדוגמאות, ועליהן $h(x_i)y_i = 1$, ולכן הן תורמות $(1-\epsilon_t)e^{-w_t}$ ל־$L^{\exp}_{S,D^{(t)}}(w_t h_t)$. מכאן,

$$\begin{aligned}
L^{\exp}_{S,D^{(t)}}(w_t h_t) &= (1-\epsilon_t)e^{-w_t} + \epsilon_t e^{w_t} \\
&= \sqrt{\frac{\epsilon_t}{1-\epsilon_t}}(1-\epsilon_t) + \sqrt{\frac{1-\epsilon_t}{\epsilon_t}}\epsilon_t \\
&= 2\sqrt{\epsilon_t(1-\epsilon_t)} \\
&= 2\sqrt{\left(\frac{1}{2}-\gamma_t\right)\left(\frac{1}{2}+\gamma_t\right)} = \sqrt{(1-2\gamma_t)(1+2\gamma_t)} \\
&= \sqrt{(1-4\gamma_t^2)} \le \exp\left(-2\gamma_t^2\right)
\end{aligned}$$

כאשר אי השיוויון האחורן נובע מכך ש־$\forall x \in \mathbb{R},\ 1+x \le e^x$ ■

הוכחה: (של משפט 3.1) משתי הלמות לעיל נובע ש־

$$L^{0-1}_S(H_T) \le L^{\exp}_S(H_T) \le \exp\left(-2\sum_{t=1}^{T}\gamma_t^2\right) L^{\exp}_S(H_0) = \exp\left(-2\sum_{t=1}^{T}\gamma_t^2\right)$$

■

3.2 שגיאת ההכללה

בפסקה הוקודמת הראינו שכאשר מריצים AdaBoost והלומד החלש מצליח להחזיר היפותזות לא טריוויאליות, השגיאה האמפירית היא נמוכה. כמובן, העובדה שהשגיאה האמפירית הינה קטנה לא מבטיחה לנו דבר על השגיאה אמיתית - יכול להיות שהאלגוריתם עשה אוברפיט! המשפט הבא מראה שכאשר הלומד החלש מחזיר היפותזות ממחלקה ממימד VC d, AdaBoost יחזיר היפותזה ממחלקה במימד VC, פחות או יותר dT. לכן, אם $m \gg dT$, AdaBoost לא יעשה אובפיט.

תהא אם כן, $B \subset \{\pm 1\}^X$ מחלקת היפותזות. עבור $T \ge 1$ נגדיר

$$\mathcal{H}(B,T) = \left\{ x \mapsto \operatorname{sign}\left(\sum_{t=1}^{T} w_t h_t(x)\right) \mid w \in \mathbb{R}^T, h_1,\ldots,h_T \in B \right\}$$

נשים לב שאם הלומד החלש מחזיר היפתזות ב־B, אז AdaBoost מחזיר היפותזה ב־ $\mathcal{H}(B,T)$.

משפט 3.4 עבור B עם $\mathrm{VC}(B) = d$ מתקיים

$$\mathrm{VC}(\mathcal{H}(B,T)) = O\left(dT\log(dT)\right)$$

נוכיח טענה מעט יותר כללית. בהינתן זוג מחלקות $B \subset \{\pm 1\}^X$ ו־$\mathcal{F} \subset \{\pm 1\}^{\{\pm 1\}^T}$, נגדיר

$$\mathcal{F} \circ B := \{x \mapsto f\left(h_1(x), \ldots, h_T(x)\right) \mid h_1, \ldots, h_T \in B,\ f \in \mathcal{F}\}$$

נשים לב ש־$\mathcal{H}(B,T) = \mathcal{F} \circ B$ כאשר $\mathcal{F}$ היא מחלקת ההיפותזות של חצאי מרחב הומגניים על $\mathbb{R}^T$. משפט 3.4 נובע אם כן מהמשפט הבא:

משפט 3.5 $\mathrm{VC}(\mathcal{F} \circ B) \leq (4\mathrm{VC}(\mathcal{F}) + 4\mathrm{VC}(B)T) \log\left(2\mathrm{VC}(\mathcal{F}) + 2\mathrm{VC}(B)T\right)$

הוכחה: נסמן $d_B = \mathrm{VC}(B),\ d_F = \mathrm{VC}(\mathcal{H})$. תהא $A \subset X$ קבוצה המנותצת ע"י $\mathcal{F} \circ B$ בגודל m. עלינו להראות ש־

$$m \leq (4d_F + 4d_B T) \log\left(2d_F + 2d_B T\right)$$

נביט במחלקה $(\mathcal{F} \circ B)|_A$. כל היפותזה במחלקה הנ"ל מוגדרת ע"י T פונקציות $h_1, \ldots, h_T \in B|_A$ וע"י פונקציה $f \in \mathcal{F}|_{h(A)}$ כאשר

$$h(A) := \{(h_1(a), \ldots, h_T(a)) | a \in A\}$$

ממשפט סאור־שלח עבור B, יש לנו $m^{2d_B} \geq$ אפשרויות לבחור כל h_i, ולכן יש לנו יש לנו $m^{2d_B T} = \left(m^{2d_B}\right)^T \geq$ אפשרויות לבחור את $h_1, \ldots, h_T$. בהינתן הבחירה הנ"ל, ממשפט סאור־שלח עבור $\mathcal{F}$, יש לנו $m^{2d_F} \geq |h(A)|^{2d_F} \geq$ אפשרויות לבחור את f. מכאן

$$|(\mathcal{F} \circ B)|_A| \leq m^{2d_F} \cdot m^{2d_B T} = m^{2d_F + 2d_B T}$$

מצד שני, A מנותצת ולכן

$$2^m \leq |(\mathcal{F} \circ B)|_A|$$

משני אי השיוויונות הללו נובע ש־

$$2^m \leq m^{2d_F + 2d_B T}$$

מכאן,

$$\frac{m}{\log(m)} \leq 2d_F + 2d_B T \Rightarrow m \leq (4d_F + 4d_B T) \log\left(2d_F + 2d_B T\right)$$

כאשר הגרירה האחרונה נובעת מהעובדה ש־

$$\forall a, x > 0,\ \frac{x}{\log(x)} \leq a \Rightarrow x \leq 2a \log(a)$$

(למה $A.1$ בספר של שי ושי). ■

3.3 מלמידות חלשה לחזקה - היסטוריה והשלכות תיאורטיות של AdaBoost

ההיסטוריה של AdaBoost החלה בשנת 1988, אז שאלו Mike Kearns ו־Leslie Valiant את השאלה התיאורטית הבאה:

שאלה: נקבע מחלקת היפותזות $\mathcal{H}$. נניח שקיים אלגוריתם יעיל שלומד את $\mathcal{H}$ במקרה הפריד, אבל **באופן חלש** - כלומר, עבור $\frac{1}{2} > \epsilon_0, \delta_0 > 0$ **קבועים** מובטח שהאלגוריתם יחזיר, בהסתברות $1 - \delta_0 \leq$ היפותזה עם שגיאה $\epsilon_0 \geq$. האם ניתן להשתמש בו על מנת לקבל אלגוריתם למידה יעיל העובד לכל ϵ, δ? כלומר, האם קיימת **סכמת האצה** כללית, המאפשרת להפוך אלגוריתמים חלשים יעילים לאלגוריתמים יעילים רגילים (חזקים)?

תשובה חיובית לשאלה ניתנה בשנת 1990 ע"י Rob Schapire, אז דוקטורנט ב־MIT. מספר שנים לאחר מכן, ב־1995, Rob Schapire, יחד עם יואב פרוינד, הציעו את AdaBoost שנתן אף הוא תשובה חיובית לשאלה הנ"ל. היתרון של AdaBoost על פני האלגוריתם הקודם הוא ש־AdaBoost מהיר הרבה יותר.

האלגוריתם של פרוינד ושפירי זכה ועודנו זוכה להצלחה מעשית רבה בשורה של בעיות, ואף זיכה את ממציאיו ב־"פרס גדל" - אחד הפרסים הבולטים במדעי המחשב.

מלבד ההצלחה הפרקטית, לאלגוריתם היו לא מעט השלכות על התיאוריה של למידה, חלקן מפתיעות. וליאנט וקרנס שאלו את השאלה על מנת לקבל כלי המאפשר לפתח אלגוריתמים לבעיות PAC. על פניו, תשובה חיובית על השאלה שלהם תקל מאד את המלאכה של עיצוב אלגוריתמי למידה - במקום לבנות אלגוריתם שצריך להחזיר היפותזה עם שגיאה מאד קטנה, די לפתח אלגוריתם המחזיר היפותזה עם שגיאה קטנה במעט מ־$\frac{1}{2}$.

בשנת 1988, רק 4 שנים לאחר שהחלו לחקור למידה חישובית כחלק ממדעי המחשב, קיום של סכמת האצה כנ"ל, היה נראה בהחלט כמו כמו כלי שיאפשר לפתח אלגוריתמים להרבה בעיות למידה. ברבות השנים התפתחה ההבנה שלמרבה הצער, מלבד חצאי מרחבים, כמעט כל בעיות הלמידה (לפחות כפי שהן מוגדרות כיום) לא ניתנות לפתרון ביעילות. לכן, אחת המטרות המקוריות שלשמן AdaBoost פותח נכשלה.

יתר על כן, באופן אירוני משהו, אחד השימושים של AdaBoost בתיאוריה של למידה הוא על מנת להראות שבעיות למידה הן **מאד** קשות! קיום של סכמת האצה מראה שאם קיים לבעיה אלגוריתם חלש, אז קיים לאותה בעיה גם אלגוריתם חזק. באופן שקול, אם לא קיים אלגוריתם חזק, אז לא קיים אפילו אלגוריתם חלש! לכן העובדה שלא קיים אלגוריתם יעיל גוררת שלא קיים אפילו אלגורים חלש לבעיה! מכאן, כפי שאמרנו מספר פעמים במהלך הקורס, ככל הנראה רוב בעיות ה־PAC הן קשות מאד - במובן שאפילו במקרה פריד לא קיים אלגוריתם יעיל המסוגל להחזיר היפותזה עם שגיאה קטנה, ולו במעט, מ־$\frac{1}{2}$.

נעיר שמלבד ההשלכה הנ"ל, ל־AdaBoost קשרים נוספים להרבה מושגים בסיסיים בלמידה, בינהם רגולריזציה, שוליים רחבים ועוד.

3.4 אפליקציה - זיהוי פנים (Viola and Jones)

ראו מצגת

3.5 דלילות ובחירת פיצ'רים על קצה המזלג

ראו מצגת