دانشکده علوم ریاضی
دانشگاه صنعتی شریف

نظریه یادگیری محاسباتی

امید اعتصامی، محمدهادی فروغمنداعرابی
بهار ۱۳۹۳

تیغ اوکام

جلسه‌های هفتم تا نهم

نگارنده: مریم غرقانی

۱ تیغ اوکام

تیغ اوکام[1] یک اصل حل مسأله است که توسط ویلیام اوکام[2]، فیلسوف و متخصص الهیات انگلیسی، بیان شده‌است. طبق این اصل، از بین فرضیات مختلفی که به یک اندازه پدیده‌های طبیعی را توجیه می‌کنند، بهترین فرضیه، کوتاه‌ترین فرضیه است. در این بخش، از منظر یادگیری، به این اصل توجه می‌کنیم و یادگیری اوکام را معرفی می‌کنیم.

۱.۱ یادگیری اوکام

فرض کنید $X = \cup_{n\geq 1} X_n$ فضای موردها، $\mathcal{C} = \cup_{n\geq 1} \mathcal{C}_n$، کلاس مفهوم و $\mathcal{H} = \cup_{n\geq 1} \mathcal{H}_n$، کلاس نمایش فرضیه است. در این بخش فرض می‌کنیم نمایش فرضیه‌ها به صورت دودویی، با الفبای $\{0,1\}$، است و بنابراین $\mathbf{size}(h)$، طول رشته دودویی

[1] Occam's razor
[2] William of Ockham

h است.

فرض کنید $c \in \mathcal{C}_n$، مفهوم هدف باشد. نمونه برچسب‌دار S با اندازه m، مجموعه‌ای از زوج‌ها به صورت زیر است:

$$S = \{\langle x_1, c(x_1)\rangle, \ldots, \langle x_m, c(x_m)\rangle\}$$

تعریف ۱. *فرض کنید $0 \leq \alpha$ و $1 > \beta > 0$ دو عدد ثابت هستند. گوییم L یک **الگوریتم** (α, β) **ــ اوکام** برای کلاس مفهوم $\mathcal{C}$ با استفاده از $\mathcal{H}$ است، اگر L با ورودی نمونه برچسب‌دار S با اندازه m، خروجی $h \in \mathcal{H}$ را بدهد که دارای شرایط زیر است:*

۱. *h با نمونه S سازگار است.*

۲. $\textbf{size}(h) \leq (n \cdot \textbf{size}(c))^\alpha m^\beta$

*گوییم L یک الگوریتم (α, β)ــ اوکام **کارا** است، اگر زمان اجرای L بر حسب n، m و $\textbf{size}(c)$ باشد.*

همان‌طور که از تعریف برمی‌آید، رشد $\textbf{size}(h)$ نسبت به m کندتر از رشد خطی است ($\beta < 1$). اگر اجازه می‌دادیم $\beta \geq 1$، الگوریتم L می‌توانست یک حفظ‌کننده باشد؛ یعنی hای را تولید می‌کرد که صرفا به هر مثال، برچسب آن را نسبت می‌دهد.

قضیه ۲. ***تیغ اوکام.*** *فرض کنید L یک الگوریتم (α, β)ــ اوکام کارا برای کلاس مفهوم $\mathcal{C}$ با استفاده از $\mathcal{H}$ باشد. همچنین فرض کنید $\mathcal{D}$، یک توزیع احتمال روی فضای موردهای x است و $c \in \mathcal{C}_n$، مفهوم هدف است و $0 < \epsilon, \delta \leq 1$. در این صورت یک ثابت $a > 0$ وجود دارد به‌طوری‌که اگر L، نمونه ورودی S شامل m مثال تصادفی تولید شده از اوراکل $Ex(c, \mathcal{D})$ را به عنوان ورودی دریافت کند، که m دارای خاصیت زیر باشد:*

$$m \geq a\left(\frac{1}{\epsilon}\log\frac{1}{\delta} + \left(\frac{(n \cdot \textbf{size}(c))^\alpha}{\epsilon}\right)^{\frac{1}{1-\beta}}\right)$$

آنگاه با احتمال حداقل $1 - \delta$، خروجی h از الگوریتم L دارای خطای حداکثر ϵ است.

طبق قضیه، اگر β به ۱ نزدیک شود، تعداد مثال‌های لازم m به سمت بی‌نهایت میل می‌کند. این موضوع بیانگر آن است که اگر اندازه خروجی h به تعداد مثال‌ها نزدیک شود، الگوریتم L تقریبا یک حفظ‌کننده خواهد بود که در این صورت برای اینکه خطای خروجی‌اش را پایین بیاورد، باید نمونه‌ای با تعداد بسیار زیادی مثال دریافت کند.

قضیه ۳. ***تیغ اوکام (نسخه اندازه‌ای).*** *فرض کنید $\mathcal{C}$ کلاس مفهوم و $\mathcal{H}$ کلاس نمایش فرضیه باشد. فرض کنید L الگوریتمی باشد که به ازای هر عدد طبیعی n و هر $c \in \mathcal{C}$، اگر L نمونه ورودی S را با m مثال برچسب‌دار از c دریافت کند، در زمان چندجمله‌ای بر حسب n، m و $\textbf{size}(c)$ اجرا می‌شود و خروجی $h \in \mathcal{H}_{n,m}$ را تولید می‌کند که با نمونه S سازگار است. آنگاه ثابت $b > 0$ وجود دارد به‌طوری‌که به ازای هر n، هر توزیع $\mathcal{D}$ روی X_n و هر مفهوم هدف $c \in \mathcal{C}_n$، اگر L یک نمونه S شامل m مثال تصادفی تولید شده از $Ex(c, \mathcal{D})$ را به عنوان ورودی بگیرد، که در آن*

$$\log|\mathcal{H}_{n,m}| \leq b\epsilon m - \log\frac{1}{\delta}$$

آنگاه L خروجی $h \in \mathcal{H}$ را می‌دهد که با احتمال حداقل $1 - \delta$، خطای h حداکثر ϵ است.

اثبات. فرض کنید $\textbf{error}(h) > \epsilon$. در این صورت با آمدن هر مثال، h با احتمال حداقل ϵ رد می‌شود. پس احتمال رد نشدن h با یک مثال حداکثر $1 - \epsilon$ است. بنابراین احتمال رد نشدن h با m نمونه مستقل حداکثر $(1 - \epsilon)^m$ است. در نتیجه داریم:

$$Pr\{\exists h \in \mathcal{H}_{n,m} : \textbf{error}(h) > \epsilon \text{ و } h \text{ پس از } m \text{ نمونه باقی بماند}\} \leq |\mathcal{H}_{n,m}|(1-\epsilon)^m$$

قرار می دهیم:

$$|\mathcal{H}_{n,m}|(1-\epsilon)^m \leq \delta \quad (1)$$

$$\Leftrightarrow |\mathcal{H}_{n,m}| \leq \frac{\delta}{(1-\epsilon)^m} \quad (2)$$

$$\Leftrightarrow \log|\mathcal{H}_{n,m}| \leq m\log\frac{1}{1-\epsilon} - \log\frac{1}{\delta} \leq mb\epsilon - \log\frac{1}{\delta} \quad (3)$$

بنابراین احتمال اینکه خطای خروجی الگوریتم بیشتر از ϵ باشد، حداکثر δ است. □

حال قضیه ۲ را ثابت می‌کنیم:

اثبات. فرض کنید L یک الگوریتم (α,β)ــاوکام است و $\mathcal{H}_{n,m}$ فضای تمام خروجی‌های تولید شده الگوریتم L باشد در صورتی‌که ورودی L یک نمونه با اندازه m برچسب‌دار شده با یک مفهوم $c \in \mathcal{C}$ باشد. اگر $h \in \mathcal{H}_{n,m}$ خروجی L باشد، طبق تعریف، $\mathbf{size}(h) \leq (n\cdot\mathbf{size}(c))^\alpha m^\beta$ و در نتیجه داریم، $\mathcal{H}_{n,m} \leq 2^{(n\cdot\mathbf{size}(c))^\alpha m^\beta}$.
کافی است ثابت کنیم اگر $m \geq a\left(\frac{1}{\epsilon}\log\frac{1}{\delta} + \left(\frac{(n\cdot\mathbf{size}(c))^\alpha}{\epsilon}\right)^{\frac{1}{1-\beta}}\right)$ ، آنگاه ثابت $b > 0$ وجود دارد به‌طوری‌که $\log|\mathcal{H}_{n,m}| \leq b\epsilon m - \log\frac{1}{\delta}$. در این صورت با استفاده از نسخه اندازه‌ای قضیه تیغ اوکام، می‌توانیم نتیجه مطلوب را به‌دست آوریم.
در واقع می‌خواهیم شرط زیر برقرار باشد:

$$m \geq \frac{1}{b\epsilon}\left(\log|\mathcal{H}_{n,m}| + \log\frac{1}{\delta}\right) \quad (4)$$

برای رسیدن به این نامساوی، کافی است دو شرط زیر برقرار باشد:

۱. $m \geq \frac{2}{b\epsilon}\log|\mathcal{H}_{n,m}|$

۲. $m \geq \frac{2}{b\epsilon}\log\frac{1}{\delta}$

شرط ۱ معدل است با:

$$m \geq \frac{2}{b\epsilon}\left(\frac{1}{\epsilon}\log\frac{1}{\delta} + \left(\frac{(n\cdot\mathbf{size}(c))^\alpha}{\epsilon}\right)^{\frac{1}{1-\beta}}\right) \quad (5)$$

$$\Leftrightarrow m \geq \left(\frac{2}{b\epsilon}(n\cdot\mathbf{size}(c))^\alpha\right)^{\frac{1}{1-\beta}} \quad (6)$$

بنابراین کافی است شرط زیر برقرار باشد تا شرایط ۱ و ۲ ارضا شوند:

$$m \geq \log\frac{1}{\delta} + \left(\frac{2}{b\epsilon}(n\cdot\mathbf{size}(c))^\alpha\right)^{\frac{1}{1-\beta}} \quad (7)$$

کافی است قرار دهیم $b = \frac{2}{a} > 0$ تا شرط بالا ارضا شود. □

۲.۱ بهبود اندازه نمونه برای یادگیری عطف

در جلسات قبل یک الگوریتم برای یادگیری عطف[3] ارائه شد که فرمول عطف $h \in \mathcal{H}_{n,m}$ را تولید می‌کرد. اگر نمونه S حداقل یک مثال داشته باشد ($m \geq 1$)، به ازای هر متغیر x_i، h یا شامل x_i است، یا $\overline{x_i}$ و یا هیچ‌یک از x_i و $\overline{x_i}$ در h وجود ندارد. اگر

[3] conjunction

هیچ مثالی تولید نشده‌باشد ($m = ۰$)، h شامل همه $۲n$ لیترال ممکن است. بنابراین تعداد کل حالاتی که h می‌تواند داشته باشد، به صورت زیر است:

$$\mathcal{H}_{n,m} \leq ۳^n + ۱ \leq ۴^n \;\; \Rightarrow \;\; \log \mathcal{H}_{n,m} \leq ۲n \tag{۸}$$

قرار می‌دهیم:

$$۲n \leq b\epsilon m - \log \frac{۱}{\delta} \tag{۹}$$

$$\Rightarrow m \geq \frac{۱}{b\epsilon}(۲n + \log \frac{۱}{\delta}) \tag{۱۰}$$

$$\Rightarrow m = O(\frac{n}{\epsilon} + \frac{۱}{\epsilon} \log \frac{۱}{\delta}) \tag{۱۱}$$

کرانی که برای اندازه نمونه از قضیه اوکام حاصل شد، از شرط $m \geq \frac{\log \frac{۲n}{\delta}}{\frac{\epsilon}{۲n}}$ که در الگوریتم یادگیری عطف به‌دست آوردیم، بهتر است. الگوریتم قبلی، فرمول عطف با بیشترین طول را که با نمونه سازگار بود، به‌دست می‌آورد. در ادامه الگوریتمی را برای یادگیری عطف ارائه می‌دهیم که در آن سعی می‌کنیم یک فرمول عطف سازگار با نمونه با کوتاهترین طول را به‌دست آوریم.

۱.۲.۱ یادگیری عطف با تعداد کمی لیترال

در این قسمت یک الگوریتم کاراتر برای یادگیری عطف ارائه می‌دهیم. الگوریتم جدید علاوه بر مثال‌های مثبت از مثال‌های منفی نیز استفاده می‌کند برای اینکه طول خروجی را کم کند. قبل از ارائه الگوریتم ابتدا یک مسأله ترکیبیاتی و یک راه‌حل تقریبی را برای آن معرفی می‌کنیم.

تعریف ۴. مسأله پوشش مجموعه‌ای[۴]. *فرض کنید یک گردایه $\mathcal{S}$ از زیرمجموعه‌های $U = \{۱, \cdots, m\}$ را به عنوان ورودی داریم؛ هدف پیدا کردن زیرگردایه $\mathcal{T} \subseteq \mathcal{S}$ است که $|\mathcal{T}|$ کمینه باشد و $\mathcal{T}$، U را بپوشاند:*

$$\cup_{t \in \mathcal{T}} = U \tag{۱۲}$$

مسأله پوشش مجموعه‌ای یک مسأله $NP-hard$ معروف است. فرض کنید $c = OPT(\mathcal{S})$، اندازه پوشش کمینه برای مورد $\mathcal{S}$ در این مسأله است. در ادامه، یک الگوریتم تقریبی حریصانه را معرفی می‌کنیم که یک پوشش $\mathcal{R}$، با اندازه حداکثر $c \;\log m$ پیدا می‌کند:

الگوریتم به این صورت است که در ابتدا $\mathcal{R}$ را برابر با یک گردایه خالی قرار می‌دهد. در هر مرحله، مجموعه $s^* \in \mathcal{S}$ با بزرگترین اندازه را به $\mathcal{R}$ اضافه می‌کند؛ سپس $\mathcal{S}$ را به این صورت به‌روز می‌کند که به ازای هر s در $\mathcal{S}$، $s - s^*$ را جایگزین می‌کند. این رویه را تا زمانی ادامه می‌دهد که $\mathcal{R}$، همه U را بپوشاند.

قضیه ۵. *الگوریتم حریصانه یک پوشش مجموعه‌ای با اندازه حداکثر $c \log m$ پیدا می‌کند.*

اثبات. فرض کنید $U_i \subseteq U$ مجموعه‌ای از عناصر است که بعد از i مرحله در الگوریتم، توسط $\mathcal{R}$ پوشیده نشده‌اند. یک زیرگردایه از S با اندازه حداکثر c وجود دارد که U_i را می‌پوشاند. (چون یک زیرگردایه با اندازه c وجود دارد که U را می‌پوشاند.) پس در مرحله $i+۱$ام، یک مجموعه در S وجود دارد که اندازه آن حداقل $\frac{U_i}{c}$ است. پس داریم:

$$|U_{i+۱}| \leq |U_i| - \frac{U_i}{c} = |U_i|(۱ - \frac{۱}{c}) \tag{۱۳}$$

[۴] the set cover problem

با استقرا روی i نتیجه می‌گیریم:

$$|U_i| \leq m\left(1-\frac{1}{c}\right)^i \leq me^{-\frac{i}{c}} = e^{\log m - \frac{i}{c}} \quad (14)$$

قرار می‌دهیم:

$$e^{\log m - \frac{i}{c}} < 1 \Rightarrow \log m - \frac{i}{c} < 0 \Rightarrow c\ \log m < i \quad (15)$$

بنابراین الگوریتم حداکثر $c\ \log m$ مرحله ادامه می‌یابد؛ در نتیجه با حداکثر $c\ \log m$ مجموعه از S، کل U را می‌پوشاند. □

حال یک الگوریتم برای یادگیری عطف ارائه می‌کنیم و سپس با استفاده از قضیه تیغ اوکام، یک کران پایین برای اندازه نمونه به‌دست می‌آوریم.

الگوریتم جدید به این صورت است که وقتی یک نمونه S با m مثال را به عنوان ورودی دریافت می‌کند، الگوریتم یادگیری عطف قدیمی را روی S اجرا می‌کند تا خروجی h را تولید کند. سپس از مثال‌های منفی استفاده می‌کند تا لیترال‌های اضافی را حذف کند. توجه کنید که حذف لیترال‌ها، تاثیری روی سازگار بودن h با مثال‌های مثبت ندارد. فقط باید لیترال‌ها را طوری حذف کنیم که فرضیه h همچنان با مثال‌های منفی سازگار باشد. برای این کار، این مسأله را به یک مسأله پوشش مجموعه‌ای تبدیل می‌کنیم و الگوریتم تقریبی حریصانه را روی آن اعمال می‌کنیم.

برای هر لیترال zدر h، N_z را مجموعه تمام مثال‌های منفی $\langle a, 0\rangle$ در S تعریف می‌کنیم که z در a، ۰ است. اگر گردایه‌ای از N_zها داشته باشیم که کل مثال‌های منفی S را می‌پوشانند، اگر h' را عطف لیترال‌های این گردایه تعریف کنیم، در این صورت h' با تمام مثال‌های منفی S سازگار است. می‌خواهیم کوچکترین گردایه از N_zها را پیدا کنیم که تمام مثال‌های منفی S را بپوشانند. این یک مسأله پوشش مجموعه‌ای است و اگر از الگوریتم حریصانه برای حل آن استفاده کنیم، یک پوشش با اندازه حداکثر $\mathbf{size}(c)\ \log m$ به‌دست می‌آوریم. بنابراین الگوریتم جدید، یک فرضیه با اندازه حداکثر $\mathbf{size}(c)\ \log m\ \ \log n$ تولید می‌کند. (فرضیه، حداکثر $\mathbf{size}(c)\ \log m$ لیترال دارد و اندازه هر لیترال $\log n$ بیت است.) در اینجا، $\mathcal{H}_{n,m}$، مجموعه تمام عطف‌های با حداکثر $\mathbf{size}(c)\ \log m$ است که برای آن داریم، $|\mathcal{H}_{n,m}| \leq 2^{\mathbf{size}(c)\ \log m\ \ \log n}$.

طبق نسخه اندازه‌ای قضیه تیغ اوکام، اگر شرط زیر برقرار باشد، الگوریتم ارائه شده، یک الگوریتم یادگیری با پارامترهای ϵ و δ است.

$$\frac{c\,(\log\,|\mathcal{H}_{n,m}| + \log\,\frac{1}{\delta})}{\epsilon} \leq m \quad (16)$$

در صورتی که شرط زیر برقرار باشد، این شرط قضیه تیغ اوکام ارضا می‌شود:

$$m \geq c_1\left(\frac{1}{\epsilon}\log\frac{1}{\delta} + \frac{\mathbf{size}(c)\ \log\ n(\log\ \mathbf{size}(c) + \log\ \ \log\ n)}{\epsilon}\right) \quad (17)$$

با کمی تغییر در این الگوریتم، می‌توانیم کران بهتری برای اندازه نمونه به‌دست آوریم. به‌طور کلی همه الگوریتم‌هایی که تاکنون معرفی کردیم، فرضیه hای را پیدا می‌کردند که با همه مثال‌های نمونه سازگار باشد. اگر این قید را برداریم و اجازه دهیم کمی خطا در نمونه داشته باشیم، ممکن است بتوانیم الگوریتم یادگیری بهتری داشته باشیم. همانند الگوریتمی که در این قسمت ارائه دادیم، عمل می‌کنیم؛ فقط الگوریتم تقریبی پوشش مجموعه‌ای را تا جایی ادامه می‌دهیم که به $|S_i| < \frac{\epsilon}{2}m$ برسیم:

$$|S_i| < me^{-\frac{i}{c}} < \frac{\epsilon}{2}m \quad (18)$$

$$\Rightarrow e^{-\frac{i}{c}} < \frac{\epsilon}{2} \quad (19)$$

$$\Rightarrow i > c\ \log\frac{2}{\epsilon} \quad (20)$$

بنابراین اگر $h \in \mathcal{H}_{n,m}$ خروجی این الگوریتم باشد، تعداد لیترال‌های h حداکثر $\textbf{size}(c) \log \frac{2}{\epsilon}$ است و چون هر لیترال را با $\log n$ بیت نمایش می‌دهیم، $\textbf{size}(h)$ حداکثر $\textbf{size}(c) \log \frac{2}{\epsilon} \log n$ است. در نتیجه داریم:

$$|\mathcal{H}_{n,m}| \leq 2^{\textbf{size}(c) \log \frac{2}{\epsilon} \log n} \tag{21}$$

قضیه ۶. *اگر در الگوریتم یادگیری عطف از الگوریتم حریصانه تقریبی با خطای $\frac{\epsilon}{2}$ استفاده کنیم، خروجی h تولید می‌شود که خطایش با زیاد شدن نمونه به صورت نمایی کم می‌شود.*

قبل از اثبات این قضیه، ابتدا کران چرنوف را معرفی می‌کنیم:

لم ۷. *کران چرنوف.* *فرض کنید $X_1, \cdots, X_m$، متغیرهای تصادفی برنولی مستقل با احتمال موفقیت p هستند. فرض کنید $S = X_1 + \cdots + X_m$ **تعداد کل موفقیت‌ها** در $X_1, \cdots, X_m$ است؛ بنابراین $\mathbb{E}[S] = \sum_{i=1}^{m} \mathbb{E}[X_i] = pm$. در این صورت اگر $0 \leq \gamma \leq 1$، آن‌گاه داریم:*

$$\mathbb{P}r[S > (1+\gamma)pm] \leq e^{-\frac{mp\gamma^2}{3}} \tag{22}$$

$$\mathbb{P}r[S < (1-\gamma)pm] \leq e^{-\frac{mp\gamma^2}{2}} \tag{23}$$

حال به اثبات قضیه می‌پردازیم:

اثبات. فرض کنید $\textbf{error}(h) > \epsilon$. در نتیجه احتمال لو رفتن h با یک مثال حداقل ϵ است. متغیر نشان‌گر X_i $(1 \leq i \leq m)$ را به این صورت تعریف می‌کنیم که اگر h با مثال i ام سازگار نباشد، مقدار آن ۱ و در غیر این صورت، مقدار آن ۰ است.

$$\mathbb{E}[X_i] = Pr\{X_i = 1\} > \epsilon \;\Rightarrow\; \mathbb{E}[\sum_{i=1}^{m} X_i] > \epsilon m \tag{24}$$

با استفاده از کران چرنوف داریم:

$$\mathbb{P}r\{\sum_{i=1}^{m} X_i < \frac{\epsilon}{2}m\} = \mathbb{P}r\{\sum_{i=1}^{m} X_i < (1-\frac{1}{2})\epsilon m\} \leq e^{-m\frac{\epsilon}{8}} \tag{25}$$

پس احتمال اینکه خطای یک h بد روی نمونه S کمتر از $\frac{\epsilon}{2}m$ باشد، حداکثر $e^{-m\frac{\epsilon}{8}}$ است. بنابراین احتمال اینکه یک h بد وجود داشته باشد که خطایش روی نمونه S کمتر از $\frac{\epsilon}{2}m$ باشد، حداکثر $|\mathcal{H}_{n,m}|e^{-m\frac{\epsilon}{8}}$ است. کافی است این مقدار را کمتر از δ قرار دهیم تا مطمئن باشیم احتمال اینکه خطای خروجی الگوریتم بیشتر از ϵ باشد، کمتر از δ خواهد بود.

$$\delta > |\mathcal{H}_{n,m}|e^{-m\frac{\epsilon}{8}} \;\Rightarrow\; \log \delta > \log |\mathcal{H}_{n,m}| - m\frac{\epsilon}{8} \tag{26}$$

بنابراین کران زیر برای اندازه نمونه حاصل می‌شود: (به ازای یک ثابت c_1)

$$m > c_1(\frac{1}{\epsilon}\log |\mathcal{H}_{n,m}| + \frac{1}{\epsilon}\log \frac{1}{\delta}) \tag{27}$$

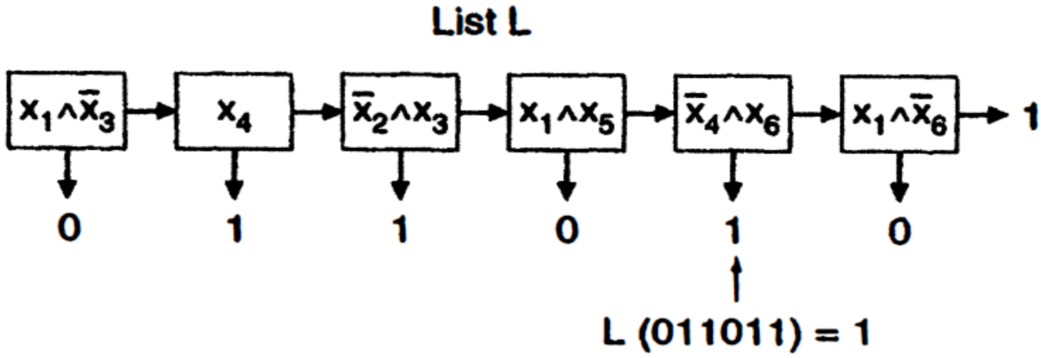
$$\Rightarrow m > c_1 \left(\frac{1}{\epsilon}\log \frac{1}{\delta} + \frac{\textbf{size}(c) \log \frac{2}{\epsilon} \log n}{\epsilon} \right) \tag{28}$$

□

۳.۱ یادگیری لیست تصمیم

تعریف ۸. یک k-*لیست تصمیم* [۵] $(k-DL)$ *روی متغیرهای بولی* $x_n, \cdots, x_1$*، یک دنباله مرتب* $L=(c_1,b_1),\cdots(c_l,b_l)$ *با یک بیت* b *است؛ به این صورت که هر* c_i*، عطف حداکثر* k *لیترال است. برای هر ورودی* $a\in\{0,1\}^n$*، اگر* j *کوچکترین اندیسی باشد که* $c_j(a)=1$*، آنگاه* $L(a)=b_j$*؛ اگر چنین اندیسی وجود نداشته باشد، آنگاه* $L(a)=b$.

شکل زیر نشان‌دهنده یک $DL-2$ به همراه مقداردهی آن روی یک ورودی خاص است:



شکل ۱: نمایش یک DL-۲

قضیه ۹. *به ازای هر عدد ثابت* $k\geq 1$*،* $k-DL$ *قابل یادگیری کارای* PAC *است.*

اثبات. یک الگوریتم یادگیری برای $DL-1$ ارائه می‌دهیم. برای k کلی، می‌توانیم به راحتی مسأله یادگیری $k-DL$ را به این مسأله تحویل کنیم. این کار همانند روشی است که برای یادگیری $k-CNF$ ارائه دادیم؛ در واقع کافی است برای هر عطف k متغیری، یک متغیر جدید تعریف کنیم و یک $DL-1$ با متغیرهای جدید را یاد بگیریم.
الگوریتم نمونه S را به عنوان ورودی می‌گیرد و ابتدا با یک لیست تصمیم خالی شروع می‌کند. در هر مرحله یک لیترال z را پیدا می‌کند که $S_z\subseteq S$ (مجموعه تمام مثال‌هایی که z در آن‌ها ۱ است)، ناتهی باشد و فقط شامل مثال‌های مثبت یا فقط شامل مثال‌های منفی باشد. در این صورت به z یک لیترال مفید می‌گوییم. سپس الگوریتم جعبه z را به انتهای لیست تصمیم اضافه می‌کند. اگر S_z فقط شامل مثال‌های مثبت باشد، بیت مربوط به z را ۱ قرار می‌دهد؛ در غیر این صورت، بیت مربوط به z را ۰ قرار می‌دهد. سپس به جای S، $S-S_z$ را جاگذاری می‌کند و همین رویه را تکرار می‌کند تا به $S=\emptyset$ برسد. در این صورت همه مثال‌ها با لیست تصمیمی که الگوریتم تولید می‌کند، سازگار هستند.
باید ثابت کنیم در هر مرحله که یک مجموعه S از مثال‌ها باقی مانده، حتما یک لیترال مفید وجود دارد. باید توجه کنیم که مفهوم هدف، یک $DL-1$ است که نمونه از آن تولید شده است. در $DL-1$ اصلی اولین لیترال z را در نظر بگیرید که S_z-S ناتهی است؛ در واقع z اولین لیترالی است که در $DL-1$ اصلی، حداقل یکی از اعضای S (مثال‌های باقی‌مانده) در آن حذف می‌شود. در این صورت z یک لیترال مفید است.
هر $DL-1$ با n متغیر، با $n\log n$ بیت کد می‌شود. زمان اجرای الگوریتم ارائه شده، بر حسب m چندجمله‌ای است. طبق نسخه اندازه‌ای قضیه تیغ اوکام، اگر $m\geq c_1((\frac{1}{\epsilon})(\log\frac{1}{\delta}+n\log n))$، یک الگوریتم یادگیری کارای PAC با پارامترهای ϵ و δ داریم. □

[۵] k- desision list

۲ یادگیری PAC انکاری

تا کنون فرض می‌کردیم مثال‌ها از یک توزیع $\mathcal{D}$ روی فضای موردها تولید شده و با یک مفهوم هدف c برچسب‌گذاری شده‌اند. در حالت کلی می‌توانیم فرض کنیم که لزوما یک مفهوم هدف وجود ندارد و مثال‌ها از یک توزیع $\mathcal{D}$ روی $\mathcal{X} \times \mathcal{Y}$ تولید می‌شوند که $\mathcal{X}$، فضای موردها و $\mathcal{Y}$، مجموعه برچسب‌هاست. در این حالت باید یک مدل کلی‌تر برای یادگیری PAC ارائه دهیم.

خطای h نسبت به توزیع $\mathcal{D}$ به صورت زیر تعریف می‌شود:

$$L_{\mathcal{D}}(h) = Pr_{(x,y)\sim\mathcal{D}}[h(x) \neq y] \tag{۲۹}$$

تعریف ۱۰. *گوییم $\mathcal{H}$ **قابل یادگیری PAC انکاری**[۶] است، اگر یک الگوریتم یادگیری L وجود داشته باشد که به ازای هر $0 < \epsilon, \delta < 1$ و هر توزیع $\mathcal{D}$ روی $\mathcal{X} \times \mathcal{Y}$، یک عدد طبیعی m وجود داشته باشد که اگر m مثال مستقل با توزیع $\mathcal{D}$ دریافت کند، خروجی h را تولید می‌کند که با احتمال حداقل $1 - \delta$ خاصیت زیر را دارد:*

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon \tag{۳۰}$$

۳ کمینه‌سازی ریسک تجربی(ERM)

هدف کلی در یادگیری این است که فرضیه‌ای را تولید کنیم که خطای آن نسبت به مفهوم هدف و توزیع $\mathcal{D}$ روی تمام فضای موردها کم باشد. اما یادگیرنده مفهوم هدف و توزیع $\mathcal{D}$ را نمی‌داند؛ یک روش برای تخمین خطای فرضیه این است که خطای تجربی (خطا روی نمونه) فرضیه h را به‌دست آوریم. خطای h روی نمونه $S = \{\langle x_1, y_1\rangle, \cdots, \langle x_m, y_m\rangle\}$ به صورت زیر تعریف می‌شود:

$$L_S(h) = \frac{1}{m}\sum_{i=1}^{m} I(h(x_i) \neq y_i) \tag{۳۱}$$

تعریف ۱۱. *اگر الگوریتم یادگیری L به ازای هر نمونه S، خروجی h با $L_S(h)$ کمینه را تولید کند، به آن یک الگوریتم ERM[۷] می‌گوییم.*

۴ همگرایی یکنواخت

تعریف ۱۲. *گوییم نمونه S یک ϵــ نماینده از $\mathcal{X}$، $\mathcal{D}$ و $\mathcal{H}$ است، اگر*

$$\forall h \in \mathcal{H} : |L_{\mathcal{D}}(h) - L_S(h)| \leq \epsilon \tag{۳۲}$$

قضیه ۱۳. *اگر S یک $\frac{\epsilon}{2}$ــ نماینده از $\mathcal{X}$، $\mathcal{D}$ و $\mathcal{H}$ باشد و h خروجی یک الگوریتم ERM با ورودی S باشد، داریم:*

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon \tag{۳۳}$$

اثبات. به ازای هر $h' \in \mathcal{H}$ داریم:

$$L_{\mathcal{D}}(h) \leq L_S(h) + \frac{\epsilon}{2} \leq L_S(h') + \frac{\epsilon}{2} \leq L_{\mathcal{D}}(h) \leq L_S(h) + \epsilon \tag{۳۴}$$

چون این نامساوی به ازای هر $h' \in \mathcal{H}$ برقرار است، حکم قضیه ثابت می‌شود. $\square$

[۶] Agnostic PAC Learnable
[۷] Empirical Risk Minimization

تعریف ۱۴. *گوییم کلاس فرضیه $\mathcal{H}$ **همگرایی یکنواخت**[۸] دارد، اگر به ازای هر ϵ و δ و هر توزیع $\mathcal{D}$، یک **عدد طبیعی** m **وجود داشته باشد** به‌طوری‌که اگر نمونه S شامل m مثال مستقل تولید شده از توزیع $\mathcal{D}$ باشد، **با احتمال حداقل** $1-\delta$، نمونه S یک ϵـ نماینده از $\mathcal{X}$، $\mathcal{D}$ و $\mathcal{H}$ باشد.*

لم ۱۵. *اگر $\mathcal{H}$ خاصیت همگرایی یکنواخت داشته باشد، $\mathcal{H}$ قابل یادگیری PAC انکاری است.*

اثبات. پارامترهای ϵ و δ و توزیع دلخواه $\mathcal{D}$ در نظر بگیرید. به ازای $\frac{\epsilon}{2}$ و δ، یک m وجود دارد که اگر نمونه S شامل m مثال مستقل تولید شده از توزیع $\mathcal{D}$ باشد، به احتمال حداقل $1-\delta$، S یک $\frac{\epsilon}{2}$ـ نماینده از $\mathcal{X}$، $\mathcal{D}$ و $\mathcal{H}$ است. بنابراین اگر نمونه S را به یک الگوریتم ERM بدهیم، خروجی h را تولید می‌کند که دارای خاصیت زیر است:

$$L_{\mathcal{D}}(h) \leq \min_{h' \in \mathcal{H}} L_{\mathcal{D}}(h') + \epsilon \qquad (۳۵)$$

□

مثال ۱۶. *کلاس همه مستطیل‌های با اضلاع موازی محورهای مختصات خاصیت همگرایی یکنواخت دارد.*

مثال ۱۷. *کلاس همه توابع $f: \{0,1\}^{2m} \rightarrow \{0,1\}$ قابل یادگیری PAC انکاری نیست و در نتیجه همگرایی یکنواخت ندارد.*

قضیه ۱۸. *اگر $\mathcal{H}$ متناهی باشد، قابل یادگیری PAC انکاری است.*

اثبات.

$$Pr_S[\exists h \in \mathcal{H} : |L_{\mathcal{D}}(h) - L_S(h)| > \epsilon] \leq \sum_{h \in \mathcal{H}} Pr_S[|L_{\mathcal{D}}(h) - L_S(h)| > \epsilon] \leq \sum_{h \in \mathcal{H} 2 e^{-2m\epsilon^2}} \leq 2|\mathcal{H}|e^{-2m\epsilon^2} \qquad (۳۶)$$

توجه کنید که $L_S(h)$ یک متغیر تصادفی با میانگین $L_{\mathcal{D}}(h)$ است و نامساوی بالا با استفاده از کران چرنوف به‌دست آمده است. قرار می‌دهیم:

$$2|\mathcal{H}|e^{-2m\epsilon^2} < \delta \qquad (۳۷)$$

$$m > \frac{\log \frac{2|\mathcal{H}|}{\delta}}{2\epsilon^2} \qquad (۳۸)$$

در این صورت، $\mathcal{H}$ همگرایی یکنواخت دارد و در نتیجه قابل یادگیری PAC انکاری است. □

قضیه ۱۹. ***نهار مجانی در کار نیست!***[۹] *فرض کنید A یک الگوریتم یادگیری برای فضای موردهای $\mathcal{X}$ است. اگر $m < \frac{|\mathcal{X}|}{2}$، آن‌گاه یک توزیع $\mathcal{D}$ روی $\mathcal{X} \times \{0,1\}$ **وجود دارد** به‌طوری‌که:*

۱. *یک تابع $f: \mathcal{X} \rightarrow \{0,1\}$ **وجود دارد** که $L_{\mathcal{D}}(f) = 0$*

۲. *اگر یک نمونه S شامل m مثال مستقل تولید شده از توزیع $\mathcal{D}$ داشته باشیم، **با احتمال حداقل** $\frac{1}{7}$، داریم: $L_{\mathcal{D}}(A(S)) \geq \frac{1}{8}$*

یک نتیجه مهم که از این قضیه می‌گیریم این است که اگر $\mathcal{H}$ مجموعه همه توابع $f: \mathcal{X} \rightarrow \{0,1\}$ باشد، تقریبا نمی‌توانیم $\mathcal{H}$ را یاد بگیریم، مگر اینکه اندازه نمونه خیلی بزرگ باشد و نمونه را حفظ کنیم. در واقع باید کلاس فرضیه را محدود کنیم تا بتوانیم با قواعد ساده (نه چندان پیچیده) آن را یاد بگیریم.
حال به اثبات قضیه می‌پردازیم:

[۸] Uniform Convergence
[۹] No free lunch

اثبات. فرض کنید $|\mathcal{X}| > m$. توزیع $\mathcal{D}$ را یک توزیع یکنواخت روی $\mathcal{X}$ تعریف می‌کنیم. فرض کنید f یک تابع کاملا تصادفی از $\mathcal{X}$ به $\{0,1\}$ است. یعنی به ازای هر $\S \in \mathcal{X}$، $f(x)$ با احتمال $\frac{1}{2}$، ۰ و با احتمال $\frac{1}{2}$، ۱ است. فرض کنید نمونه S شامل m مثال مستقل با توزیع $\mathcal{D}$ از $\mathcal{X}$ تولید شده و با تابع f برچسب‌گذاری شده‌است. الگوریتم A، نمونه S را دریافت می‌کند و یک فرضیه $h = A(S)$ را تولید می‌کند. می‌دانیم که حداقل m مورد $x \in \mathcal{X}$ وجود دارد که در نمونه S نیامده‌اند. چون مفهوم هدف، یک تابع کاملا تصادفی است، بهترین تصمیمی که الگوریتم A می‌تواند برای موارد $x \in \mathcal{X} - S$ بگیرد، این است که برچسب آن‌ها را حدس بزند. یعنی با احتمال $\frac{1}{2}$، هر یک از برچسب‌های ۰ و ۱ را انتخاب کند. در این صورت، به هر یک از این موارد با احتمال $\frac{1}{2}$، برچسب اشتباه نسبت می‌دهد. بنابراین داریم:

$$E_f[E_S[L_{\mathcal{D}}(A(S))]] = E_f[\frac{1}{2m}\sum_{x\in\mathcal{X}} \mathbb{P}r\{h(x) \neq f(x)\}] \quad (۳۹)$$

$$\geq E_f[\frac{1}{2m}\sum_{x\in\mathcal{X}-S} \mathbb{P}r\{h(x) \neq f(x)\}] \quad (۴۰)$$

$$= \frac{1}{2m}\frac{1}{2}|X - S| \geq \frac{1}{2} \quad (۴۱)$$

$$\geq \frac{1}{2m}(\frac{1}{2}m) = \frac{1}{4} \quad (۴۲)$$

بنابراین یک f وجود دارد که به ازای آن داریم:

$$E_S[L_{\mathcal{D}}(A(S))] \geq \frac{1}{4}$$

که در آن D توزیعی روی $\mathcal{X} \times \{0,1\}$ به صورت زیر است:

$$D(x,y) = \begin{cases} \frac{1}{|\mathcal{X}|} & \text{اگر } y = f(x) \\ 0 & \text{در غیر این صورت} \end{cases}$$

واضح است که $L_{\mathcal{D}}(f) = 0$. در ادامه ثابت می‌کنیم با احتمال حداقل $\frac{1}{7}$ داریم، $L_{\mathcal{D}}(A(S)) \geq \frac{1}{8}$:
$L_{\mathcal{D}}(A(S))$ یک متغیر تصادفی است که مقدار آن در بازه $[0,1]$ است. این متغیر تصادفی را با Y نشان می‌دهیم. همچنین قرار می‌دهیم، $p = \mathbb{P}r\{Y \geq \frac{1}{8}\}$. داریم:

$$\mathbb{E}[Y] = \mathbb{E}[Y|Y \geq \frac{1}{8}]\mathbb{P}r\{Y \geq \frac{1}{8}\} + \mathbb{E}[Y|Y < \frac{1}{8}]\mathbb{P}r\{Y < \frac{1}{8}\} \quad (۴۳)$$

$$= p\,\mathbb{E}[Y|Y \geq \frac{1}{8}] + (1-p)\,\mathbb{E}[Y|Y < \frac{1}{8}] \quad (۴۴)$$

$$\leq p + (1-p)\,\frac{1}{8} = \frac{7}{8}p + \frac{1}{8} \quad (۴۵)$$

$$\Rightarrow \frac{7}{8}p + \frac{1}{8} \geq \mathbb{E}[Y] \geq \frac{1}{4} \quad (۴۶)$$

$$\Rightarrow \quad p \geq \frac{1}{7} \quad (۴۷)$$

□