مسئلهٔ پیش‌بینی وضعیت هوا و یک ایدهٔ الگوریتمی پرکاربرد

عباس محرابیان

دانشگاه بریتیش کلمبیا

abbas.mehrabian@gmail.com

۲۷ بهمن ۱۳۹۴

فرض کنید می‌خواهیم برای T روز متوالی پیش‌بینی کنیم که آیا باران می‌بارد یا خیر، بدون این که هیچ‌گونه دانشی در زمینهٔ هواشناسی داشته باشیم. در این حالت راهی بهتر از این نداریم که هر روز یک سکه بیندازیم، و با توجه به شیر یا خط آمدن آن، پیش‌بینی‌مان را انجام دهیم. برای این که مسئله را جالب‌تر کنیم، فرض می‌کنیم n «کارشناس» هواشناسی وجود دارند که هر روز صبح پیش‌بینی‌شان را به ما می‌گویند، و ما می‌توانیم از این اطلاعات استفاده کنیم. در عین حال هیچ تضمینی وجود ندارد که این کارشناسان واقعاً کارشان درست باشد؛ ولی ما هدفمان اینست که الگوریتمی طراحی کنیم که مطمئن باشیم اگر دست کم یکی از کارشناسان کارش را بلد باشد، ما هم کمابیش به خوبی او پیش‌بینی کنیم. دقت کنید که ما در ابتدا هیچ اطلاعاتی راجع به برتری کارشناسان به هم نداریم، به علاوه کاملاً ممکن است که کارشناسی در آغاز خوب پیش‌بینی کند ولی به تدریج کارش بد بشود و کارشناس دیگری از او بهتر بشود. الگوریتم ما باید در چنین وضعیتی هم خوب عمل کند. در این مقاله الگوریتمی ارائه می‌کنیم که تعداد اشتباهاتش حداکثر $\sqrt{T\ln(n)}$ ۲ تا از دقیق‌ترین کارشناس بیشتر است.[۱]

این الگوریتم، که کرانی که ارائه می‌دهد در نگاه اول شگفت‌آور است، بر یک ایدهٔ ساده و شهودی استوار است: به کارشناسان «اعتبار» نسبت می‌دهیم، هرگاه کارشناسی درست پیش‌بینی کرد اعتبارش افزایش پیدا می‌کند، و بالعکس. نحوهٔ کاهش/افزایش اعتبار هم، ضرب با/تقسیم بر یک عدد ثابت است. این ایده کاربردهای گسترده‌ای در علوم کامپیوتر، از جمله در یادگیری ماشین، بهینه‌سازی، و نظریهٔ بازی‌ها دارد و هدف اصلی این مقاله در حقیقت تبیین این ایده به کمک مثال پیش‌بینی وضعیت هواست.

اگر می‌خواهید بیشتر در مورد این ایده و کاربردهایش بدانید مقالهٔ [۱] را ببینید، که همان‌طور که از عنوانش پیداست،

[۱] اگر بخواهیم دقیق‌تر صحبت کنیم، چون الگوریتمی که ارائه می‌کنیم تصادفی است، باید بگوییم امیدریاضی تعداد اشتباهات الگوریتم ما حداکثر $\sqrt{T\ln(n)}$ ۲ تا از دقیق‌ترین کارشناس بیشتر است.

این ایده را یک «متاالگوریتم» نامیده است که هم خودش و هم تحلیلش ساده هستند و در عین حال بسیار پرکاربرد است. باید به این متاالگوریتم مثل یک ابزار پایه‌ای و کلی نگاه کرد که هر کسی که می‌خواهد الگوریتم یاد بگیرد باید آن را بداند (مثلاً شبیه «جستجوی دودویی» ولی پیچیده‌تر). در این مقاله کاربردهای متنوعی از آن ارائه شده است، از جمله الگوریتم وینو (در یادگیری ماشین) برای یادگیری جداساز خطی[۲]، حل تقریبی بازی‌های صفر-مجموع[۳]، حل تقریبی مسئلهٔ بیشینه کردن شارّ[۴]، دادن ضریب تقریبی برای دستهٔ وسیعی از مسائل ان‌پی-سخت، الگوریتم Ada Boost در یادگیری ماشین، و در بهینه‌سازی در لحظه[۵]. همچنین در انتهای این مقاله یک نسخهٔ «ماتریسی» از این ایده ارائه شده و کاربرد آن در حل تقریبی برنامه‌ریزی‌های نیمه‌مثبت معین[۶] بیان شده است.

مقاله‌ای که در دست دارید در حقیقت ترجمه‌ای است از جزوهٔ یک کلاس درس «جلوه‌هایی از علوم کامپیوتر نظری» مربوط به سال ۲۰۱۲ در دانشگاه ای‌پی‌اف‌ال. استاد درس، الکساندر مدری[۷]، برخی از پیشرفت‌های جدید و بسیار جالب در زمینهٔ علوم کامپیوتر نظری—که خیلی از آن‌ها هنوز در کتاب‌های درسی ارائه نشده‌اند— را مرور کرده است. جزوات این درس در اینترنت موجود است[۸] و نگاه کردن به آن‌ها را به علاقمندان به این شاخه توصیه می‌کنم.

برگردیم به مسئلهٔ پیش‌بینی وضعیت هوا. در این مقاله پنج الگوریتم معرفی می‌کنیم که یکی از دیگری بهترند. ابتدا با الگوریتم‌های قطعی[۹] شروع می‌کنیم و سپس به الگوریتم‌های تصادفی[۱۰] روی می‌آوریم.

نمادگذاری برای عدد طبیعی T، مجموعهٔ $\{۱, ۲, \ldots, T\}$ را با $[T]$ نشان می‌دهیم. برای $t \in [T]$، اگر روز tاُم باران بارید قرار می‌دهیم $X(t) = ۱$، وگرنه قرار می‌دهیم $X(t) = ۰$. برای $t \in [T]$ و $i \in [n]$، پیش‌بینی کارشناس iاُم در روز tاُم را با $x_i(t)$ نشان می‌دهیم. توجه کنید که $x_i(t) \in \{۰, ۱\}$.

۱ الگوریتم تنصیف

صورت مسئله را یادآوری می‌کنیم: برای هر $t \in [T]$، صبح روز tاُم بیدار می‌شویم و مقادیر $x_۱(t), x_۲(t), \ldots, x_n(t)$ را می‌بینیم، و پیش‌بینی می‌کنیم که باران می‌بارد یا خیر. در پایان روز، مقدار واقعی $X(t)$ روشن می‌شود و می‌فهمیم اشتباه کرده‌ایم یا خیر. می‌خواهیم تعداد کل اشتباهاتمان را کمینه کنیم.

برای شروع، در این بخش حالتی را بررسی می‌کنیم که از قبل می‌دانیم دست‌کم یک کارشناس بدون خطا وجود دارد. شاید اولین ایده‌ای که به ذهن می‌رسد این باشد که همواره مطابق نظر اکثریت کارشناسان پیش‌بینی کنیم. ولی این ایدهٔ خوبی نیست، چرا که اگر یک کارشناس همیشه درست پیش‌بینی کند و $n - ۱$ کارشناس دیگر همیشه اشتباه کنند، ما در همهٔ روزها اشتباه می‌کنیم. ولی به راحتی می‌توان این روش را اصلاح کرد: کافی است اصلاً به نظر کارشناسانی که (حتی

---

[۲] Winnow algorithm for learning a linear classifier
[۳] approximately solving zero-sum games
[۴] approximating multicommodity flow problem
[۵] online optimization
[۶] approximately solving semi-definite programs
[۷] Aleksander Madry
[۸] http://thl.epfl.ch/gems/
[۹] deterministic
[۱۰] randomized

یک بار) اشتباه کرده‌اند توجه نکنیم. الگوریتم زیر را الگوریتم تنصیف[11] می‌نامیم.

۱. یک مجموعهٔ S نگه می‌داریم و در اول کار قرار می‌دهیم $S = [n]$. این مجموعه در هر لحظه شامل کارشناسانی است که تا این لحظه هیچ اشتباهی نداشته‌اند.

۲. در هر روز t،

(آ) پیش‌بینی‌مان را براساس نظر اکثریت کارشناسان در S انجام می‌دهیم (اگر تعداد آن‌هایی که می‌گویند باران می‌بارد با آن‌هایی که می‌گویند باران نمی‌بارد برابر بود، به دلخواه تصمیم می‌گیریم).

(ب) در پایان روز که مقدار $X(t)$ معلوم شد، کارشناسانی را که امروز اشتباه کرده‌اند از S حذف می‌کنیم.

طبق فرضی که کردیم مبنی بر این که دست‌کم یک کارشناس بدون خطا وجود دارد، مجموعهٔ S همواره ناتهی باقی خواهد ماند. گزارهٔ زیر کارآیی این الگوریتم را تحلیل می‌کند.

گزارهٔ ۱. اگر کارشناس بدون خطایی وجود داشته باشد، تعداد اشتباهات الگوریتم بالا از $\log_2 n$ بیشتر نیست.

اثبات. هر روزی که الگوریتم تنصیف اشتباه می‌کند، همان شب نصف یا بیشتر از نصف اعضای S را حذف می‌کنیم. چون S در ابتدا n عضو دارد و در انتها دست‌کم یک عضو دارد، تعداد اشتباهات الگوریتم از $\log_2 n$ بیشتر نیست. □

جالب است که کران گزارهٔ بالا از تعداد روزها یعنی T مستقل است و تنها به تعداد کارشناسان یعنی n بستگی دارد. می‌توان نشان داد که اساساً الگوریتمی قطعی وجود ندارد که تعداد خطاهایش به طور تضمینی از $\log_2 n$ کم‌تر باشد.

۲ یک الگوریتم اعتباری

در بخش قبل فرض کردیم کارشناسی بدون اشتباه وجود دارد. در این بخش این فرض غیرواقع‌گرایانه را حذف می‌کنیم. فرض کنید بهترین کارشناس m^* اشتباه می‌کند (و البته مقدار m^* بر الگوریتم پوشیده است) و ما می‌خواهیم تعداد خطاهای الگوریتممان کران بالایی برحسب m^* داشته باشد.

اولین ایده شاید این باشد که الگوریتم تنصیف را به صورت زیر تغییر بدهیم: مشابه قبل، مجموعهٔ S را اول کار برابر مجموعهٔ همهٔ کارشناسان قرار می‌دهیم، و هر شب کارشناسانی که اشتباه کرده‌اند را از S حذف می‌کنیم. مشکل اینست که ممکن است S پس از مدتی تهی بشود. در این صورت، S را مجدداً برابر مجموعهٔ همهٔ کارشناسان قرار می‌دهیم و این کار را مقداردهی مجدد[12] مجموعهٔ S می‌نامیم.

گزارهٔ ۲. تعداد اشتباهات الگوریتم بالا از $(m^* + ۱)(\log_2 n + ۱)$ کم‌تر است.

[11] halving algorithm
[12] reset

اثبات. اول، دقت کنید که تعداد مقداردهی‌های مجدد مجموعهٔ Sاز m^* بیشتر نیست. علت اینست که بهترین کارشناس m^* بار اشتباه می‌کند، و هر بار که S را مقداردهی مجدد می‌کنیم، معنی‌اش اینست که این کارشناس دست‌کم یک اشتباه جدید انجام داده است.

دوم، دقت کنید که در فاصلهٔ بین دو مقداردهی مجدد، الگوریتم حداکثر $\log_2 n + 1$ اشتباه کرده است. علت اینست که بلافاصله بعد از هر مقداردهی مجدد، S دارای n عضو است، و بعد از هر اشتباه، دست‌کم نصف اعضای آن حذف می‌شوند. بنابراین پس از حداکثر $\log_2 n$ اشتباه، S تک‌عضوی می‌شود، و سپس یک اشتباه دیگر لازم است تا تهی بشود.

سوم، دقت کنید که بعد از آخرین مقداردهی مجدد S، الگوریتم حداکثر $\log_2 n$ اشتباه دیگر انجام داده است، چرا که در پایان الگوریتم، S دست‌کم یک عضوی است. با در نظر گرفتن سه نکتهٔ بالا می‌بینیم که تعداد کل اشتباهات الگوریتم از $m^*(1 + \log_2 n) + \log_2 n$ بیشتر نیست. □

گرچه الگوریتم بالا این خوبی را دارد که تعداد اشتباهاتش تنها به تعداد کارشناسان و اشتباهات بهترین کارشناس بستگی دارد (و از تعداد روزها مستقل است)، همچنان جا برای بهتر شدن دارد. ایرادی که به آن وارد است، اینست که وقتی S را مقداردهی مجدد می‌کند، عملاً از صفر شروع می‌کند و همهٔ اطلاعاتی را که در روزهای قبلی دربارهٔ کیفیت کارشناسان به دست آورده دور می‌ریزد.

شاید تقسیم‌بندی مطلق کارشناسان به دو دستهٔ خوب و بد، قابل اعتماد و غیرقابل اعتماد، درست نباشد، و بهتر این باشد که کارشناسان را خاکستری ببینیم. برای این که این ایده را به صورت الگوریتمی پیاده‌سازی کنیم، می‌توانیم به کارشناسان «اعتبار» نسبت دهیم. هر چه کارشناسی اعتبارش بیشتر باشد، یعنی اعتماد بیشتری به پیش‌بینی‌اش داریم. اول کار، چون چیزی دربارهٔ کارشناسان نمی‌دانیم، اعتبار همهٔ آن‌ها را برابر عدد ثابتی (مثلاً ۱) قرار می‌دهیم. در آغاز هر روز کارشناسان را دو دسته می‌کنیم: آن‌هایی که می‌گویند باران می‌بارد، و آن‌هایی که می‌گویند باران نمی‌بارد. اگر مجموع اعتبارات دستهٔ اول بیشتر بود ما هم پیش‌بینی می‌کنیم که باران می‌بارد، و بالعکس. در پایان روز، اعتبار کارشناسانی را که اشتباه کرده‌اند نصف می‌کنیم.

حالا کد الگوریتم را ارائه می‌کنیم. فرض کنید $w_1(t), \ldots, w_n(t)$ نشان‌دهندهٔ اعتبارات کارشناسان در ابتدای روز tاُم باشد. برای $J \subseteq [n]$ تعریف می‌کنیم $w_J(t) = \sum_{j \in J} w_j(t)$.

$w_i(1) \leftarrow 1$ for all $i \in [n]$
for $t = 1, 2, \ldots, T$ **do**
  $J \leftarrow \{i : x_i(t) = 1\}$
  if $\sum_{j \in J} w_j(t) > \sum_{i \notin J} w_i(t)$ **then**
    Predict "It rains on day t."
  else
    Predict "It does not rain on day t."

```
    end if
    Observe X(t)
    for i = 1, 2, ..., n do
        if x_i(t) ≠ X(t) then
            w_i(t+1) ← w_i(t)/2
        else
            w_i(t+1) ← w_i(t)
        end if
    end for
end for
```

گزارهٔ ۳. تعداد اشتباهات الگوریتم بالا از $۳ \times (m^* + \log_۲ n)$ کمتر است.

اثبات. تعریف کنید $W(t) = w_۱(t) + \cdots + w_n(t)$. دقت کنید که $W(۱) = n$ و مقدار W در طول زمان هیچ‌وقت زیاد نمی‌شود. نکتهٔ کلیدی اثبات، تحلیل تغییر مقدار W در روزهایی است که الگوریتم اشتباه پیش‌بینی کرده است. ادعا می‌کنیم اگر الگوریتم روز kاُم اشتباه کند، آن‌گاه

$$W(k+۱) \le \frac{۳}{۴} W(k) \qquad (۱)$$

برای اثبات ادعا، فرض کنیم $I \subseteq [n]$ مجموعهٔ کارشناسانی باشند که روز kاُم اشتباه کرده‌اند، و $C = [n] \setminus I$. از آن‌جا که الگوریتم در این روز مطابق رأی اعضای I عمل کرده است، داریم

$$w_I(k) \ge w_C(k)$$

از طرف دیگر، $w_I(k) + w_C(k) = W(k)$، بنابراین

$$w_I(k) \ge W(k)/۲.$$

در انتهای روز، اعتبار همهٔ کارشناسان حاضر در I نصف می‌شود:

$$w_I(k+۱) = w_I(k)/۲, \qquad w_C(k+۱) = w_C(k).$$

بنابراین،

$$W(k) - W(k+۱) = [w_I(k) - w_I(k+۱)] + [w_C(k) - w_C(k+۱)] = w_I(k)/۲ \ge W(k)/۴,$$

و (۱) ثابت می‌شود.

اگر تعداد کل اشتباهات الگوریتم را m بنامیم، نتیجه می‌گیریم

$$W(T+۱) \leq \left(\frac{۳}{۴}\right)^m W(۱) = \left(\frac{۳}{۴}\right)^m \times n \qquad (۲)$$

از طرف دیگر، فرض کنید j بهترین کارشناس باشد. طبق الگوریتم، هر بار که او اشتباه می‌کند اعتبارش نصف می‌شود. بنابراین

$$w_j(T+۱) = \left(\frac{۱}{۲}\right)^{m^*}.$$

چون $w_j(T+۱) \leq W(T+۱)$، از ترکیب نامساوی بالا و (۲) می‌رسیم به

$$\left(\frac{۱}{۲}\right)^{m^*} = w_j(T+۱) \leq W(T+۱) \leq \left(\frac{۳}{۴}\right)^m \times n.$$

اگر از دو طرف لگاریتم در مبنای دو بگیریم، می‌رسیم به

$$-m^* \leq m \log_۲(۳/۴) + \log_۲ n,$$

که نتیجه می‌دهد

$$m \leq \frac{m^* + \log_۲ n}{\log_۲(۴/۳)} < ۳ \times (m^* + \log_۲ n)$$

و حکم اثبات می‌شود. □

برای nهای بزرگ، این الگوریتم به مراتب از الگوریتمی که در ابتدای همین بخش ارائه کردیم بهتر است، چرا که عبارت $\log_۲ n$ به صورت «جمعی» و نه «ضربی» در آن ظاهر شده است. سؤالی که مطرح می‌شود اینست که ضریب ۳ را چقدر می‌توان کوچک کرد. می‌توان نشان داد که با الگوریتم‌های قطعی (غیراحتمالی)، این ضریب را نمی‌توان از ۲ کوچک‌تر کرد.

۳ الگوریتم تنصیف تصادفی

. در ادامهٔ مقاله به الگوریتم‌های تصادفی روی می‌آوریم که نتایج بهتری به ارمغان می‌آورند. تعداد اشتباهات یک الگوریتم تصادفی خودش یک متغیر تصادفی است و ما در این‌جا می‌خواهیم «امیدریاضی»[۱۳] این متغیر را کمینه کنیم.

در این بخش دوباره فرض می‌کنیم کارشناسی وجود دارد که هیچ اشتباهی نمی‌کند، و در بخش بعدی این فرض را حذف می‌کنیم. الگوریتم تنصیف را بدین طریق تغییر می‌دهیم: در هر روز، به جای این که مطابق رأی اکثریت کارشناسان عمل کنیم، مطابق رأی یک کارشناس تصادفی عمل می‌کنیم. به صورت دقیق‌تر، الگوریتم به شرح زیر عمل می‌کند: یک مجموعهٔ S از کارشناسان «قابل اعتماد» داریم که در ابتدای کار شامل همهٔ کارشناسان می‌شود. در ابتدای هر روز، یک کارشناس را از S به صورت تصادفی انتخاب می‌کنیم و پیش‌بینی‌مان را مطابق نظر او انجام می‌دهیم. در پایان روز، کارشناسانی را که اشتباه پیش‌بینی کرده‌اند از S حذف می‌کنیم.

[۱۳] expected value

گزارهٔ ۴. امیدریاضی تعداد اشتباهات الگوریتم بالا از $\ln n$ بیشتر نیست.

اثبات. فرض کنیم m تعداد اشتباهات الگوریتم باشد، که یک متغیر تصادفی است. قرار می‌دهیم $s_۰ = n$ و s_t را اندازهٔ مجموعهٔ S در پایان روز tاُم تعریف می‌کنیم. دقت کنید که $s_۰, s_۱, \ldots, s_t$ عدد هستند و متغیر تصادفی نیستند! در ابتدای روز tاُم، اندازهٔ S برابر $s_{t-۱}$ است، پس دقیقاً $s_{t-۱} - s_t$ کارشناس در این روز اشتباه پیش‌بینی می‌کنند. چون الگوریتم ما مطابق پیش‌بینی یک کارشناس تصادفی از S عمل می‌کند، احتمال این که الگوریتم در این روز اشتباه کند برابر است با $(s_{t-۱} - s_t)/s_{t-۱}$. به دلیل خطی بودن امیدریاضی داریم

$$\mathbb{E}[m] = \sum_{t\in[T]} \frac{s_{t-۱} - s_t}{s_{t-۱}} = T - \sum_{t\in[T]} \frac{s_t}{s_{t-۱}}.$$

چون دست‌کم یک کارشناس بدون اشتباه وجود دارد، $s_T \geq ۱$. بنابراین از نامساوی حسابی-هندسی داریم

$$\frac{۱}{T}\sum_{t\in[T]} \frac{s_t}{s_{t-۱}} \geq \sqrt[T]{\prod_{t\in[T]} \frac{s_t}{s_{t-۱}}} = \sqrt[T]{\frac{s_T}{s_۰}} \geq \sqrt[T]{\frac{۱}{n}} = e^{-\ln n/T}.$$

برای هر عدد حقیقی x داریم $e^x \geq ۱ + x$ بنابراین $e^{-\ln n/T} \geq ۱ - \ln n/T$. نتیجه می‌گیریم

$$\mathbb{E}[m] = T - \sum_{t\in[T]} \frac{s_t}{s_{t-۱}} \leq T - T\left(۱ - \frac{\ln n}{T}\right) = \ln n,$$

و حکم ثابت می‌شود. □

یادآوری می‌کنیم که $\ln n \approx ۰/۷ \log_۲ n$ بنابراین عملکرد این الگوریتم از نسخهٔ غیرتصادفی آن بهتر است. البته، دقت کنید که الگوریتم غیرتصادفی تعداد خطاها را به صورت «تضمینی» کم‌تر از $\log_۲ n$ نگه می‌داشت، حال آن که الگوریتم تصادفی تعداد خطاها را به صورت «متوسط» کم‌تر از $\ln n$ نگه می‌دارد.

۴ الگوریتم اعتباری تصادفی

در این بخش حالتی را در نظر می‌گیریم که بهترین کارشناس m^* اشتباه می‌کند. الگوریتمی ارائه می‌کنیم که بسیار شبیه الگوریتم اعتباری بخش ۲ است، با این تفاوت که رأی‌گیری از اکثریت را با رأی‌گیری «تصادفی» جایگزین می‌کنیم (شبیه بخش قبلی). به علاوه، یک پارامتر ε هم به الگوریتم اضافه می‌کنیم، و اگر کارشناسی اشتباه کرد، اعتبارش را در $\varepsilon - ۱$ ضرب می‌کنیم (در بخش ۲، داشتیم $\varepsilon = ۱/۲$).

حالا کد الگوریتم را ارائه می‌کنیم. یادآوری می‌کنیم که $w_i(t)$ نشان‌دهندهٔ اعتبار کارشناس iاُم در ابتدای روز tاُم است، و $W(t) = \sum_{i\in[n]} w_i(t)$.

$w_i(1) \leftarrow 1$ for all $i \in [n]$
for $t = 1, 2, \ldots, T$ **do**
  $J \leftarrow \{i : x_i(t) = 1\}$, $C \leftarrow [n] \setminus J$

With probability $\frac{w_J(t)}{W(t)}$ predict "It rains on day t."
With probability $\frac{w_C(t)}{W(t)}$ predict "It does not rain on day t."
Observe $X(t)$
for $i = 1, 2, \ldots, n$ **do**
if $x_i(t) \neq X(t)$ **then**
$w_i(t+1) \leftarrow (1-\varepsilon) w_i(t)$
else
$w_i(t+1) \leftarrow w_i(t)$
end if
end for
end for

گزارهٔ ۵. اگر $\varepsilon \in (۰, \frac{۲}{۳}]$، امیدریاضی تعداد خطاهای الگوریتم بالا از $(۱+\varepsilon)m^* + \frac{\ln n}{\varepsilon}$ بیشتر نیست.

برای اثبات این گزاره از نامساوی زیر استفاده خواهیم کرد، که اثبات آن به خواننده واگذار می‌شود:

$$-x - x^۲ \leq \ln(۱-x) \leq -x \qquad \forall x \leq ۲/۳. \tag{۳}$$

یک راه اثبات این نامساوی، استفاده از مشتق‌گیری و مقعر بودن تابع لگاریتم است، ولی برای این که از درستی آن مطمئن شوید کافی است نمودار توابع موردنظر را به کمک نرم‌افزار ریاضی مورد علاقه‌تان رسم کنید، به عنوان مثال در گوگل عبارت `ln(1-x)+x+x^2` را جستجو کنید.

اثبات گزارهٔ ۵. فرض کنیم m تعداد اشتباهات الگوریتم باشد، و فرض کنیم $F(t)$ مجموعهٔ کارشناسانی باشد که در روز tاُم اشتباه پیش‌بینی کرده‌اند، و تعریف می‌کنیم

$$f(t) = \sum_{i \in F(t)} w_i(t)/W(t).$$

احتمال این که الگوریتم در روز tاُم اشتباه پیش‌بینی کند برابر است با $f(t)$، پس داریم $\mathbb{E}[m] = \sum_{t \in [T]} f(t)$.

از یک طرف، بهترین کارشناس m^* اشتباه می‌کند پس اعتبارش در پایان روز Tاُم برابر است با $(۱-\varepsilon)^{m^*}$. بنابراین

$$W(T+۱) \geq (۱-\varepsilon)^{m^*}. \tag{۴}$$

از طرف دیگر، طبق نحوهٔ تغییر دادن اعتبارها، برای هر روز $t \in [T]$ داریم

$$W(t+۱) = \sum_{i \in F(t)} w_i(t)(۱-\varepsilon) + \sum_{i \notin F(t)} w_i(t) = W(t) - \varepsilon \sum_{i \in F(t)} w_i(t) = W(t)(۱ - \varepsilon f(t)).$$

چون $W(۱) = n$ پس

$$W(T+۱) = n \prod_{t\in[T]} (۱ - \varepsilon f(t)) \tag{۵}$$

با داشتن فرمول‌های (۴) و (۵)، بقیهٔ اثبات صرفاً انجام محاسبات است. این فرمول‌ها ضربی هستند و معمولاً در چنین مواقعی گرفتن لگاریتم کار را ساده‌تر می‌کند. با لگاریتم گرفتن می‌رسیم به

$$m^* \ln(۱ - \varepsilon) \leq \ln W(T+۱) \leq \ln n + \sum_{t\in[T]} \ln(۱ - \varepsilon f(t)).$$

حالا $\ln W(T+۱)$ را حذف می‌کنیم، سپس از هر دو طرف نامساوی (۳) استفاده می‌کنیم و می‌رسیم به

$$m^*(-\varepsilon - \varepsilon^۲) \leq \ln n - \sum_{t\in[T]} \varepsilon f(t) = \ln n - \varepsilon \mathbb{E}[m],$$

پس $\mathbb{E}[m] \leq (۱+\varepsilon)m^* + \ln n/\varepsilon$ و حکم ثابت می‌شود. □

الگوریتم بالا یک پارامتر ε دارد که با تغییر دادنش کران بالایی که برای تعداد خطاهایش داریم تغییر می‌کند. مثلاً اگر قرار دهیم

$$\varepsilon = \sqrt{\ln n/T},$$

می‌رسیم به

$$\mathbb{E}[m^*] - m \leq \varepsilon m^* + \ln n/\varepsilon \leq \varepsilon T + \ln n/\varepsilon = ۲\sqrt{T \ln n},$$

یعنی امیدریاضی تعداد خطاهای الگوریتم حداکثر $۲\sqrt{T \ln n}$ تا از تعداد خطاهای بهترین کارشناس بیشتر است. این همان نتیجه‌ای است که در مقدمه بیان کرده بودیم. در قضیهٔ ۴.۱ در مقالهٔ [۱] به کمک روش‌های احتمالاتی اثبات شده است که این نتیجه را از نظر مرتبه‌ای نمی‌توان بهتر کرد (ولی ممکن است بتوان ضریب ۲ را با عدد ثابت کوچک‌تری جایگزین نمود). به عبارت دیگر، عدد مثبت c وجود دارد که به ازای هر الگوریتمی، می‌توان طوری وضعیت هوا و پیش‌بینی کارشناسان را مقداردهی کرد، که امیدریاضی تعداد اشتباهات الگوریتم دست‌کم $c\sqrt{T \ln n}$ تا از بهترین کارشناس بیشتر باشد.

۵ یک تعمیم

در این بخش تعمیمی از الگوریتم بخش قبلی ارائه می‌کنیم. فرض کنیم n کارشناس اقتصادی داریم و در ابتدای هر روز، هر یک از آنان یک طرح اقتصادی ارائه می‌کنند. ما باید یکی از کارشناسان را انتخاب کنیم و طرح پیشنهادی او را انجام دهیم (الگوریتم انتخاب می‌تواند تصادفی هم باشد). در پایان روز، مشخص می‌شود که هر یک از طرح‌ها چقدر سود یا ضرر داشته است. حال می‌خواهیم به کمک ایدهٔ الگوریتم بخش قبلی، الگوریتم مناسبی برای انتخاب کارشناسان ارائه کنیم، که امیدریاضی سودی که در نهایت عایدمان می‌شود از سودی که با دنبال کردن بهترین کارشناس عایدمان می‌شود خیلی کم‌تر نباشد (طبیعتاً مشکل اینست که ما نمی‌دانیم بهترین کارشناس کدام است).

حال صورت مسئله را به صورت دقیق‌تر بیان می‌کنیم. فرض کنیم ρ یک عدد حقیقی داده شده است و کارشناسان از ۱ تا n شماره‌گذاری شده‌اند.

۱. برای هر $t \in [T]$، در ابتدای روز tاُم الگوریتم ما یک توزیع احتمال مثل $(p_1(t), p_2(t), \ldots, p_n(t))$ در نظر می‌گیرد و کارشناس iاُم را به احتمال $p_i(t)$ انتخاب می‌کند و طرح او را دنبال می‌کند.

۲. در پایان روز، برای کارشناس iاُم عدد $\ell_i(t) \in [-\rho, \rho]$ بر الگوریتم روشن می‌شود که میزان «ضرری» است که دنبال کردن طرح iاُم به همراه دارد. دقت کنید که $\ell_i(t)$ می‌تواند منفی باشد، که به معنای اینست که طرح iاُم سودآور بوده است. (چون در بخش‌های قبلی می‌خواستیم تعداد اشتباهات را کمینه کنیم و برای این که ساده‌تر ارتباط با بخش‌های قبلی را نشان دهیم، در این بخش هم به جای «سود» با «ضرر» کار می‌کنیم.)

۳. امیدریاضی ضرر الگوریتم ما در روز tاُم، طبق تعریف برابر است با $\sum_{i\in[n]} p_i(t)\ell_i(t)$.

توجه کنید که در حالتی که $\ell_i(t) \in \{0, 1\}$، دقیقاً به مسئلهٔ پیش‌بینی وضعیت هوا که در بخش‌های قبلی مطرح شد برمی‌گردیم، و $\ell_i(t) = 1$ اگر و تنها اگر کارشناس iاُم در روز tاُم اشتباه پیش‌بینی کرده باشد. و در این صورت امیدریاضی مجموع ضرر الگوریتم دقیقاً برابر می‌شود با امیدریاضی تعداد اشتباهات الگوریتم.

الگوریتم بخش قبلی را می‌توان به صورت طبیعی به حالتی که در این بخش بررسی می‌کنیم، یعنی $\ell_i(t) \in [-\rho, \rho]$ تعمیم داد، و تحلیل الگوریتم جدید هم مشابه قبل است. الگوریتم به کارشناس iاُم یک اعتبار w_i نسبت می‌دهد و در طول زمان اعتبارات را تغییر می‌دهد. یک پارامتر $\varepsilon \in (0, 2/3]$ هم مشابه قبل داریم. به صورت دقیق‌تر، الگوریتم به صورت زیر کار می‌کند:

۱. در ابتدا قرار می‌دهیم $w_1(1) = \cdots = w_n(1) = 1$.

۲. در ابتدای روز tاُم قرار می‌دهیم $W(t) = \sum_{i\in[n]} w_i(t)$، و $p_i(t) = w_i(t)/W(t)$، و کارشناس iاُم را به احتمال $p_i(t)$ انتخاب می‌کنیم.

۳. در پایان روز tاُم که مقادیر $\ell_i(t)$ روشن شدند، برای همهٔ iها قرار می‌دهیم

$$w_i(t+1) = \left(1 - \frac{\varepsilon}{\rho}\ell_i(t)\right) w_i(t).$$

دقت کنید که چون $\ell_i(t) \in [-\rho, \rho]$، هر روز اعتبار هر کارشناس در عددی بین $1-\varepsilon$ و $1+\varepsilon$ ضرب می‌شود. به علاوه، هر چقدر کارشناس در این روز بدتر عمل کرده باشد، مقدار $\ell_i(t)$ بزرگ‌تر است، پس اعتبار کارشناس کم‌تر می‌شود. برعکس، هر چقدر کارشناس در این روز خوب عمل کرده باشد، مقدار $\ell_i(t)$ کوچک‌تر است، پس اعتبار کارشناس زیاد می‌شود.

این الگوریتم، در مقالهٔ [۱] به اسم multiplicative weights update algorithm نام‌گذاری شده است، و کاربردهای گوناگونی از آن در بخش سوم این مقاله ارائه شده است.

گزارهٔ ۶. برای هر کارشناس مانند j، پس از T روز، امیدریاضی ضرر الگوریتم بالا از

$$\rho \ln n/\epsilon + \sum_{t\in[T]} \ell_j(t) + \varepsilon \sum_{t\in[T]} |\ell_j(t)|$$

بیشتر نیست.

توجه کنید که به طور خاص این گزاره در مورد بهترین کارشناس هم درست است. به علاوه عبارت $\sum_{t\in[T]} \ell_j(t)$ دقیقاً ضرری است که از دنبال کردن کارشناس jاُم در کل T روز عایدمان می‌شود.

اثبات. فرض کنیم $\ell(t)$ امیدریاضی ضرر الگوریتم در روز tاُم باشد. بنابراین،

$$\ell(t) = \sum_{i\in[n]} \frac{w_i(t)}{W(t)} \times \ell_i(t),$$

و امیدریاضی مجموع ضرر الگوریتم در کل T روز برابر است با $\sum_{t\in[T]} \ell(t)$.

با توجه به نحوهٔ به‌روزسازی اعتبارات، برای هر $t \in [T]$ داریم

$$\begin{aligned} W(t+۱) &= \sum_{i\in[n]} w_i(t+۱) = \sum_{i\in[n]} \left(۱ - \frac{\varepsilon}{\rho}\ell_i(t)\right) w_i(t) = \sum_{i\in[n]} w_i(t) - \sum_{i\in[n]} \frac{\varepsilon}{\rho}\ell_i(t) w_i(t) \\ &= W(t) - \frac{\varepsilon}{\rho}\ell(t)W(t) = W(t)\left(۱ - \frac{\varepsilon}{\rho}\ell(t)\right). \end{aligned}$$

با استفاده از استقرای ریاضی و با توجه به این که $W(۱) = n$ می‌رسیم به

$$W(T+۱) = W(۱) \prod_{t\in[T]} \left(۱ - \frac{\varepsilon}{\rho}\ell(t)\right) = n \prod_{t\in[T]} \left(۱ - \frac{\varepsilon}{\rho}\ell(t)\right).$$

دقت کنید که اعتبار کارشناس j در پایان روز Tاُم برابر است با $\prod_{t\in[T]} \left(۱ - \frac{\varepsilon}{\rho}\ell_j(t)\right)$، در نتیجه با توجه به این که اعتبارات کارشناسان اعدادی مثبت هستند، داریم

$$\prod_{t\in[T]} \left(۱ - \frac{\varepsilon}{\rho}\ell_j(t)\right) = w_j(T+۱) \le W(T+۱) = n \prod_{t\in[T]} \left(۱ - \frac{\varepsilon}{\rho}\ell(t)\right).$$

ادامهٔ کار صرفاً انجام محاسبات و ساده کردن نامساوی بالاست. با لگاریتم گرفتن از دو طرف می‌رسیم به

$$\sum_{t\in[T]} \ln\left(۱ - \frac{\varepsilon}{\rho}\ell_j(t)\right) \le \ln n + \sum_{t\in[T]} \ln\left(۱ - \frac{\varepsilon}{\rho}\ell(t)\right).$$

با استفاده از نامساوی (۳) داریم

$$\sum_{t\in[T]}\left(-\frac{\varepsilon}{\rho}\ell_j(t)-\frac{\varepsilon^{۲}}{\rho^{۲}}\ell_j^{۲}(t)\right)\leq \ln n-\sum_{t\in[T]}\frac{\varepsilon}{\rho}\times\ell(t).$$

با ساده‌سازی نامساوی بالا، می‌رسیم به

$$\sum_{t\in[T]}\ell(t)\leq\frac{\rho}{\varepsilon}\times\ln n+\sum_{t\in[T]}\ell_j(t)+\varepsilon\sum_{t\in[T]}\frac{۱}{\rho}\ell_j^{۲}(t)$$

چون $\rho\leq|\ell_j(t)|$ پس $\frac{۱}{\rho}\ell_j^{۲}(t)\leq|\ell_j(t)|$ و لذا

$$\sum_{t\in[T]}\ell(t)\leq\frac{\rho}{\varepsilon}\times\ln n+\sum_{t\in[T]}\ell_j(t)+\varepsilon\sum_{t\in[T]}|\ell_j(t)|,$$

و حکم ثابت می‌شود. □

مراجع

[1] Arora S., Hazan E., Kale S. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8:121–164, 2012. Available from `http://theoryofcomputing.org/articles/v008a006/`