خلاصه های میان ترم یادگیری ماشین

امیر پورمند

۸ اردیبهشت ۱۴۰۰

۱ جلسه ۱

تابع f همون تابعی هست که میخوایم یادبگیریم. goal یا g خروجی الگوریتم ما هستش. H همون مجموعه توابع هستش. میگن مدل یادگیری شامل دو چیزه

۱. مجموعه فرضیه

۲. الگوریتم یادگیری

۳. تابع خطا

سومی رو خودم اضافه کردم! چون به نظرم مهمه!

الگوریتم یادگیری پرسپترون چیه؟ خب مشخصه مجموعه فرضیه اس توابع خطی ان. الگوریتم یادگیری اش

$$w(t+1) = w(t) + y_n x_n \quad (۱)$$

هست. تابع خطاش هم تعداد نقاط اشتباه کلاس بندی شده است! البته ما نقاط اشتباه کلاس بندی شده را با فرمول بالا اپدیت میکنیم. بهش میگن PLA .

۲ جلسه ۲

خب آیا واقعا یادگیری ممکنه؟ باید احتمالاتی به قضیه نگاه کنیم که بگیم بله. مسئله رو معادل مسئله تخمین گر میکنند که آیا تخمین گر میتونه با دقت خوبی اون پارامتر جمعیت رو مدل کنه یا نه؟

میگن بله میشه منتهی یه احتمالی داریم. احتمالش هم میشه همون باند هافدینگ و vc و بقیه علما.

باند هافدینگ میگه احتمال این که دو تا تخمین و پارامتر از هم بیشتر از فلان مقدار فاصله داشته باشند چقدره؟

$$P(|\mu - \nu| > \epsilon) \leq 2e^{-2\epsilon^2 N} \quad (۲)$$

در واقع جمله بالا احتمالا درسته و مساوی اند probably approximately correct یا PAC

حالا چه ربطی به لرنینگ داشت؟ اون قرمز ها رو فرض بگیر تابع های غلط و اون سبز ها تابع های درست. میخوایم احتمال این رو حدس بزنیم که تابع ما درست باشه! و این احتماله با کم کردن باند زیاد میشه! اما اما این h عه یه تابع فیکسه و در واقع ما احتمال این که یک تابع فرضیه خوب باشه رو گفتیم. حالا تو یه روش خیلی ناشیانه میان میگن ما M تا تابع توی فضای فرضیه داریم و هر کدوم ممکن هستند خوب باشند یا نباشند که باند هافدینگ تعمیم پیدا میکنه اینجوری و میشه

$$P(|E_{in}(h) - E_{out}(h)| > \epsilon) \leq 2Me^{-2\epsilon^2 N} \quad (۳)$$

راستی تعریف تابع ساین تو این درس خروجی منفی ۱ و مثبت ۱ داره.

۳ جلسه سوم

الگوریتم پاکت روی بهترین خطای سمپل خروجی برمیگردونه که بهتره! مثل پرسپترون معمولی فقط کانورج شده رو برنمیگردونه.

جواب مسئله رگرسیون خطی یه جواب بسته اس به فرم $w = (X^T X)^{-1} X^T y$

۴ جلسه چهارم

فرمول خطای in-sample

$$E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N} e(h(x_n), f(x_n)) \qquad (۴)$$

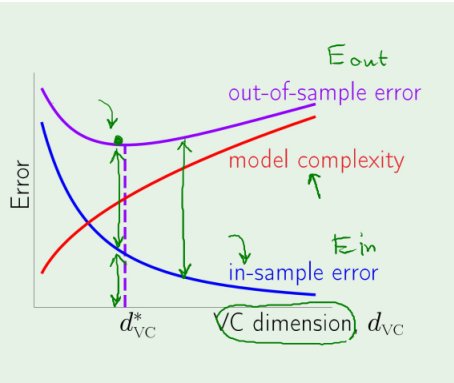
خطای out-of-sample

$$E_{out}(h) = E_x[e(h(x), f(x)] \qquad (۵)$$

در اینجا دو تا مفهوم معادل قدیم داریم که به False Positive میگن False Accept و به False Negative میگن False Reject و تو مثال های مختلف معمولا یکی مهم تره!

تابع هدف ما همیشه میتونه تابع نباشه! یعنی به ازای دو تا ورودی عین هم خروجی مختلف داشته باشه! در این حالت ما $f(x) = E[y|x]$ بدست میاریم!

خب تا حالا میدونیم که یادگیری امکان پذیره. چرا؟ چون به احتمال زیاد که فرمول داره داریم $E_{in}(h) \simeq E_{out}(h)$ و البته داریم $E_{in}(h) \simeq 0$



۵ جلسه پنجم

مفهوم دایکاتمی Dichotomy اینه که اگر n تا نقطه داشته باشیم چند تا برچسب دهی مختلف داریم که فضای فرضیه ما بتونه تولیدش کنه! دی که میدونی معنی دو میده. دایکاتمی یعنی تقسیم به دو بخش. چند حالت میشه تقسیم به دو قسمت کرد.

بهش میگن فرضیه کوچک. اگر تعداد توابع فضای حالت بی نهایت هم باشه تعداد دایکاتمی ها میشه ۲ به توان تعداد نقاط حداکثر!

تعداد حداکثر دایکاتمی ها تو یک فضای مشخص میشه $m_H(N)$. به این تابع رشد هم میگن. این تابع خیلی جالبه یا بریک پوینت داریم که میشه چندجمله ای و خیلی هم مشخصه فرمش یا بریک پوینت نداریم و میشه ۲ به توان N.

پس اگر بتونیم بجای M همون m را بذاریم خیلی خوب میشه! شرطش اینه که تابع m چند جمله ای باشه فقط که ثابت میشه اگر بریک پوینت داشته باشه چند جمله ای هست و یه فرم مشخصی هم داره.

بریک پوینت یه مفهوم گره خورده با بعد vc هست. میگن که اگر بتونیم هر دیتاست از سایز N رو با فضای فرضیه مون دو قسمت کنیم اینجا یعنی بعد vc ما حداقل N عه. ماکزیمم تعداد نقاطی که میشه کامل shatter کرد رو میگن بعد VC. و یکی بیشتر بعد VC میشه بریک پوینت که هست $k_{breakpoint} = d_{vc} - 1$

پس تعریف بریک پوینت میشه حداقل تعداد نقاطی که نتوان همه حالاتش رو shatter کرد.

۶ جلسه ششم

دو تا اثبات داریم یکی این که اگر بریک پوینت داشته باشیم $m_H(N)$ چند جمله ای هست.

یکی دیگه این که میشه به جای M بزرگ گذاشت m.

به یه فرمولی میرسیم که داریم:

$$m_H(N) \leq B(N,K) \leq \sum_{i=0}^{k-1}\binom{N}{k} \leq N^{k-1} = N^d \qquad (۶)$$

که میگن B یعنی تعداد دایکاتمی ها وقتی n تا نقطه و بریک پوینت k رو داریم.

البته با جایگذاری فرمول کمی فرق میکنه و باند وی سی به شکل زیر بدست میاد:

$$P(|E_{in}(h) - E_{out}(h)| > \epsilon) \leq 4m_H(2N)e^{-\frac{1}{8}\epsilon^2 N} \qquad (۷)$$

۷ جلسه هفتم

خب ما باید بتونیم بعد VC رو اثبات کنیم. برای این دو طرف معادله باید ثابت بشه. یک: وجود داره یه روش خاصی برای n نقاط که همه حالاتش رو بشه shatter کرد. این یعنی $d_{vc} \geq N$

دو: هیچ حالتی از N+۱ نقطه رو نمیشه shatter کرد!

$$P(|E_{in}(h)-E_{out}(h)| > \epsilon) \leq 4m_H(2N)e^{-\frac{1}{8}\epsilon^2 N} \simeq N^d e^{-N} \quad (۸)$$

که از این نتیجه میگیریم به عنوان یک قاعده که N باید حداقل ۱۰ برابر بعد VC باشه که مسئله generalize شه.

$$\delta = 4m_H(2N)e^{-\frac{1}{8}\epsilon^2 N} \quad (۹)$$

$$\epsilon = \sqrt{\frac{8}{N} ln \frac{4m_H(N)}{\delta}} \quad (۱۰)$$

که در نظر بگیر که دلتا احتمال اتفاق بد هست. و رابطه خودمون رو هم کلا به این صورت مینویسیم

$$E_{out}(h) \leq E_{in}(h) + \delta \quad (۱۱)$$

۸ جلسه هشتم

ببین یه تریدآف داریم هر چه تابع پیچیده تر باشه تخمین یا approximation بهتره و هر چی تابع ساده تر باشه generalization بهتره. باند بایاس واریانس میخوایم بدست بیاریم منتهی دو تا مسئله هست یک تابع خطا هست که MSE هست و دو مسئله هست که باید رگرسیون باشد.

خب میرسیم به فرمول بایاس واریانس:

$$\begin{aligned} &E_D[(g^{(D)}(x) - f(x))^2] = \\ &E_D[g^{(D)}(x) - \bar{g}(x))^2] + \\ &(\bar{g}(x) - f(x))^2 \end{aligned} \quad (۱۲)$$

یک درس مهم اینه که پیچیدگی مدل برابر هست با پیچیدگی دیتا نه لزوما پیچیدگی تابع!

۹ جلسه نهم

نگاه کردن به داده قبل از درست کردن مدل ممکنه باعث بشه یه جستجویی تو ذهن ما شکل بگیره و یه سری مدل ها محدود بشه به خاطر همین جستجو که بهش میگیم data snooping

برای رگرسیون logistic نیاز به سیگموید داریم چون خاصیت زیر را دارد و برامون مهمه!

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x} = 1 - S(-x) \quad (۱۳)$$

مدل ما در اینجا یک مدل خطیه

$$g(x) = \theta(w^T x) \quad (۱۴)$$

خطای ما باید حساب بشود

$$P(y|x) = \begin{cases} h(x), & y = 1 \\ 1 - h(x), & y = -1 \end{cases} = \theta(y w^T x) \quad (۱۵)$$

و بعد با روش MLE خطا رو بصورت زیر بدست میاریم:

$$E_{in}(w) = \frac{1}{N} \sum_{n=1}^{N} ln(1 + e^{-y_n w^T x_n}) \quad (۱۶)$$

که به این خطا خطای کراس انتروپی میگیم.

الگوریتم یادگیری مون هم باشه مینیمایز کردن همین تابع که با روش گرادیان کاهشی پیش میریم به امید خدا.

پس در هر مرحله گرادیان کل داده ها را بدست میاریم و میریم جلو

$$w(t+1) = w(t) - \eta \nabla E_{in}(w(t)) \quad (۱۷)$$

پس سه تا روش خطی یاد گرفتیم: یکی پرسپترون برای مسئله کلاس بندی که با خطای کلاس بندی پیش میره و الگوریتم یادگیری اش PLA و Pocket هست و دیگری رگرسیون که با خطای MSE پیش میره و الگوریتم یادگیری اش pseudo-inverse است و دیگری رگرسیون لاچیستیک که با خطای کراس انتروپی پیش میره و الگوریتم یادگیری اش گرادیان کاهشی!

۱۰ جلسه دهم

خب روش ما تا حالا GD بوده و بهتر است از گرادیان تصادفی استفاده کنیم که گرادیان بردار وزن یک داده است.

الگوریتم بک پروپگیشن و شبکه عصبی رو از جای دیگه ای باید بخونم خوب نگفته.

۱۱ جلسه یازدهم

حتی اگر بدونیم که تابع ما مثلا درجه ۱۰ هست و داده کافی نداشته باشیم نباید تابع درجه ۱۰ فیت کنیم زیرا نویز را یاد میگیرد و overfit میشود!

حالا نویز تصادفی و نویز قطعی چی هستند؟ یه مساله ای رو در نظر بگیریم که به یه دیتایی مدل مرتبه ۱۰ و مرتبه ۲ فیت کنیم. مشخصه که E_{out} هر کدوم بیشتر باشه بدتره. میایم خطای out-of-sample مرتبه ۱۰ رو منهای ۲ میکنیم.

میبینیم که دو تا عامل تاثیر دارن روی این که این معیار منفی یا مثبت بشه. یکی نویز تصادفی هست که همون نویزی هست که تو دیتا بوده و دیگری نویز قطعی که با نشون دهنده پیچیدگی مدله.

حالا فهمیدن وقتی تعداد دیتا کمه تقریبا همیشه مرتبه ۲ بهتره و هیچ لولی از نویز تصادفی رو نمیتونه تحمل کنه و اگر تعداد دیتا زیاد باشه کم کم تحمل مدل نسبت به نویز تصادفی زیاد میشه که البته با σ^2 نشونش میدن و همین جور رابطه ای رو هم کما بیش با نویز قطعی فهمیدن.

نویز قطعی رو میگن قسمتی از تابع اصلی که فرضیه ما نمیتونه کپچر کنه و در واقع انقدری پیچیده نیست که بتونه بفهمه. تفاوت اصلی اش با نویز تصادفی اینه که به فضای فرضیه بستگی داره. بعضی علما میگن معادل بایاس هست نویز قطعی.

حالا که فهمیدیم اصل قضیه overfitting مال نویزه و نویزها رو هم شناختیم میرسیم به داستان راه حل که علما میگن دو تا روش هست

۱. Validation

۲. Regularization