Лекция 21 . Изчислителната теория на самообучение – продължение

В предишната лекция сме разгледали един теоретичен подход, позволяващ да характеризира трудността на различни задачи за машинно самообучение и възможности на различни типове алгоритми за самообучение. Тази теория търси отговор на такива въпроси като „При какви условия е възможно (или не е) успешно научаване на понятия?" и „При какви условия един конкретен алгоритъм за самообучение със сигурност може да научи понятие успешно?". Ние започнахме да разглеждаме една конкретна схема за анализиране на алгоритми за самообучение, наречена „вероятностно приблизително точно (*PAC*) самообучение". Тази схема ни позволи да идентифицираме класове от понятия, които могат или не могат да бъдат научени от полиномиален брой обучаващи примери. В настоящата лекция ще разширим този подход на случаи, когато пространството на хипотези е безкрайно. Ще въведем и една естествена мярка за сложността на пространства от хипотези, която позволява определяне на граница (оценка) за броя обучаващи примери, необходими за индуктивното самообучение. След това ще разгледаме и друг подход наречен „граница на грешки", който позволява да оценим броя на обучаващи грешки, които ще бъдат направени от една система за самообучение преди тя да намери точната хипотеза.

21.1. Извадковата сложност за безкрайни пространства от хипотези

В предишната лекция ние показахме, че извадковата сложност на PAC самообучение нараства като логаритъм от размера на пространството на хипотези. Макар че формулата (20.2) от предишната лекция е доста полезна, тя има два основни недостатъка, свързани с начина за характеризиране на извадковата сложност в термини на $|H|$. Първо, тя може да доведе до намирането на много широки граници (напомням, че границата върху вероятността за неуспешно научаване на понятия δ може да бъде значително по-голяма от 1 при голям $|H|$). Второ, в случая на безкрайни пространства от хипотези тази формула изобщо не може да бъде използвана!

В настоящия раздел ще разгледаме една друга мярка за сложността на H, наречена размерността на Вапник-Червоненкис (VC размерност или просто *VC(H)*). Както ще видите по-нататък, ние можем да определим границите на извадковата сложност, използвайки *VC(H)* вместо $|H|$. В много случаи границите на извадковата сложност, базирани на *VC(H)*, ще бъдат по-тесни от тези, задавани от формула (20.2). Освен това тези граници позволяват да характеризираме извадковата сложност на множество безкрайни пространства от хипотези при това в доста тесен диапазон.

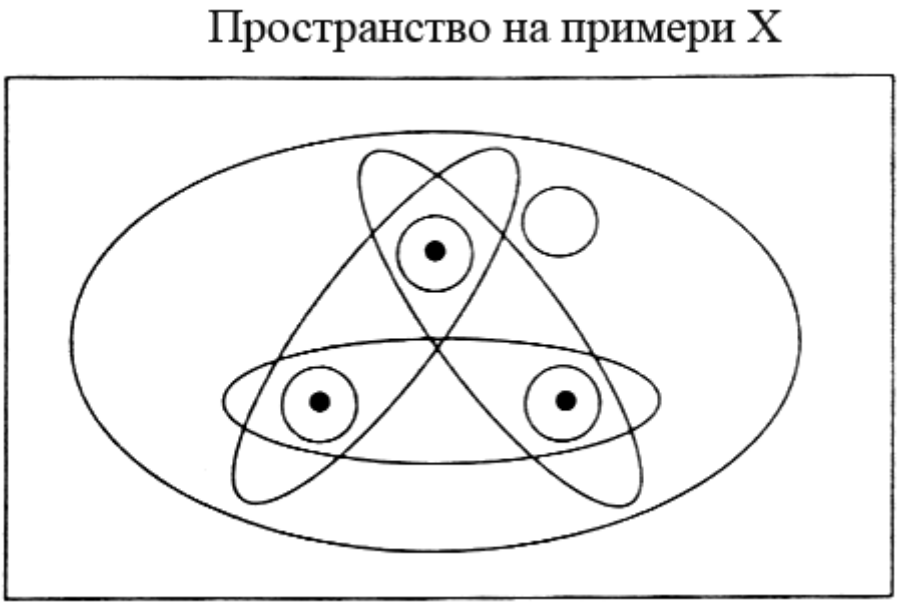
21.1.1. Раздробяване на множество от примери

VC размерност измерва сложността на пространството от хипотези H, но не чрез броя на различни хипотези $|H|$, а чрез броя на различни примери от X, които могат напълно да бъдат разграничени чрез използване на H.

За да направим това понятие по-точно, първо ще дефинираме понятие *раздробяване (shattering)* на едно множество от примери. Да разгледаме някое подмножество от примери $S \subseteq X$. Всяка хипотеза h от H налага определена дихотомия (разполовяване) на S, т.е. h разделя S на два подмножества $\{x \in S|h(x) = 1\}$ и $\{x \in S|h(x) = 0\}$. При зададено множество от примери S, съществуват $2^{|S|}$ възможни дихотомии, макар че H може и да не е в състояние да представи някои от тях. Ще казваме, че H раздробява S, ако всяка възможна дихотомия на S може да бъде представена от някоя хипотеза от H.

Определение. Едно множество от примери S ***се раздробява*** от пространството на хипотези H тогава и само тогава, когато за всяка дихотомия на S съществува някоя хипотеза от H, съвместима с тази дихотомия.

Фиг. 21-1. илюстрира множество S, състоящо от три примера, които са раздробени от пространството на хипотези. Обърнете внимание, че всяка от 2^3 дихотомии на тези три примера е покрита от някоя хипотеза.



Фиг. 21-1. *Множество от три примера, раздробени от осем хипотези.* За всяка възможна дихотомия на примери съществува по една съответна хипотеза.

Ако едно множество от примери не е раздробимо от едно пространство на хипотези, то това означава, че съществува някое понятие, което може да бъде дефинирано върху тези примери, но не може да бъде представено от избраното пространство от хипотези. Следователно, способността на H да раздробява множества от примери е мярка за способността на пространството от хипотези да представя целеви понятия, определени върху тези примери.

21.1.2. Размерността на Вапник-Червоненкис

Способността на раздробява определено множество от примери е тясно свързана с индуктивното пристрастие на пространството от хипотези. Да припомним от първите две лекции, че едно „безпристрастно" пространство от хипотези е този, което е в

състояние да представи всяко възможно понятие (дихотомия), което може да бъде дефинирано върху пространство от примери X. Накратко, едно безпристрастно пространство от хипотези H е този, което раздробява цялото пространство на примери X. Какво става, ако H не може да раздроби X, но може да раздроби някое голямо подмножество S на X? Интуитивно изглежда разумно да приемем, че колко по-големи подмножества от X могат да бъдат раздробени, толкова по-изразително е H. *VC* размерност на H е точно такава мярка.

Определение. ***Размерността на Вапник-Червоненкис*** *VC(H)* на пространство от хипотези H, дефинирано върху пространство от примери X, е размер на най-голямото крайно подмножество на X, което може да бъде раздробено от H. Ако произволно големи крайни множества от X могат да бъдат раздробени от H, то $VC(H) = \infty$.

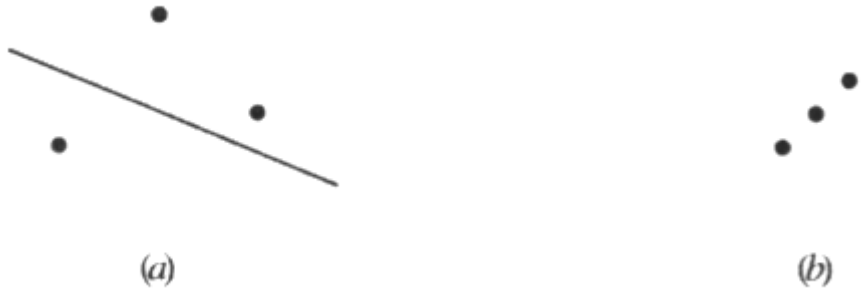
Обърнете внимание, че за всяко крайно H, $VC(H) \leq log_2|H|$. За да се убедим в това, да предположим, че $VC(H) = d$. Тогава H ще изисква 2^d различни хипотези, за да раздроби d примери. Следователно $2^d \leq |H|$ и $d = VC(H) \leq log_2|H|$.

21.1.2.1. Примери

За да развием интуитивното разбиране за *VC(H)*, да разгледаме няколко примера на пространства от хипотези. Да започнем от предположение, че пространството на примери X е множество от всички реални числа $X = \Re$ (например описващо *височината* на хора), а H е множество от интервали от реални числа. С други думи H е множеството от хипотези във вида $a < x < b$, където a и b могат да бъдат всякакви реални константи. Какво е *VC(H)*? За да отговорим на този въпрос трябва да намерим най-голямото подмножество на X, което може да бъде раздробено от H. Да разгледаме едно конкретно подмножество, съдържащо два различни примера, например $S = \{3.1, 5.7\}$. Може ли S да бъде раздробено от H? Да. Например, четирите хипотези $(1 < x < 2)$, $(1 < x < 4)$, $(4 < x < 7)$ и $(1 < x < 7)$ могат да го направят. Заедно те представят всички четири дихотомии на S, покриващи такива случаи, като: нито един от примерите, само по един от примерите и двата примера заедно, съответно. Тъй като ние намерихме едно множество с размер 2, което може да бъде раздробено от H, ние вече знаем, че *VC* размерност на H е не по-малка от 2. Дали едно множество от три произволни примера може да бъде раздробено? Да разгледаме едно множество $S = \{x_0, x_1, x_2\}$, съдържащо три произволни примера. Без загубата на общност можем да предположим, че $x_0 < x_1 < x_2$. Ясно е, че това множество не може да бъде раздробено, тъй като дихотомията, която включва x_0 и x_2, но невключва x_1, не може да бъде представена чрез един единствен затворен интервал. Следователно, няма подмножества S с размер 3, които могат да бъдат раздробени, което означава, че $VC(H) = 2$. Обърнете внимание, че пространството H е безкрайно, но *VC(H)* е крайно.

Сега да разгледаме множеството от примери X, съответстващи на точки в плоскостта x, y (виж Фиг. 21-2). Нека G е множеството от всички линейни повърхнини на решения върху тази плоскост. С други думи, H е пространството от хипотези, отговарящи на един единствен възел от типа на персептрон с два входа (виж лекция 7). Каква е *VC* размерност на това H? Лесно е да се види, че всеки две различни точки от плоскостта могат да бъдат раздробени от H, тъй като ние можем да намерим четири линейни повърхности, които включват нито една, само по една или всички точки заедно. А какво става с множество от три точки? Когато точките не са колинеарни, ние сме в състояние да намерим 2^3 линейни повърхнини, които раздробяват тези точки. Обаче,

три колинеарни точки не могат да бъдат раздробени (по същата причина като и в предишния пример). Каква е *VC(H)* в този случай – 2 или 3? Тя е най-малко 3. Определението за *VC* размерност указва, че ако не можем да намерим *някакво* множество от примери с размер *d*, което може да бъде раздробено, то $VC(H) \geq d$. За да покажем, че $VC(H) < d$, трябва да покажем, че *не съществува нито едно множество* с размер *d*, което може да бъде раздробено. В този пример няма никакви множества с размер *4*, които могат да бъдат раздробени, така че $VC(H) = 3$. В по-общ случай може да бъде показано, че *VC* размерност на линейни повърхнини на решения в едно *r*-мерно пространство (т.е. *VC* размерност на персептрона с *r* входа) е $r + 1$.



Фиг. 21-2. *VC размерност за линейни повърхнини на решения в плоскостта x, y е 3*. (a) Едно множество от три точки, което може да бъде раздробено, използвайки линейни повърхнини на решения. (b) Едно множество от три точки, които не могат да бъдат раздробени.

И на края, да разгледаме пример, в който предполагаме, че всеки пример в *X* се описва чрез конюнкцията от точно три булеви литерала, и да предположим, че всяка хипотеза в *H* се описва чрез конюнкцията от до три булеви литерала. Какво е *VC(H)*? Можем да покажем, че тя е най-малко 3 по следния начин. Нека да представим всеки пример като 3-битов низ, съответстващ на стойността на всеки от тези три литерала l_1, l_2 и l_3. Да разгледаме следното множество от примери:

пример$_1$: 100
пример$_2$: 010
пример$_3$: 001

Това множество от три примера може да бъде раздробено от *H*, тъй като за всяка искана дихотомия може да бъде конструирана по една хипотеза по следния начин: ако дихотомията трябва да не включва *пример*$_i$, добавяме литерал $\neg$ *пример*$_i$ към хипотезата. Например, искаме да включим *пример*$_2$, но да изключим *пример*$_1$ и *пример*$_3$. В този случай можем да използваме хипотезата $\neg l_1 \wedge l_2 \wedge \neg l_3$. Този аргумент може лесно да бъде разширен от три признака на *n* признака. Следователно *VC* размерността на конюнкции от *n* булеви литерала е най-малко *n*. Фактически, тя е точно *n*, макар че доказването на това е по-сложно, тъй като изисква показване, че не съществува нито едно множество от $n + 1$ примера, което може да бъде раздробено.

21.1.3. Извадковата сложност и VC размерността

По-рано ние сме разглеждали въпроса „Колко случайно взети обучаващи примера са достатъчни, за да научим вероятностно приблизително точно някакво целево понятие от *C*?" (т.е. колко примера са достатъчни, за ε-изчерпване на пространството на версии с вероятност $(1 - \delta)$?). Използвайки *VC(H)* като избраната мярка за сложността на *H*, е възможно да изведем един алтернативен отговор на този въпрос, който е аналогичен на по-рано изведената граница от формула (20.2) в предишната лекция. Тази нова граница (виж Blumer et al. 1989) е:

$$m \geq \frac{1}{\varepsilon}(4\log_2(2/\delta) + 8VC(H)\log_2(13/\varepsilon)) \qquad (21.1)$$

Обърнете внимание, че както при границата от ф-ла (20.2) от предишната лекция, броят на необходими обучаващи примери *m* расте логаритмично по отношение на $1/\delta$. Обаче сега той расте не като по-рано линейно, а логаритмично по отношение и на $1/\varepsilon$. Важно е, че термът $ln|H|$ в по-ранната граница е вече заменен с алтернативната мярка за сложността на пространства от хипотези – *VC(H)* (напомням, че $VC(H) \leq log_2|H|$).

Формула (21.1) дава една горна граница на броя обучаващите примери достатъчни за вероятностно приблизително точно научаване на всяко целево понятие от *C* за всяко желано ε и δ. Възможно е да бъде изчислена и долната граница, която се определя от следната теорема (Ehrenfeuch et al. 1989):

Теорема. Долната граница на извадковата сложност. Да разгледаме някой клас от понятия *C* такъв, че $VC(H) \geq 2$, някоя система за самообучение *L* и някои $0 < \varepsilon < 1/8$ и $0 \leq \delta \leq 1/100$. Тогава съществува някое разпределение $\mathcal{D}$ и някое целево понятие в *C* такива, че ако *L* обработва по-малко примери от

$$\max\left[\frac{1}{\varepsilon}\log(1/\delta), \frac{VC(C)-1}{32\varepsilon}\right]$$

то, с вероятност не по-малка от δ, *L* ще изведе хипотеза *h*, имаща $error_{\mathcal{D}}(H) > \varepsilon$.

Теоремата утвърждава, че ако броят на обучаващите примери е прекалено малък, то никаква система за самообучение не може да PAC-научи всички целеви понятия в едно нетривиално пространство *C*. По този начин теоремата предоставя една долна граница на броя на обучаващи примери, *необходими* за успешно самообучение, допълвайки известната от по-рано горната граница, която дава *достатъчния* брой на тези примери. Обърнете внимание, че долната граница се определя от сложността на класа от понятия *C*, докато по-ранната ни граница беше определена от *H*.

Долната граница показва, че горната граница в формула (21.1) е доста тясна. И двете граници са логаритмични по отношение на $1/\delta$ и линейни по отношение на *VC(H)*. Единствената разлика в размера на тези граници е една допълнителна $log(1/\varepsilon)$ зависимост в горната граница.

21.1.4. VC размерност за невронни мрежи

В лекция 6 сме разгледахме невронни мрежи с право разпространение на активация, обучавани с алгоритъма BACKPROPAGATION. Каква е *VC* размерност на подобни мрежи? В настоящия раздел ще представим едни общи резултати, позволяващи да се изчисли *VC* размерност на многонивови ациклични невронни мрежи, използвайки данни за структурата на тези мрежи и *VC* размерност на техните отделни възли. Тази VC размерност може да се използва за ограничаване на броя на обучаващи примери, достатъчни за вероятностно приблизително точно самообучение на една мрежа с право разпространение на активация с желаните стойности на ε и δ.

Да разгледаме една мрежа *G*, състояща от възли, които формират един многослоен насочен ацикличен граф. *Насоченият ацикличен граф* е граф, в който дъгите имат направление (например имат входните и изходни възли), и в който няма насочени цикли. *Многослойният граф* е граф, в който възлите могат да бъдат разделени на слоеве, такива че всички насочени дъги от възли, принадлежащи към ниво l влизат във възли от ниво $l + 1$. Многослойните невронни мрежи с право разпространение на активация, обсаждани в лекция 8, са примери на такива многослойни насочени ациклични графи.

Ще покажем, че можем да получим оценка за *VC* размерността на такива мрежи на базата на тяхната графова структура и на V*C* размерност на базовите възли, формиращи мрежите. За да формализираме това утвърждение, трябва да дефинираме няколко нови термина. Нека n е броят на входове на мрежата *G*, и нека да допуснем, че мрежата има само един изходен възел. Нека всеки вътрешен възел N_i на *G* (т.е. всеки възел, който не е входен) има най-много r входа и имплементира някоя булева функция c_i: $\mathcal{R}^r \rightarrow \{0, 1\}$ от някой клас функции *C*. Например, ако вътрешните възли са персептрони, то *C* ще бъде класът от линейни прагови функции, дефинирани на $\mathcal{R}^r$.

Ние можем да дефинираме *G-композицията на C* да бъде класът на всички функции, които могат да бъдат имплементирани от мрежата *G*, подразбирайки, че отделните възли на *G* реализират функции от класа *C*. Накратко, *G*-композицията на *C* е пространство от хипотези, представими от мрежата *G*.

Следващата теорема ограничава *VC* размерността на *G*-композицията на *C*, използвайки *VC* размерността на *C* и структурата на *G* (Kearns and Vazirani 1994):

Теорема. VC размерност на насочени ациклични многослойни мрежи. Нека *G* е многослоен ацикличен насочен граф с n входни възела и $s \geq 2$ вътрешни възела, всеки от които имат най-много r входа. Нека *C* е класът от понятия, дефиниран върху $\mathcal{R}^r$ с *VC* размерност d, съответстващ на множество функции, които могат да бъдат описани от всеки от s вътрешни възли. Нека C_G е *G*-композицията на *C*, съответстваща на множество функции, които могат да бъдат представени от *G*. Тогава $VC(C_G) \leq 2ds\log(es)$, където e означава основа на натуралния логаритъм.

Обърнете внимание, че границата на *VC* размерността на мрежата *G* нараства линейно по отношение на *VC* размерността d на свои отделни възли и логаритмично по отношение на s – броя на прагови възли в мрежата.

Да предположим, че разглеждаме ациклични многослойни мрежи, чиито отделни възли са персептрони. Напомням, че един персептрон с *r* входа използва линейни повърхнини на решения, за да представи булевите функции върху $\Re^r$. Както беше отбелязано по-рано, *VC* размерност на линейни повърхнини на решения върху $\Re^r$ е *r + 1*. По тази причина, един персептрон с *r* входа има *VC* размерност *r + 1*. Можем да използваме този факт заедно с представената по-горе теорема, за да оценим *VC* размерност на ациклични многонивови мрежи, съдържащи *s* персептрона, всеки от които има по *r* входа:

$$VC(C_G^{perseptrons}) \le 2(r+1)s\log(es)$$

Сега можем да оценим броя *m* на обучаващи примери, достатъчни за научаване (с вероятност не по-малка от *(1 - δ)*) на всяко целево понятие от $C_G^{perseptrons}$ в рамките на грешка ε. Заменяйки горния израз за *VC* размерност на мрежата в формула (21.1), получаваме:

$$m \ge \frac{1}{\varepsilon}(4\log_2(2/\delta) + 8VC(H)\log_2(13/\varepsilon)) \ge \frac{1}{\varepsilon}(4\log_2(2/\delta) + 16(r+1)s\log)es)\log(13/\varepsilon))$$

(21.2)

Този пример с мрежата от персептрони показва, че доказаната по-горе теорема е интересна, тъй като предоставя един общ метод за оценяване на *VC* размерност на многослойни ациклични мрежи от възли, на базата на структурата на мрежата и VC размерността на нейните отделни възли. За съжаление получения по-горе резултат не може директно да се приложи към мрежите, обучени с алгоритъма BACKPROPAGATION по следните две причини. Първо, този резултат е приложим за мрежи от персептрони, а не към мрежи от сигмоидни възли, към които се прилага алгоритъм BACKPROPAGATION. Не зависимо от това, обърнете внимание, че *VC* размерността на сигмоидни възли ще бъде не по-голяма от тези на персептрони, тъй като един сигмоиден възел може да апроксимира един персептрон с произволна точност чрез използване на достатъчно големи тегла. По тази причина, изведената по-горе граница върху *m* ще бъде от същия порядък като и за ациклични многослойни мрежи от сигмоидни възли. Вторият недостатък на по-горния резултат е, че той не може да обясни факта, че алгоритмът BACKPROPAGATION обучава една мрежа, започвайки с теглата, близки до нула, и след това итеративно ги модифицира, до като не бъде намерена някоя приемлива хипотеза. Следователно, алгоритмът BACKPROPAGATION, използван заедно с метода за крос-валидация като критерий за спиране, реализира едно индуктивно пристрастие в полза на мрежи с малки тегла. Това индуктивно пристрастие, ефективно намаляващо VC размерност, не е отчетено при проведения по-горе анализ.

21.2. Модел „Граница на грешки" на самообучение

До сега ние сме се фокусирахме само върху един модел за самообучение – PAC самообучение, обаче изчислителната теория на самообучение разглежда и множество други модели. Например, анализират се такива аспекти на самообучението като как се генерират обучаващи примери (например, пасивно наблюдение на случайни примери или активно създаване на заявки от страна на система за самообучение), наличието на шума в данни (например зашумени данни или данни без наличието на шума), начин за

определяне на успеха (например целевото понятие трябва да бъде научено точно или само вероятностно и приблизително), предположения, правени от системата за самообучение (например, касаещи се разпределение на примери и дали $C \subseteq H$), както и мярката, която се използва за оценяване на системата за самообучение (например броя на използвани обучаващи примери, броя на грешки, общо време за обучение и т.н.).

В този раздел ще разгледаме модела за самообучение наречен *„граница на грешки" (mistake bound)*, в който системата за самообучение се оценява чрез общия брой грешки, които тя прави преди да намери коректната хипотеза. Както и при постановката на задачата за PAC самообучение, ние ще подразбираме, че системата за самообучение получава една последователност от обучаващи примери. Обаче, в този случай ще изискаме от системата при получаване на всеки пример x да предсказва стойността на неговия целеви атрибут $c(x)$ *преди* учителят да й съобщи правилната класификация на примера. Въпросът, който ще се разглежда е: „Колко *грешки* ще направи една система за самообучение в своите предсказвания преди да научи целевото понятие?". Този въпрос е важен за практически условия, когато самообучението трябва да се извършва до като системата се намира в режима на реалната експлоатация, а не в някой тренировъчен „оф-лайн" режим. Например, когато системата трябва да научи да предсказва, коя покупка с кредитна карта трябва да бъде одобрена и коя е резултат от измама, използвайки данни, събрани по време на нейното използване, ние сме заинтересувани в минимизация на общия брой грешки, които системата ще прави преди да намери коректната целева функция. В този случай общият брой грешки може дори да е по-важен от общия брой на обучаващи примери.

Този проблем на самообучение, свързан с границата на грешки, може да бъде изучаван при различни специфични постановки на задачата. Например можем да преброим количество на грешки преди PAC научаване на целевото понятие. В приведените по-долу примери ще разглеждаме следната постановка на задачата – се търси броят на грешки, направени преди *точното* научаване на целевото понятие. Точното научаване на целевото понятие означава намиране на хипотеза, за която $(\forall x)\, h(x) = c(x)$.

21.2.1. Границата на грешки на алгоритъма Find-S

Като илюстрация, да разгледаме отново пространството на хипотези H, състоящо се от конюнкции от до n булеви литерала $l_1, \ldots, l_n$ и техните отрицания (например *богат* $\wedge \neg$ *красив*). Припомням, че алгоритмът Find-S (от лекция 1) инкрементално изчислява максимално специфичната хипотеза, съвместима с обучаващите примери. Една проста реализация на този алгоритъм за пространството от хипотези H може да се опише по следния начин:

Find-S:

- Инициализирай h с най-специфичната хипотеза $l_1 \wedge \neg l_1 \wedge l_2 \wedge \neg l_2 \wedge l_3 \wedge \neg l_3 \ldots \wedge l_n \wedge \neg l_n$
- За всеки положителен обучаващ пример x направи:
  - Изтрий от h всеки литерал, който не удовлетворен от x
- Изведи хипотеза h.

Find-S има сходимост към хипотеза, която не прави никакви грешки при условие, че $C \subseteq H$, и че обучаващите данни не съдържат шум. Алгоритмът започва работата си с най-специфичната хипотеза (която класифицира всеки пример като отрицателен), а след това инкрементално обобщава тази хипотеза по такъв начин, че тя да покрива наблюдавани до сега положителните примери. За използваното в случая представяне на хипотези тази обобщаваща стъпка се състои в изтриването на неудовлетворени литерали.

Можем ли ние да намерим границата на общия брой грешки, които Find-S ще направи преди точно да научи целевото понятие *c*? Отговорът е „да". За да се види това, да припомня, че, първо, ако $c \in H$, то Find-S никога няма да класифицира погрешно нито един отрицателен пример като положителен. Причината за това е, че текущата хипотеза *h* винаги е толкова специфична, както и целевото понятие *c*. По тази причина, за да изчислим броя на грешки, които алгоритъм може да направи, трябва само да преброим броя на грешки, които Find-S може да на прави, класифицирайки действително положителни примери като отрицателни. Колко такива грешки може да допусне алгоритмът преди да научи *c* точно? Да разгледаме първия положителен пример, срещан от Find-S. Системата за самообучение сигурно ще го класифицира неправилно, тъй като нейната начална хипотеза класифицира всички примери като отрицателни. Обаче, резултатът ще бъде, че половината от *2n* терма на тази първоначална хипотеза ще бъдат елиминирани, оставяйки само *n* терма. При всеки следващ положителен пример, който е класифициран неправилно от текущата хипотеза, най-малко един от останалите *n* терма трябва да бъде елиминиран от хипотезата. Следователно общия брой грешки може да бъде най-много *n + 1*. Този брой грешки отговаря на най-лошия случай, съответстващ на научаване на най-общото възможно понятие $(\forall x)\, c(x) = 1$ и то при най-лошата възможна последователност от примери, при която се изтрива само по един литерал на грешка.

21.2.2 Граница на грешки за разполовяващ алгоритъм

Като втори пример да разгледаме един алгоритъм, който научава чрез обработка на описанието на пространството на версии, инкрементално уточнявайки това пространство при постъпване на всеки нов пример. Алгоритмите CANDIDATE-ELIMINATION и LIST_THEN-ELIMINATE от лекция 2 са примери на такива алгоритми. В този раздел ще изведем границата на грешки за най-лошия случай, които ще направи една система за самообучение, използваща такъв алгоритъм и едно крайно пространство на хипотези *H*. Предполага се, че целевото понятие трябва да бъде научено абсолютно точно.

За да анализираме броя на грешки, направени при самообучение, трябва отначало да укажем точно, как системата за самообучение ще прави свои предсказания при постъпване на някой нов пример *x*. Нека да предположим, че това предсказване се прави чрез отчитане на най-много гласове при гласуване на всички хипотези от текущото пространство на версии (т.е мажоритарното гласуване). Ако повечето от хипотезите от пространството на версии класифицират новия пример като положителен, това предсказване ще се извежда като предсказването на системата за самообучение. В противен случай се извежда отрицателното предсказване.

Тази комбинация от научаване на пространството на версии и използване на мажоритарното гласуване често се нарича HALVING (Разполовяващ) алгоритъм. Какъв е максималния брой грешки, които може да на прави HALVING алгоритъм за произволно крайно пространство от хипотези H преди на научи точно целевото понятие? Забележете, че научаването на целевото понятие „точно" съответства на достигане на едно състояние, при който пространството на версии съдържа само една единствена хипотеза (както винаги, предполагаме, че целевото понятие c се съдържа в H).

За да изведем границата на грешка трябва да отчетем, че разполовяващият алгоритъм може да направи грешка само, ако повечето от хипотези от текущото пространство на версии неправилно класифицират новия пример. В този случай, когато правилната класификация се съобщава на системата за самообучение, пространството на версии се съкращава най-много до половина от своя текущ размер (т.е. само тези хипотези, които са гласували миноритарно ще бъдат запазени). Отчитайки, че всяка грешка съкращава размер на пространството на версии най-много на половина и че началното пространство на версии съдържа само $|H|$ хипотези, максималният брой на грешки, възможни преди пространството на версии да съдържа само една хипотеза е $log_2|H|$. Фактически, може да се покаже, че границата е $\lfloor log_2|H| \rfloor$. Да разгледаме, например, случаят, в който $|H| = 7$. Първата грешка ще намали $|H|$ до най-много 3, а втората грешка – до 1.

Обърнете внимание че $\lfloor log_2|H| \rfloor$ е границата на грешка в най-лошия случай, така че е възможно, че разполовяващият алгоритъм да научи целевото понятие точно без никакви грешки изобщо! Това може да се случи, тъй като дори мажоритарното гласуване е правилно, алгоритмът все едно ще изтрива неправилните миноритарни хипотези. Ако това важи за цялата последователност от обучаващи примери, пространството на версии може да бъде намалено до една единствена хипотеза без да се правят никакви грешки при този процес.

Едно интересно разширение на разполовяващия алгоритъм е да позволим различни хипотези да имат различно тегло при гласуване. В лекция 10 ние разгледахме оптималния Бейсов класификатор, който прави такова претеглено гласуване на хипотези. При оптималния Бейсов класификатор теглото, назначавано на всяка от хипотезите, представлява приближения към апостериорна вероятност, че тази хипотеза описва целевото понятие при известни обучаващи данни. В един от следващите раздели ще разгледаме един друг алгоритъм, наречен WEIGHTED-MAJORITY, който използва схемата за претеглено мажоритарно гласуване.

21.2.3 Оптимална граница на грешки

Приведеният по-горе анализ дава границата на грешки за най-лошия случай за два конкретни алгоритъма: FIND-S и CANDIDATE-ELIMINATION. Интересен е въпросът, а каква е оптималната граница на грешки за един произволен клас от понятия C при предположение, че $H = C$? Под оптималната граница на грешки ще разбираме най-ниската граница на грешки в най-лошия случай върху всички възможни алгоритми за самообучение. За да бъдем по-точни, за всеки алгоритъм за самообучение A и произволно целево понятие c, нека чрез $M_A(c)$ да означим максимум по всички възможни последователности от обучаващи примери на броя на грешки, направени от

A, за да научи точно c. Сега за всеки непразен клас от понятия C, нека $M_A(C) = max_{c \in C} M_A(c)$. Обърнете внимание, че по-рано сме показали, че $M_{FIND\text{-}S}(C) = n + 1$, когато C е класът на понятия, описвани с до n булеви литерала. Ние също така показахме, че $M_{HALVING}(C) \leq log_2(|C|)$ за всеки клас от понятия C.

Определение. Нека C е произволен непразен клас от понятия. ***Оптималната граница на грешка*** за C, $Opt(C)$, е минимумът от $M_A(C)$ върху всички възможни алгоритми за самообучение A:

$$Opt(C) = \min_{A \in Алгоритми\ за\ самообучение} M_A(C)$$

Неформално казано, тази дефиниция заявява, че $Opt(C)$ е броят на грешки, направени от най-добрия алгоритъм за самообучение при научаване на най-тежкото понятие в C и при използване на най-тежката последователност от обучаващите примери. Littlestone (1987) показа, че за всеки клас от понятия C, съществува една интересна зависимост между оптималната граница на грешка за C, границата на разполовяващия алгоритъм и VC размерност на C:

$$VC(C) \leq Opt(C) \leq M_{HALVING}(C) \leq log_2(|C|)$$

Освен това, съществуват класове от понятия, за които четири посочените по-горе неравенства се превръщат в точните равенства. Един от такива класове от понятия е множество-степен C_P на произволно крайно множество от примера X. В този случай $VC(C_P) = |X| = log_2(|C|)$, така че всички четири терма в неравенството трябва да са равни. Littlestone (1987) показа примери на други класове от понятия, за които $VC(C)$ е строго по-малко от $Opt(C)$, и за които $Opt(C)$ е строго по-малко от $M_{HALVING}(C)$.

21.2.4. Алгоритъм за претеглено мажоритарно гласуване

В този раздел ще разгледаме едно обобщение на алгоритъма HALVING, наречено алгоритъм за претеглено мажоритарно гласуване (WEIGHTED-MAJORITY). Този алгоритъм прави предсказвания чрез претегляне на предсказвания на едно множество (ансамбъл) от отделни класификационни алгоритми. Той се самообучава чрез промяна на теглата на предсказвания на алгоритмите от ансамбъла. Такива класификационни алгоритми могат да бъдат алтернативни хипотези от H или алтернативни алгоритми за самообучения, които на свой ред могат да се променят с течение на време. Единственото, което ние ще изискваме от един класификационен алгоритъм, е той да предсказва стойността на целевото понятие на дадения пример. Едно интересно свойство на WEIGHTED-MAJORITY алгоритъм е, че той е в състояние да се пригоди към противоречиви обучаващи данни. Причината за това е, че той не отхвърля една хипотеза, за която е намерено, че е несъвместима с някой обучаващ пример, а само намалява нейното тегло. Второто интересно свойство е, че ние можем да ограничим броя на грешки, правени от този алгоритъм, в термините на броя на грешки, допуснати от най-добрия алгоритъм в ансамбъла.

Алгоритмът за претеглено мажоритарно гласуване започва с назначаване на тегло, равно на 1, на всеки от класификационните алгоритми в ансамбъла, а след това разглежда обучаващите примери. Когато един класификационен алгоритъм сгреши при класифициране на някой нов пример, неговото тегло се намалява чрез умножение върху

определена константа β, $0 \leq \beta \leq 1$. Точното описание на алгоритъма е приведено в Таблица 21-1.

---

a_i означава i-я класификационен алгоритъм от ансамбъла A
w_i означава тегло, асоциирано с a_i

- За всички i инициализирай $w_i \leftarrow 1$
- За всеки обучаващ пример $\{x, c(x)\}$ направи:
  - Инициализирай $q_0 \leftarrow 0$; $q_1 \leftarrow 0$
  - За всеки алгоритъм a_i направи:
    - Ако $a_i(x) = 0$ то $q_0 \leftarrow q_0 + w_i$
    - Ако $a_i(x) = 1$ то $q_1 \leftarrow q_1 + w_i$
  - Ако $q_1 > q_0$ то предскажи $c(x) = 1$
    Ако $q_0 > q_1$ то предскажи $c(x) = 0$
    Ако $q_1 = q_0$ то предскажи по случаен начин 1 или 0 за $c(x)$
  - За всеки алгоритъм a_i в A направи:
    - Ако $a_i(x) \neq c(x)$ то $w_i \leftarrow \beta w_i$

Таблица 21-1. Алгоритъм за претеглено мажоритарно гласуване

Обърнете внимание, че алгоритъм WEIGHTED-MAJORITY е аналогичен на алгоритъм HALVING при $\beta = 0$. От другата страна, ако изберем някаква друга стойност за β, то нито един от алгоритмите в ансамбъла никога няма да бъде елиминиран напълно. Ако един алгоритъм неправилно класифицира един обучаващ пример, той просто получава по-малкото тегло на своя глас за в бъдеще.

Сега ще покажем, че броят на грешки, допускани от алгоритъма за претеглено мажоритарно гласуване, може да бъде ограничен в термините на броя на грешки, направени от най-добрия алгоритъм в ансамбъла.

Теорема. Относителна граница на грешка за алгоритъма WEIGHTED-MAJORITY. Нека D е произволна последователност от обучаващи примери, A – произволно множество от класификационни алгоритми, а k – минималния брой на грешки, направени от някой алгоритъм от A за обучаващата последователност D. Тогава броят на грешки върху D, направени от WEIGHTED-MAJORITY алгоритъм при използване на $\beta = ½$ е най-много

$$2.4(k + \log_2 n)$$

Доказателство. Ще докажем тази теорема чрез сравняване на финалните тегла на най-добрия от алгоритмите със сумата на тегла на всички алгоритми. Нека a_j означава някой алгоритъм от D, който е допуснал оптималния брой k на грешки. Финалното тегло w_j, асоциирано с a_j ще бъде $(½)^k$, тъй като неговото начално тегло е 1 се умножава по ½ за всяка грешка. Сега да разгледаме сума $W = \sum_{i=1}^{n} w_i$ на теглата, асоциирани с всички n алгоритъма в A. В началото W е равно на n. За всяка грешка, направена от WEIGHTED-MAJORITY алгоритъм, W се намалява до най-много $(3/4)W$. Това е така, тъй като алгоритмите, „спечелили” в претеглено мажоритарно гласуване, трябва да имат общо най-малко половина от цялото тегло W, а тъй като предсказването е грешно, то тази част от общото тегло ще намалее на половина от фактора $\beta = ½$. Нека M означава

общия брой на грешки, допуснати от WEIGHTED-MAJORITY алгоритъм за обучаващата последователност *D*. Тогава крайното общо тегло *W* ще е най-много $n(\frac{3}{4})^M$. Тъй като крайното тегло w_j не може да е по-голямо от крайното общо тегло, получаваме:

$$\left(\tfrac{1}{2}\right)^k \leq n\left(\tfrac{3}{4}\right)^M$$

Преподреждайки термове, получаваме:

$$M \leq \frac{(k + \log_2 n)}{-\log_2(\frac{3}{4})} \leq 2.4(k + \log_2 n)$$

което и доказва теоремата.

И така, тази теорема утвърждава, че броят на грешки, направени от алгоритъма за претеглено мажоритарно гласуване никога няма да бъде по-голям от един постоянен фактор, умножен по брой на грешки, направени от най-добрия член на ансамбъла, плюс един терм, който нараства логаритмично по отношение на размера на ансамбъла.

Тази теорема е обобщена от Littlestone и Warmuth (1991), които показали, че за всяко $0 \leq \beta \leq 1$ горната граница е

$$\frac{k \log_2 \frac{1}{\beta} + \log_2 n}{\log_2 \frac{2}{1+\beta}}$$

21.3. Заключение

И така, в последните две лекции сме разгледали някои основни аспекти от изчислителната теория на самообучение, сред които:

- Вероятностното приблизително точно (PAC) самообучение разглежда алгоритми, които научават целеви понятия от определен клас от понятия *C*, използвайки обучаващи примери, взети по случаен начин съгласно едно неизвестно, но фиксирано, вероятностно разпределение. PAC моделът изисква от системата за самообучение да научи вероятностно (с вероятност не по-малка от $(1 - \delta)$) една хипотеза, която е приблизително точна (с грешка ε), при зададени изчислителни ресурси и обучаващи примери, които растат само полиномиално по отношение на $1/\varepsilon$, $1/\delta$, размера на примери и размера на целевото понятие.
- При постановка на задачата в рамките на PAC модела на самообучение, всяка „логична" система за самообучение, използваща едно крайно пространство от хипотези *H*, в което $C \subseteq H$, ще извежда (с вероятност $(1 - \delta)$) хипотеза с грешка ε за вида на целевото понятия, след обработка на *m* случайно взети обучаващи примери, където:

  $$m \geq \frac{1}{\varepsilon}(\ln(1/\delta) + \ln(|H|)$$

  Това задава една граница върху броя на обучаващи примери, достатъчни за успешно

научаване на целевото понятие в рамките на PAC модела на самообучение.

- Едно ограничаващо предположение на PAC модела на самообучение се състои в това, че системата за самообучение трябва да знае предварително някой ограничен клас от понятия *C*, който съдържа целевото понятие, подлежащо на научаване. В отличие от това *агностичният модел* на самообучение разглежда по-обща постановка на задачата, в която не се прави никакви предположения за класа, от който се взима целевото понятие. Вместо това системата за самообучение извежда хипотезата от *H*, която има най-малката (възможно ненулева) грешка върху обучаващите данни. В рамките на този по-малко ограничаващ агностичен модел на самообучение, системата за самообучение гарантирано с вероятност *(1 - δ)* извежда с грешка *ε* възможно най-добрата хипотеза в *H* след обработка на *m* случайно избрани обучаващи примери, където

$$m \geq \frac{1}{2\varepsilon^2}(\ln(1/\delta) + \ln(|H|)$$

- Броят на обучаващи примери, необходими за успешното научаване, е строго зависи от сложността на пространството на хипотези, разглеждано от системата за самообучение. Една полезна мярка за сложността на едно пространство от хипотези *H* е неговата Вапник-Червоненкис размерност – *VC(H)*, която представлява размер на най-голямото подмножество от примери, които могат да бъдат раздробени (т.е. разделени по всички възможни начини) от *H*.
- Една алтернативна горна граница на броя на обучаващи примери, достатъчни за успешно самообучение в рамките на PAC модела, се задава в термини на *VC(H)*:

$$m \geq \frac{1}{\varepsilon}(4\log_2(2/\delta) + 8VC(H)\log_2(13/\varepsilon)$$

като долната граница е:

$$m \geq \max\left[\frac{1}{\varepsilon}\log_2(1/\delta), \frac{VC(C)-1}{32\varepsilon}\right]$$

- Алтернативният модел на самообучение, наречен *моделът на граница на грешки*, се използва за анализиране на броя на обучаващи примери, които една система за самообучение ще класифицира неправилно преди да научи точно целевото понятие. Например, разполовяващият алгоритъм ще направи най-много $\lfloor log_2|H| \rfloor$ грешки, преди точно да научи произволно целево понятие от *H*. За всеки произволен клас от понятия *C* в най-лошия случай този алгоритъм ще направи *Opt(C)* грешки, където:

$$VC(C) \leq Opt(C) \leq log_2(|C|)$$

- Алгоритмът за претеглено мажоритарно гласуване комбинира претеглените гласове на множество предсказващи (класификационни) алгоритми (ансамбъл), за да класифицира нови примери. Той научава тегла за всеки от тези предсказващи алгоритми на базата на грешки, направени върху последователността от обучаващи примери. Интересно е, че броят на грешки, направени от този алгоритъм, може да бъде ограничен в термините на грешки, направени от най-добър предсказващ алгоритъм в ансамбъла.