Obsah

Kapitola 1

Úvod do teórie strojového učenia

Chceme sa naučiť na základe nejakých vstupných dát x predikovať y. Môžeme si to predstaviť tak, že príroda vie poskytovať pozorovania, každé v tvare dvojice (x, y). Dostali sme od nej sadu t pozorovaní, na základe ktorých chceme navrhnúť nejakú funkciu h, ktorá predpovedá y na základe x. Dobrá funkcia je taká, ktorá je schopná *zovšeobecňovať*, teda sa jej "dobré darí" aj na dátach mimo trénovacej množiny. Proces, ktorým h zostrojíme, si môžeme predtaviť ako algoritmus, ktorý berie ako vstupy trénovacie dáta a vráti nám funkciu.

1.1 Matematický model

Z matematického hľadiska, prírodu vieme formalizovať ako pravdepodobnostnú distribúciu P. Množinu všetkých možných x označíme X, množinu možných y označíme Y.

V tejto časti sa nebudeme zaoberať výpočtovou stránkou strojového učenia, od detailov ako časová zložitosť, ..., abstrahujeme. Algoritmus si teda predstavíme iba ako niečo, čo vezme ako vstup trénovacie dáta $(x_1, y_1), \ldots, (x_t, y_t)$ a na výstup vráti funkciu $h : X \to Y$. Túto funkciu budeme volať *hypotéza*. Množinu všetkých možných funkcii, ktoré môže náš algoritmus vrátiť, budeme volať *množina hypotéz* a značiť ho H.

Chyba hypotézy. Ako vyjadriť mieru toho, že sa funkcii "dobre darí"? Spravíme tak pomocou *chybovej funkcie* $\mathrm{err} : Y \times Y \to \mathbb{R}^+$, ktorej význam je nasledovný: $\mathrm{err}(y, y')$ vyjadruje, ako veľmi sa od seba líšia y a y'. Pomocou tejto funkcie vieme odmerať priemernú chybu hypotézy h, ktorú budeme tiež označovať err, nasledovne:

$$\mathrm{err}(h) = \mathop{\mathrm{E}}_{x,y}\left[\mathrm{err}(h(x), y)\right]$$

Pod $\mathrm{E}_{x,y}$ sa rozumie stredná hodnota cez (x, y) z pravdepodobnostnej distribúcie P, teda $(x, y) \sim P$. Pri klasifikácii sa zvykne používať chybová funkcia

$$\mathrm{err}(y, y') = \begin{cases} 0, & \text{ak } y = y' \\ 1, & \text{inak} \end{cases}$$

a potom zrejme

$$\mathop{\mathrm{E}}_{x,y}\left[\mathrm{err}(h(x), y)\right] = \mathop{\mathrm{P}}_{x,y}\left(h(x) \neq y\right).$$

Pri regresii máme viacero možností, bežné voľby sú kvadratická chyba $(y - y')^2$ a absolútna chyba $|y - y'|$.

Chyba algoritmu. Ako vyjadriť chybu celého učiaceho algoritmu? Uvedomme si, že výstup algoritmu je závislý od trénovacích dát $T = \{(x_1, y_1), \ldots, (x_t, y_t)\}$, ktoré dostane. Takže výstupná funkcia je od nich závislá, budeme ju označovať $\hat{h}$. Potom priemerná chyba algoritmu (alebo inak *priemerná chyba priemernej hypotézy*), braná cez všetky možné vzorky trénovacích dát, je rovná

$$\mathop{\mathrm{E}}_{T}\left[\mathrm{err}(\hat{h})\right] = \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[\mathrm{err}(\hat{h}(x), y)\right]\right].$$

Pod E_T sa rozumie stredná hodnota cez všetky možné t-tice trénovacích dát T, brané nezávisle z pravdepodobnostnej distribúcie P.

Trénovacia chyba. Pri vyššie uvedených chybách sme vždy merali vzhľadom na skutočnú distribúciu P. Môže nás ale zaujímať, aká je priemerná chyba hypotézy na trénovacích dátach T. Túto chybu budeme označovať $\mathrm{err}_T(h)$, a vypočítame ju ako

$$\mathrm{err}_T(h) = \mathop{\mathrm{E}}_{x_i,y_i}\left[\mathrm{err}(h(x_i), y_i)\right] = \frac{1}{t} \cdot \sum_{i=1}^{t} \mathrm{err}(h(x_i), y_i).$$

Priemerná trénovacia chyba z pohľadu algoritmu bude

$$\mathop{\mathrm{E}}_{T}\left[\mathrm{err}_T(\hat{h})\right].$$

V nasledujúcom texte budeme vynechávať premenné, cez ktoré prebiehajú stredné hodnoty, všade tam, kde budú zrejmé z kontextu.

1.2 Analýza veľkostí chýb

V tejto časti sa podrobnejšie pozrieme na to, ako závisia vyššie uvedené štatistiky (tj. priemerná testovacia a trénovacia chyba priemernej hypotézy) od veľkosti trénovacej množiny T a od veľkosti množiny hypotéz H.

V celej časti budeme predpokladať, že úloha je regresného charakteru a chyba sa meria ako kvadratická odchýlka, teda

$$\mathrm{err}(y, y') = (y - y')^2.$$

1.2.1 Teoretické limity

Najprv sa ale pozrieme na teoretické limity toho, ako dobrá vôbec môže nejaká funkcia byť. Označme $h^\square$ najlepšiu možnú funkciu, nemusí byť nutne z H. Teda

$$h^\square = \arg\min_h \left(\mathrm{err}(h)\right) = \arg\min_h \left(\mathop{\mathrm{E}}_{x,y}\left[(h(x) - y)^2\right]\right).$$

Jediné obmedzenia kladené na h sú, že je to funkcia: pre každé x musí vrátiť vždy jednu a tú istú hodnotu. Distribúcia P ale nemusí pre dané x vždy vrátiť to isté y: môže byť zašumená, alebo jednoducho x neobsahuje dostatočnú informáciu. Napríklad, ak podľa plochy bytu určujeme jeho cenu, niektoré dva byty môžu mať rovnakú plochu a predsa rôznu cenu. Ako uvidíme, tento nedeterminizmus je jediný dôvod, prečo hypotéza $h^\square$ nemusí mať nulovú chybu.

Chybu ľubovoľnej hypotézy h vieme upraviť nasledovne:

$$\mathrm{err}(h) = \mathop{\mathrm{E}}_{x,y}\left[(h(x) - y)^2\right] \tag{1.1}$$

$$= \mathop{\mathrm{E}}_{x}\left[\mathop{\mathrm{E}}_{y|x}\left[(h(x) - y)^2\right]\right] \tag{1.2}$$

Pozrime sa na vnútornú strednú hodnotu. V nej je x konštanta, a teda aj $h(x) = c$ je konštanta. Aká konštanta minimalizuje danú strednú hodnotu? Nie je ťažké vidieť (napríklad zderivovaním), že minimum sa nadobúda pre $c = \mathrm{E}[y]$. Takže

$$h^{\square}(x) = \mathop{\mathrm{E}}_{y|x}[y],$$

a jeho priemerná chyba je

$$\mathrm{err}(h^{\square}) = \mathop{\mathrm{E}}_{x}\left[\mathop{\mathrm{E}}_{y|x}[(y - \mathrm{E}[y])]\right] = \mathop{\mathrm{E}}_{x}\left[\mathop{\mathrm{Var}}_{y|x}(y)\right].$$

Vidíme teda, že pokiaľ je y jednoznačne určené x-om, tak $h^{\square}$ bude mať nulovú chybu.

1.2.2 Bias-variance tradeoff

V tomto odseku si ukážeme zaujímavý výsledok, ktorý nám za určitých predpokladov umožňuje vyjadriť chyby pomocou iných, jasnejších veličín: tzv. *výchylky* a *rozptylu*. Označme najlepšiu hypotézu z množiny H ako $h^{\star}$, teda

$$h^{\star} = \arg\min_{h}\left(\mathrm{err}(h)\right).$$

Budeme upravovať výraz reprezentujúci priemernú chybu priemernej hypotézy $\hat{h}$.

$$\begin{aligned}
\text{chyba algoritmu} &= \mathop{\mathrm{E}}_{T}\left[\mathrm{err}(\hat{h})\right] && (1.3)\\
&= \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(\hat{h}(x) - y)^2\right]\right] && (1.4)\\
&= \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[\left((\hat{h}(x) - h^{\star}(x)) + (h^{\star}(x) - y)\right)^2\right]\right] && (1.5)
\end{aligned}$$

V tomto momente prichádza netriviálny technický krok, ktorý si vyžaduje dodatočné predpoklady. Tieto technické detaily prenecháme na koniec časti, sústreďme sa na to hlavné.

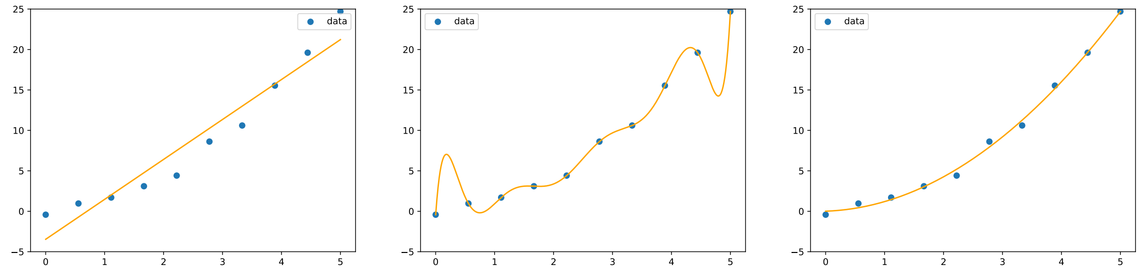
$$\text{chyba algoritmu} = \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(\hat{h}(x) - h^{\star}(x))^2\right]\right] + \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(h^{\star}(x) - y)^2\right]\right]$$

Druhý zo sčítancov sa dá ešte zjednodušiť. Kedže $h^{\star}$ ani y nezávisia od trénovacích dát, môžeme sa zbaviť vonkajšej strednej hodnoty. Dostávame tak výslednú rovnosť

$$\text{chyba algoritmu} = \underbrace{\mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(\hat{h}(x) - h^{\star}(x))^2\right]\right]}_{\text{rozptyl}} + \underbrace{\mathop{\mathrm{E}}_{x,y}\left[(h^{\star}(x) - y)^2\right]}_{\text{výchylka}}.$$

Prvý zo sčítancov budeme volať *rozptyl*. Trénovací algoritmus s malým rozptylom vracia funkcie, ktoré sú blízko optima v množine H. Tým, že mu zväčšíme množinu trénovacích dát, si veľmi neprilepšíme. Naopak, algoritmus s veľkým rozptylom vracia funkcie ďaleko od optima, teoreticky by sme sa teda vedeli k optimu priblížiť tým, že zväčšíme množstvo trénovacích dát.

Druhý zo sčítancov budeme volať *výchylka*. Vyjadruje chybu, ktorá je spôsobená tým, že sa náš algoritmus obmedzil na nejakú konkrétnu množinu hypotéz H. Čím väčšia množina hypotéz, tým menšia výchylka (nakoľko $h^{\star}$ je najlepšia hypotéza v množine H, jej zväčšením si môžeme iba prilepšiť). Zložitejšia množina hypotéz ale ľahšie “napasuje” na ľubovoľné trénovacie dáta. To zvyšuje riziko toho, že výsledná hypotéza bude špecifická pre obdržané dáta, a nebude schopná zovšeobecňovať mimo nich. Je teda potreba väčšieho množstva trénovacích dát.



Obr. 1.1: Podučenie, preučenie, akurát.

Ak máme fixné trénovacie dáta T, pri voľbe množiny hypotéz H sa snažíme nájsť kompromis medzi malým rozptylom a malou výchylkou. Zložité H bude mať malú výchylku ale veľký rozptyl, čo vedie k tzv. *preučeniu*. Jednoduché H bude mať malý rozptyl, ale veľkú výchylku, tzv. *podučenie*.

Na obrázku 1.1 ilustrujeme oba koncepty: úlohou je modelovať kvadratickú funkciu. Ak za množinu hypotéz zvolíme lineárne funkcie, ich chyby sa nebudú veľmi od seba líšiť, ale všetky budú zle. Ak za množinu hypotéz zvolíme polynómy nejakého vysokého stupňa, ľahko nájdeme polynóm prechádzajúci cez trénovacie dáta, avšak mimo nich bude dávať výsledky úplne mimo.

Výchylku vieme upraviť ďalej. Hypotéza $h^\star$ ani y nezávisia od trénovacej množiny T. Z ich pohľadu sú teda testovacie dáta x, y a trénovacie dáta x_i, y_i nerozoznateľné. Takže na meranie chyby $h^\star$ môžeme použiť trénovacie dáta (berúc v úvahu ich náhodný výber):

$$\text{výchylka} = \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x_i,y_i}\left[(h^\star(x_i) - y_i)^2\right]\right] \tag{1.6}$$

$$= \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x_i,y_i}\left[\left((h^\star(x_i) - \hat{h}(x_i)) + (\hat{h}(x_i) - y_i)\right)^2\right]\right] \tag{1.7}$$

Použitím ďalšieho technického kroku dostaneme:

$$\text{výchylka} = \underbrace{\mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x_i,y_i}\left[(h^\star(x_i) - \hat{h}(x_i))^2\right]\right]}_{\text{trénovací rozptyl}} + \underbrace{\mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x_i,y_i}\left[(\hat{h}(x_i) - y_i)^2\right]\right]}_{\text{priemerná trénovacia chyba}} \tag{1.8}$$

Prvý zo sčítancov budeme volať *trénovací rozptyl*. Uvedomme si, že pre ľubovoľné trénovacie dáta T platí

$$\mathrm{err}_T(\hat{h}) \leq \mathrm{err}_T(h^\star),$$

nakoľko $\hat{h}$ je optimálna hypotéza pre danú množinu trénovacích dát. Hypotéza h síce je najlepšia pre H, trénovacie dáta sú ale len malá vzorka z H. Trénovací rozptyl teda môžeme chápať ako mieru toho, ako veľmi reprezentatívnu vzorku trénovacích dát sme dostali. Čím menší je, tým viac reprezentatívna vzorka je.

Druhý zo sčítancov budeme volať *priemerná trénovacia chyba*. Je to priemerná chyba, ktorej sa dopustí výstu z algoritmu $\hat{h}$ na tých istých dátach, pomocou ktorých sme $\hat{h}$ zostrojili.

Platí

$$\text{priemerná trénovacia chyba} \leq \text{výchylka} \leq \text{chyba algoritmu}.$$

Na konkrétnych trénovacích dátach ale nemusí platiť, že trénovacia chyba je menšia ako testovacia chyba: mohli sme si (síce s malou pravdepodobnosťou, ale predsa) vytiahnuť zlé trénovacie dáta, ktoré sa výrazne líšia od skutočných dát.

Na základe dosiaľ uvedeného vieme graficky znázorniť, ako sa zhruba správajú rozptyl, výchylka, trénovací rozptyl a priemerná trénovacia chyba, v závislosti od veľkosti trénovacej množiny (obrázok **??**) a od zložitosti množiny hypotéz (obrázok **??**).

TODO obrázok kriviek učenia, vysvetlenie

Technické detaily. Nakoniec sa vyjadríme k spomínanému technickému kroku. Začneme jeho znením a potom uvedieme jeho predpoklady.

Veta 1.1. *Predpokladajme, že vstupom do hypotéz sú vektory reálnych čísel (tj. $X = \mathbb{R}^n$), cieľom je predpovedať jedno reálne číslo (tj. $Y = \mathbb{R}$), a že pravdepodobnostné rozdelenie P je spojité.*

Nech množina hypotéz H je uzavretá na lineárne kombinácie a na limity (teda ak postupnosť funkcii v H konverguje, jej limita je tiež v H).

Ďalej predpokladajme, že trénovací algoritmus vždy vráti takú funkciu $\hat{h} \in H$, ktorá minimalizuje trénovaciu chybu. Inak zapísané,

$$\hat{h} = \arg\min_{h \in H} \left(\mathop{\mathrm{E}}_{T} [\mathrm{err}_T(h)] \right).$$

Potom platí

$$\mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[\left((\hat{h}(x) - h^\star(x)) + (h^\star(x) - y)\right)^2\right]\right] = \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(\hat{h}(x) - h^\star(x))^2\right]\right] + \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(h^\star(x) - y)^2\right]\right]$$

Poznámka 1.1. Dokazovaná rovnosť je ekvivalentná s nasledovnou, stručnejšou:

$$\mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(\hat{h}(x) - h^\star(x)) \cdot (h^\star(x) - y)\right]\right] = 0.$$

Túto kratšiu verziu získame roznásobením a použitím linearity strednej hodnoty. V dôkaze budeme dokazovať túto rovnosť.

Poznámka 1.2. Všimnite si, že potrebujeme uzavretosť množiny H na limity na to, aby vôbec $\arg\min_{h \in H}(\ldots)$ existovalo. Vo všeobecnosti nemusí existovať taká funkcia, ale môže existovať nekonečná postupnosť funkcii, každá ďalšia lepšia, ako tá predchádzajúca. (Inak povedané, neexistuje minimum, iba infimum.)

Poznámka 1.3. Je namieste otázka, či je $\arg\min_{h \in H}(\ldots)$ dobre definované, teda či je taká funkcia h práve jedna. Za chvíľu uvidíme, že naše predpoklady to zaručujú.

Poznámka 1.4. Veta by sa dala rozšíriť aj na iné množiny X, Y, napríklad keď predpovedaná premenná je vektor ($Y = \mathbb{R}^m$), ... Možno ani P nemusí byť spojité. Pre jednoduchosť argumentu ale budeme uvažovať vetu tak, ako je popísaná vyššie.

Poznámka 1.5. Predpoklady vety sú značne obmedzujúce. Napríklad si uvedomte, že ju nie je možné použiť na klasifikáciu, či dokonca ani na ľubovoľnú ohraničenú regresiu (kde rozumné hodnoty y sú ohraničené). Ale taká je teória.

Pri našom dôkaze využijeme niekoľko vlastností funkcii, ktoré uvádzame v nasledujúcom odseku. Skúsený čitateľ-matematik ho môže preskočiť.

Definícia 1. (Skalárny súčin.) Nech f, g sú funkcie z X do $\mathbb{R}$, z nejakej príjemne sa správajúcej množiny funkcii (tj. rovnomerne spojité, ..., čokoľvek, aby nasledujúce argumenty prešli). Definujeme ich skalárny súčin $\langle \cdot, \cdot \rangle$ ako

$$\begin{aligned} \langle f, g \rangle &= \int f(x) \cdot g(x) \; d\rho x && (1.9) \\ &= \mathop{\mathrm{E}}_{x} [f(x) \cdot g(x)], && (1.10) \end{aligned}$$

kde ρ je hustota pravdepodobnosti distribúcie P. Rozmyslite si, že takto definovaný skalárny súčin má všetky vlastnosti, ktoré sa bežne požadujú od skalárnych súčinov:

- Je symetrický od svojich argumentov, teda $\langle f, g\rangle = \langle g, f\rangle$.
- Je lineárny: $\langle f, g+h\rangle = \langle f, g\rangle + \langle f, h\rangle$ a tiež $\langle k\cdot f, g\rangle = k\cdot\langle f, g\rangle$.
- $\langle f, f\rangle \geq 0$ pre ľubovoľné f, pričom rovnosť nastáva práve vtedy, keď je f konštantne nulové.

Definícia 2. (Kolmosť.) Dve funkcie f, g sú na seba kolmé, ak ich skalárny súčin je 0. Značíme $f \perp g$.

Definícia 3. (Norma.) Podľa skalárneho súčinu definujeme normu funkcie (jej "dĺžku"):

$$\|f\| = \sqrt{\langle f, f\rangle} = \sqrt{\mathop{\mathrm{E}}_{x}\left[f^2(x)\right]}$$

Spĺňa *trojuholníkovú nerovnosť*: pre ľubovoľné funkcie f, g platí

$$\|f\| + \|g\| \geq \|f+g\|\,.$$

Definuje nám teda (euklidovskú) metriku nad funkciami, podľa ktorej definujeme limity a konvergenciu.

Lemma 1.2. *(Pytagorova veta.) Nech $f \perp g$. Potom platí:*

$$\|f\|^2 + \|g\|^2 = \|f+g\|^2$$

Dôkaz. Pozrime sa na pravú stranu. Iba v nej zapíšeme normu ako skalárny súčin a využijeme jeho linearitu a symetriu:

$$\|f+g\|^2 = \langle f+g, f+g\rangle \tag{1.11}$$
$$= \langle f, f\rangle + \langle g, g\rangle + 2\cdot\langle f, g\rangle \tag{1.12}$$

Pretože $f \perp g$, posledný sčítanec je nulový, čím dostávame dokazované tvrdenie. □

Definícia 4. (Projekcia na množinu.) *Projekciu* funkcie f na množinu H budeme označovať f_H a budeme pod ňou rozumieť nasledovný výraz:

$$f_H = \arg\min_{h\in H} d(f, h)$$

Poznámka 1.6. Ako sa už spomínalo, nie je zrejmé, že projekcia je dobre definovaná. Preto v nasledujúcej lemme definujeme f_H trochu iným spôsobom, ako jednu z možno viacerých funkcii, ktoré minimalizujú vzdialenosť k H.

Lemma 1.3. *(Kolmosť projekcie.) Pre ľubovoľnú funkciu $h \in H$ platí $h \perp f - f_H$.*

Dôkaz. Sporom, predpokladajme, že $h \not\perp f - f_H$. Takže $\langle h, f - f_H\rangle \neq 0$. Ukážeme, že potom existuje v H funkcia, ktorá je k funkcii f bližšie, ako funkcia f_H. To bude hľadaný spor s definíciou f_H.

Pozrime sa na všetky funkcie, ktoré ležia na priamke $f_H + \Delta\cdot h$. Tieto funkci sú v množine H, pretože $f_H, h \in H$ a množina H je uzavretá na lineárne kombinácie. Každú z týchto funkcii vieme asociovať s jedným reálnym číslom Δ. Pozrime sa na ich vzdialenosti od funkcie f, vyjadrené ako funkcia od Δ:

$$\mathrm{dist}(\Delta) = d(f, f_H + \Delta\cdot h) \tag{1.13}$$
$$= \langle (f - f_H) + \Delta\cdot h, (f - f_H) + \Delta\cdot h\rangle \tag{1.14}$$
$$= \langle f - f_H, f - f_H\rangle + 2\Delta\cdot\langle h, f - f_H\rangle + \Delta^2\cdot\langle h, h\rangle \tag{1.15}$$

Pozrime sa na deriváciu tejto funkcie. Podľa definície f_H by malo byť $f - f_H$ najkratšie možné, teda pre $\Delta = 0$ by mala funkcia dist nadobúdať minimum, a teda mať tam nulovú deriváciu. Uvidíme, že tomu tak nie je:

$$\begin{aligned} \frac{\partial \text{dist}}{\partial \Delta}(0) &= \lim_{\Delta \to 0} \left(\frac{\text{dist}(\Delta) - \text{dist}(0)}{\Delta} \right) && (1.16) \\ &= \lim_{\Delta \to 0} \left(\frac{2\Delta \cdot \langle h, f - f_H \rangle + \Delta^2 \cdot \langle h, h \rangle}{\Delta} \right) && (1.17) \\ &= 2 \cdot \langle h, f - f_H \rangle && (1.18) \end{aligned}$$

To je nenulové, nakoľko $h \not\perp f - f_H$. Čo je hľadaný spor. $\square$

Lemma 1.4. *Projekcia na množinu H je dobre definovaná, teda vždy existuje nanajvýš jedna funkcia $f_H \in H$, ktorá minimalizuje vzdialenosť k f.*

Dôkaz. Predpokladajme, že také funkcie sú dve, označme ich g, h. Ukážeme, že potom nutne $g = h$.

Podľa predchádzajúcej lemmy platí

$$\begin{aligned} f - g \perp g, \text{ odkiaľ} \langle f - g, g \rangle &= 0 && (1.19) \\ \langle f - h, g \rangle &= 0 && (1.20) \\ \langle f - g, h \rangle &= 0 && (1.21) \\ \langle f - h, h \rangle &= 0 && (1.22) \end{aligned}$$

Z týchto rovností dostaneme

$$\langle g, g \rangle = \langle g, h \rangle = \langle h, g \rangle = \langle h, h \rangle.$$

Nakoniec, pozrime sa na normu funkcie $g - h$:

$$\begin{aligned} \|g - h\| &= \sqrt{\langle g - h, g - h \rangle} && (1.23) \\ &= \sqrt{\langle g, g \rangle - \langle g, h \rangle - \langle h, g \rangle + \langle h, h \rangle} && (1.24) \\ &= 0 && (1.25) \end{aligned}$$

To môže nastať jedine vtedy, keď $g = h$. $\square$

Lemma 1.5. *Hypotéza $h^\star$ je projekciou $h^\square$ na H, teda $h^\star = h^\square_H$.*

Dôkaz. Vychádzajme z definície $h^\star$.

$$\begin{aligned} h^\star &= \arg\min_{h \in H} \mathop{\mathrm{E}}_{x,y} \left[(h(x) - y)^2 \right] && (1.26) \\ &= \arg\min_{h \in H} \mathop{\mathrm{E}}_{x,y} \left[\left((h(x) - h^\square(x)) + (h^\square(x) - y) \right)^2 \right] && (1.27) \\ &= \arg\min_{h \in H} \left(\begin{array}{rl} & \mathrm{E}_{x,y} \left[(h(x) - h^\square(x))^2 \right] \\ + & \mathrm{E}_{x,y} \left[(h^\square(x) - y)^2 \right] \\ +2 \cdot & \mathrm{E}_{x,y} \left[(h(x) - h^\square(x)) \cdot (h^\square(x) - y) \right] \end{array} \right) && (1.28) \end{aligned}$$

Druhý sčítanec je konštanta, teda nám arg min nijak neovplyvňuje. Tretí sčítanec vieme upraviť nasledovne:

$$\begin{aligned} \text{tretí sčítanec} &= \mathop{\mathrm{E}}_{x} \left[\mathop{\mathrm{E}}_{y|x} \left[(h(x) - h^\square(x)) \cdot (h^\square(x) - y) \right] \right] && (1.29) \\ &= \mathop{\mathrm{E}}_{x} \left[(h(x) - h^\square(x)) \cdot \mathop{\mathrm{E}}_{y|x} \left[h^\square(x) - y \right] \right] && (1.30) \\ &= 0 && (1.31) \end{aligned}$$

A teda je to tiež konštanta. Dostávame tak

$$h^\star = \arg\min_{h\in H} \mathop{\mathrm{E}}_{x,y}\left[(h(x) - h^\square(x))^2\right],$$

čo je presne definícia projekcie $h^\square$ na množinu H. $\square$

Vyzbrojení týmito znalosťami, môžeme sa vrhnúť na dôkaz vety 1.1. Pripomeňme si ešte pred tým dokazovanú rovnosť:

$$\mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(\hat{h}(x) - h^\star(x)) \cdot (h^\star(x) - y)\right]\right] = 0.$$

Dôkaz. Ľavú stranu dokazovanej rovnosti vieme prepísať do nasledovného, ekvivalentného tvaru:

$$= \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(\hat{h}(x) - h^\star(x)) \cdot ((h^\star(x) - h^\square(x)) + (h^\square(x) - y))\right]\right]$$

Vieme, že $\varepsilon := h^\square(x) - y$ sa správa pre dané x ako náhodná premenná, ktorá má strednú hodnotu 0 a je nezávislá od ostatných premenných vystupujúcich vo výraze. Z výrazu ju teda môžeme vyhodiť, dostaneme tak

$$= \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x,y}\left[(\hat{h}(x) - h^\star(x)) \cdot (h^\star(x) - h^\square(x))\right]\right]$$

Stačí nám teda dokázať $\hat{h} - h^\star \perp h^\star - h^\square$. To ale vyplýva z lemmy o kolmosti projekcie (1.3). Overíme, že jej podmienky sú splnené: z uzavretosti na lineárne kombinácie platí $\hat{h} - h^\star \in H$, a podľa lemmy 1.5 platí $h^\star = h^\square_H$, odkiaľ $h^\star - h^\square = -(h^\square - h^\square_H)$. Záporné znamienko na kolmosti nič nemení. $\square$

Podobným spôsobom sa dá dokázať aj korektnosť druhého technického kroku. To prenechávame čitateľovi ako cvičenie.

1.2.3 Bias-variance tradeoff, verzia 2.

V literatúre pod názvom *bias-variance tradeoff* vystupuje aj podobný, ale predsa odlišný výsledok, ako bolo uvedené vyššie. Ukážeme a odvodíme si ho.

Veta 1.6. *Nech $y : X \to \mathbb{R}$ je funkcia, ktorú sa snažíme modelovať. Predpokladajme, že sa dá rozložiť na časti: $y = f(x) + \varepsilon$, kde ε hrá rolu šumu: je nezávislý od všetkého a* $\mathrm{E}[\varepsilon] = 0$. *Označíme jeho pravdepodobnostnú distribúciu E.*

Nech výstupom trénovacieho algoritmu je $\hat{f}$. Za chybovú funkciu zvoľme kvadratickú chybu. Chybu algoritmu vieme teda vypočítať nasledovne:

$$\text{chyba algoritmu} = \mathop{\mathrm{E}}_{(x,y)\sim P, T\sim P^t, \varepsilon\sim E}\left[(\hat{f}(x) - y)^2\right].$$

Tvrdíme, že sa dá rozložiť na tri nasledovné časti:

$$\text{chyba algoritmu} = \underbrace{\mathrm{Var}(\hat{f}(x) - f(x))}_{\text{rozptyl}} + \underbrace{(\mathrm{E}[\hat{f}(x)] - \mathrm{E}[f(x)])^2}_{\text{výchylka}^2} + \underbrace{\mathrm{Var}(\varepsilon)}_{\text{šum}}$$

Poznámka 1.7. V poslednej rovnici sme kvôli stručnosti vynechali pri stredných hodnotách a rozptyloch premenné a distribúcie, z ktorých ich berieme. V dôkaze budeme vždy brať všetky premenné z ich príslušných distribúcii.

Poznámka 1.8. Funkcia f hrá v podstate tú istú rolu, čo najlepšia možná hypotéza spomedzi všetkých funkcii (nielen tých v množine hypotéz), $h^\square$.

Poznámka 1.9. V tomto znení bias-variance tradeoff-u názvy *rozptyl* a *výchylka* zodpovedajú príslušným štatistickým/pravdepodobnostným pojmom.

Poznámka 1.10. Na rozdiel od predchádzajúcej verzie bias-variance tradeoff-u, tu nebudeme potrebovať žiadne dodatočné predpoklady od algoritmu ani od jeho množiny hypotéz. (Nemusí teda vracať hypotézu, ktorá je spomedzi hypotéz v H najlepšia na daných trénovacích dátach. Takisto od množiny hypotéz nepožadujeme žiadne vlastnosti.)

Dôkaz. Upravujme pôvodný výraz.

$$\begin{aligned} \text{chyba algoritmu} &= \mathrm{E}\left[(\hat{f}(x)-y)^2\right] && (1.32)\\ &= \mathrm{E}\left[(\hat{f}(x)-f(x)-\varepsilon)^2\right] && (1.33)\\ &= \mathrm{E}\left[(\hat{f}(x)-f(x))^2\right]+\mathrm{E}\left[\varepsilon^2\right]-2\cdot\mathrm{E}\left[\varepsilon\cdot(\hat{f}(x)-f(x))\right] && (1.34)\\ &= \mathrm{E}\left[(\hat{f}(x)-f(x))^2\right]+\mathrm{E}\left[\varepsilon^2\right] && (1.35)\end{aligned}$$

Výraz sme upravili, roznásobili a využili linearitu strednej hodnoty. V poslednom kroku sme použili $\mathrm{E}[ab]=\mathrm{E}[a]\cdot\mathrm{E}[b]$, ktorý platí pre ľubovoľné nezávislé premenné, s $a:=\varepsilon$, $b:=\hat{f}(x)-f(x)$. Zamerajme sa ďalej na prvý sčítanec.

$$\begin{aligned} \text{prvý sčítanec} &= \mathrm{E}\left[(\hat{f}(x)-f(x))^2\right] && (1.36)\\ &= \mathrm{E}[\hat{f}(x)^2]+\mathrm{E}[f(x)^2]-2\cdot\mathrm{E}[\hat{f}(x)\cdot f(x)] && (1.37)\\ &= (\mathrm{Var}(\hat{f}(x))+\mathrm{E}[\hat{f}(x)]^2)+(\mathrm{Var}(f(x))+\mathrm{E}[f(x)]^2)-2\cdot\mathrm{E}[\hat{f}(x)\cdot f(x)] && (1.38)\end{aligned}$$

V poslednom kroku sme využili vzťah $\mathrm{Var}(a)=\mathrm{E}[a^2]-\mathrm{E}[a]^2$. Pokračujme ďalej v úpravách.

$$\begin{aligned} \text{prvý sčítanec} &= \mathrm{Var}(\hat{f}(x))+\mathrm{Var}(f(x))+(\mathrm{E}[\hat{f}(x)]-\mathrm{E}[f(x)])^2\\ &\quad+2\cdot\mathrm{E}[\hat{f}(x)]\cdot\mathrm{E}[f(x)]-2\cdot\mathrm{E}[\hat{f}(x)\cdot f(x)] && (1.39)\\ &= \mathrm{Var}(\hat{f}(x))+\mathrm{Var}(f(x))+(\mathrm{E}[\hat{f}(x)]-\mathrm{E}[f(x)])^2-2\cdot\mathrm{Cov}(\hat{f}(x),f(x)) && (1.40)\\ &= \mathrm{Var}(\hat{f}(x)-f(x))+(\mathrm{E}[\hat{f}(x)]-\mathrm{E}[f(x)])^2 && (1.41)\end{aligned}$$

Využili sme najprv vzťah $\mathrm{Cov}(a,b)=\mathrm{E}[ab]-\mathrm{E}[a]\cdot\mathrm{E}[b]$, a potom $\mathrm{Var}(a-b)=\mathrm{Var}(a)+\mathrm{Var}(b)-2\cdot\mathrm{Cov}(a,b)$. Keď to teda celé dáme do jednej rovnice, dostaneme

$$\text{chyba algoritmu}=\underbrace{\mathrm{Var}(\hat{f}(x)-f(x))}_{\text{rozptyl}}+\underbrace{(\mathrm{E}[\hat{f}(x)]-\mathrm{E}[f(x)])^2}_{\text{výchylka}^2}+\underbrace{\mathrm{Var}(\varepsilon)}_{\text{šum}}$$

□

1.3 Ako sa vysporiadať s preučením/podučením?

V tejto časti sa budeme zaoberať otázkou: “Ako zvoliť vhodne zložitú množinu hypotéz?” Ako sme videli, príliš jednoduché hypotézy vedú k síce malému rozptylu, ale veľkej výchylke, zatiaľ čo príliš zložité hypotézy vedú k malej výchylke, ale veľkému rozptylu.

Predstavme si, že máme na výber z viacerých množín hypotéz, čím ďalej tým zložitejších:

$$H_1\subseteq H_2\subseteq H_3\subseteq\dots$$

Z ktorej množiny hypotéz chceme vybrať?

Pri trénovaní sa snažíme nájsť hypotézu h, ktorá minimalizuje chybu na trénovacích dátach $\mathrm{err}_T(h)$. Táto chyba nám ale nehovorí nič o rozptyle. Ak by sme si graficky znázornili testovacie a trénovacie chyby najlepších hypotéz z jednotlivých množín, vyzeralo by to zhruba ako na obrázku **??**.

TODO obrázok

1.3.1 Regularizácia

V tomto prístupe do minimalizovaného výrazu umelo pridáme člen, ktorý aproximuje rozptyl: pokuta(h), pričom z čím zložitejšej množiny hypotéza h je, tým väčšia pokuta. Takže výstupom algoritmu je

$$\hat{h} = \underset{h \in H_1 \cup H_2 \cup \ldots}{\arg\min} \left(\mathrm{err}_T(h) + \mathrm{pokuta}(h)\right).$$

Uvedomte si, že vrámci jednej množiny H_i ostáva ako najlepšia hypotéza stále tá istá, ako pred zavedením pokuty. V jednej množine sú totiž všetky hypotézy penalizované rovnako, nerobí to teda rozdiel. Penalizácia nám ale umožňuje "férovejšie" porovnávať hypotézy z rôznych množín, nakoľko bez pokuty by na tom boli (neprávom) lepšie zložitejšie hypotézy.

Množiny H_i nemusia byť explicitné, môžu byť implicitne skryté v tom, aký tvar má výraz pokuta(h). Do jednej množiny patria tie hypotézy, ktoré majú rovnakú penalizáciu.

Uvedieme si niekoľko príkladov výrazov, ktoré môžu byť použité ako pokuta. Vo všetkých prípadoch je pokuta je parametrizovaná reálnym parametrom λ hovoriacim, ako veľké pokuty chceme udeľovať. Budeme predpokladať, že celá množina hypotéz, z ktorej vyberáme (tj. $H_1 \cup H_2 \cup \ldots$) je množina lineárnych funkcii $\mathbb{R}^n \to \mathbb{R}$. Hypotézy majú teda tvar

$$h(x) = a_1x_1 + a_2x_2 + \ldots + a_nx_n.$$

- L_2 regularizácia (známa aj ako *ridge regression*). V nej penalizujeme veľké váhy: čím dôležitejší atribút, tým väčšie váhy si môže dovoliť mať.

  $$\mathrm{pokuta}(h) = \lambda \cdot \|(a_1, a_2, \ldots, a_n)\|^2 = \lambda \cdot (a_1^2 + a_2^2 + \ldots + a_n^2)$$

- L_1 regularizácia (známa aj ako *lasso*). Opäť penalizujeme veľké váhy, avšak pokuta je iná.

  $$\mathrm{pokuta}(h) = \lambda \cdot (|a_1| + |a_2| + \ldots + |a_n|)$$

  Táto pokuta "tlačí" nepotrebné atribúty do nuly, čo je výhodné: nulové atribúty vôbec nemusíme uvažovať, čo nám zníži výpočtové nároky. Na druhej strane sa táto pokuta neoptimalizuje ľahko (z optimalizačného hľadiska).

1.3.2 Holdout testing

V tomto prístupe si rozdelíme dostupné dáta na dve časti: trénovaciu množinu a *validačnú množinu*. Pomocou validačnej množiny budeme odhadovať testovacie chyby pre jednotlivé množiny hypotéz, na základe ktorých zistíme, ktorá množina hypotéz je pre náš problém najvhodnejšia. Konkrétnejšie:

1. Trénovaciu množinu použijeme na natrénovanie hypotéz z jednotlivých množín.

2. Ako odhad testovacej chyby jednotlivých hypotéz použijeme ich chybu na validačnej množine. Podľa týchto odhadov zistíme, ktorá množina hypotéz je pre náš problém najvhodnejšia.

3. Použijeme všetky dáta, ktoré máme k dispozícii (tj. z trénovacej aj validačnej množiny), na natrénovanie najlepšej možnej hypotézy. Berieme samozrejme v úvahu iba hypotézy z najlepšej množiny hypotéz. Výsledná hypotéza je výstupom.

V kroku 2 je dôležité, aby bola validačná množina nezávislá od trénovacej. Prečo je to dôležité? Môžeme uvažovať extrémny prípad, keď je validačná množina totožná s trénovacou. Potom ale ako náš "odhad" dostaneme trénovaciu chybu, ktorá rozhodne nie je dobrým odhadom testovacej chyby. Nezávislosť nám teda zaručuje, že odhad získaný na validačnej množine je dobrý.

Poznámka 1.11. Treba podotknúť, že chyba na validačnej množine je iba odhad. Ak by sme mali dostatočne veľa rôznych modelov, z ktorých vyberáme (tj. $H_1, H_2, \ldots$), pre niektorý z nich by sa mohlo stať čistou náhodou, že jeho validačná chyba je nízka, napriek tomu, že jeho skutočná testovacia chyba je vysoká.

Toto je podobné, ako pri overovaní vedeckých hypotéz: napríklad si predstavme 10 hypotéz, každá s 10% šancou, že bude konzistentná s nazbieranými dátami. Potom môžeme očakávať, že jedna z nich bude konzistentná s nazbieranými dátami, napriek tomu, že všetky sú úplne náhodné.

V oboch prípadoch sa to dá samozrejme eliminovať jedným spôsobom: viac dát.

k-fold evaluation. Pri tomto prístupe je dôležité mať dobrý odhad testovacej chyby pre jednotlivé množiny hypotéz. Dát ale môže byť málo, a v takom prípade môže byť odhad nestabilný/nepresný. Môžeme ale experiment zopakovať niekoľkokrát: v každej iterácii teda zvolíme inú trénovaciu a inú validačnú množinu, a dostaneme iný odhad testovacej chyby. Keď tieto odhady spriemerujeme, dostaneme oveľa presnejší odhad, ako keby sme vykonali iba jednu iteráciu.

V tomto konkrétnom prístupe je k iterácii, a množiny sa volia nasledovne: všetky dáta sa rozdelia na k zhruba rovnako veľkých a navzájom nezávislých množín $K_1, K_2, \ldots, K_k$. Následne, v iterácii i sa ako validačné dáta použije množina K_i. Všetko ostatné budú trénovacie dáta.

Testovacia množina. Ak chceme zmerať testovaciu chybu výstupnej hypotézy, musíme si na to rezervovať ďalšiu časť dát: *testovaciu množinu.* Tú nepoužívame ani pri trénovaní, ani pri validácii. Iba úplne na konci celého procesu na nej vypočítame chybu našej hypotézy.

Poznámka 1.12. "Ak sa mi model trénuje príliš dobre, väčšinou to je veľmi zle!"

1.4 Cvičenia

V nasledujúcich dvoch cvičeniach môžete predpokladať, že trénovací algoritmus vždy vráti nejakú funkciu (nemusí byť len jedna) s minimálnou chybou na trénovacích dátach.

1.1. Je rozumné predpokladať (a všade vyššie sme tak činili), že s väčším množstvom trénovacích dát sa nám bude testovacia chyba zmenšovať. Sú ale zostrojiteľné situácie, kedy tomu tak nie je. Nájdite jednu takú situáciu.

Konkrétne, nájdite takú množinu hypotéz H funkcii $\mathbb{R}^n \to \mathbb{R}$ a pravdepodobnostné rozdelenie P, pre ktoré sa nám bude testovacia chyba so zvyšujúcim sa počtom trénovacích chýb *zvyšovať.* Jediná podmienka je kladená na množinu hypotéz: pre každú možnú trénovaciu množinu T musí existovať hypotéza v H, ktorá minimalizuje trénovaciu chybu. (Teda vždy musí existovať minimum, vo všeobecnosti existuje iba infimum.)

1.2. Za určitých podmienok ale skutočne platí, že viac trénovacích dát nám vo veľkom merítku neuškodí. Nech množina hypotéz H je konečná a všetky jej funkcie ($\mathbb{R}^n \to \mathbb{R}$) sú ohraničené. Dokážte, keď $t \to \infty$, tak chyba hypotézy $\hat{h}$ sa bude blížiť k chybe najlepšej možnej hypotézy $h^\star$. Konkrétnejšie, dokážte

$$\lim_{t\to\infty} \mathop{\mathrm{E}}_{T} \left[\operatorname{err}(\hat{h}) - \operatorname{err}(h^\star)\right] = 0.$$

1.3. Dokážte korektnosť druhého technického kroku, v odvodení rozkladu výchylky na trénovací rozptyl a priemernú trénovaciu chybu. Konkrétnejšie, dokážte

$$\mathop{\mathrm{E}}_{T} \left[\mathop{\mathrm{E}}_{x_i, y_i} \left[(h^\star(x_i) - \hat{h}(x_i)) \cdot (\hat{h}(x_i) - y_i) \right] \right] = 0.$$

Predpoklady kladené na množinu hypotéz sú rovnaké: musí byť uzavretá na lineárne kombinácie a na limity.

1.4. Jednou výhodou L_2 regularizácie oproti L_1 regularizácie je, že sa ľahšie minimalizuje výsledný výraz. Ako príklad uvedieme lineárnu regresiu. V nej je hypotéza parametrizovaná stĺpcovým vektorom $\theta = (\theta_1, \ldots, \theta_n)^T$. Výstupom pre vstup $x = (x_1, \ldots, x_n)$ je $x \cdot \theta$.

Označme X maticu, ktorej riadkami sú vstupe jednotlivých trénovacích príkladov. Ďalej nech y je stĺpcový vektor cieľových výstupov na jednotlivých príkladoch. Ako určite vieme, optimálnymi parametrami lineárnej hypotézy je taký stĺpcový vektor θ, ktorý je riešením rovnice

$$X^T X \cdot \theta = X^T y.$$

Dokáže, že keď k minimalizovanej hodnote pridáme pokutu vo forme $\lambda \cdot \|\theta\|^2$, tak sa optimálnymi parametrami stane θ riešiace rovnicu

$$(X^T X + \lambda I) \cdot \theta = X^T y.$$

Rozmyslite si taktiež, že takýto explicitný "vzorec" nie je možné priamočiaro získať pre L_1 regularizáciu.

Kapitola 2

PAC učenie

V tejto kapitole sa budeme zaoberať otázkou toho, ako závisí chyba algoritmu od veľkosti trénovacej množiny. Konkrétne sa budeme zaoberať otázkami ako:

- "Pri danej veľkosti trénovacej množiny t, akú chybu algoritmu môžeme očakávať?"
- "Pri danom t, s akou pravdepodobnosťou nám algoritmus vráti hypotézu, ktorej chyba je menšia ako ε?"

Na základe odpovedí na tieto dve otázky potom budeme schopní zodpovedať nasledovné, príbuzné otázky:

- "Akú veľkú trénovaciu množinu máme zvoliť, aby sme dosiahli dostatočne malú ($\leq \epsilon$) chybu algoritmu?"
- "Aké t máme zvoliť, aby sme s vysokou pravdepodobnosťou ($\geq \delta$) dostali dostatočne dobrú ($\leq \varepsilon$) hypotézu?"

Odtiaľ sa odvíja názov *PAC učenie* (z anglického *probably approximately correct learning*).

Uvedomte si, že obe typy "chýb" sú potrebné, keď sa chceme rozprávať o tom, aký vplyv má veľkosť trénovacej množiny na trénovací algoritmus. Po prvé, ε je potrebné ako miera toho, čo je dostatočne dobrá hypotéza. Po druhé, δ je potrebné, nakoľko vo všeobecnosti nevieme garantovať, že dostaneme dobrú hypotézu: mohli sme si (s malou pravdepodobnosťou) vytiahnuť zlé trénovacie dáta.

2.1 Konečné množiny hypotéz

Zameriame sa zatiaľ iba na konečné množiny hypotéz, a vrámci toho na klasifikačné úlohy, v ktorých je cieľom rozlíšiť medzi reprezentantmi nejakého konceptu od nereprezentantov. Napríklad daný koncept môže byť "písmeno A". Hypotéza dostane na vstupe obrázok 32×32 a má povedať, či tento obrázok vyobrazuje písmeno A alebo nie.

Nech v našom probléme nevystupuje šum: pre dané x je správne y vždy práve jedno, a vždy, keď ho naša hypotéza vráti, tak to bude správna odpoveď.

Nech v našej množine hypotéz je hypotéza f, ktorá vždy vracia správny výstup. Teoreticky teda vieme dosiahnuť nulovú chybovosť. Dá sa na to pozerať aj tak, že jednotlivé hypotézy zodpovedajú rôznym konceptom. Množina hypotéz je potom množina konceptov, spomedzi ktorých spočiatku nevieme rozhodnúť, ktorý je ten správny.

Nakoniec, budeme predpokladať, že algoritmus vždy vráti hypotézu konzistentnú s trénovacími dátami. To znamená, že pre ľubovoľnú trénovaciu množinu T platí $\mathrm{err}_T(\hat{h}) = 0$.

2.1.1 Základné výsledky

Veta 2.1. *Nech je dané $\varepsilon > 0$. Hypotézu nazveme* zlú, *ak jej chybovosť je väčšia ako ε. Potom vieme pomocou počtu trénovacích príkladov t odhadnúť pravdepodobnosť, že nám algoritmus vráti zlú hypotézu, nasledovne:*

$$\operatorname*{P}_T(\mathrm{err}(\hat{h}) > \varepsilon) < |H| \cdot e^{-\varepsilon t}$$

Dôkaz. Zoberme si nejakú zlú hypotézu h. Pravdepodobnosť, že je konzistentná s trénovacími príkladmi ("prejde trénovacou fázou"), je rovná $(1 - \mathrm{err}(h))^t$, čo vieme odhadnúť nasledovne:

$$(1 - \mathrm{err}(h))^t < (1 - \varepsilon)^t \leq e^{-\varepsilon t}$$

Zlých hypotéz je nanajvýš toľko, koľko všetkých hypotéz, teda H. Pravdepodobnosť, že aspoň jedna z nich bude konzistentná s príkladmi, sa dá odhadnúť zhora ako súčet ich pravdepodobností:

$$\operatorname*{P}_T(\text{aspoň jedna zlá}) < |H| \cdot e^{-\varepsilon t}$$

Ak žiadna zlá hypotéza nie je konzistentná s príkladmi, tak výstupom algoritmu nemôže byť zlá hypotéza. Takže platí

$$\operatorname*{P}_T(\mathrm{err}(\hat{h}) > \varepsilon) \leq \operatorname*{P}_T(\text{aspoň jedna zlá}),$$

odkiaľ

$$\operatorname*{P}_T(\mathrm{err}(\hat{h}) > \varepsilon) < |H| \cdot e^{-\varepsilon t}.$$

□

Poznámka 2.1. Vo vyššie uvedenom dôkaze nijakým spôsobom nevystupovali pravdepodobnostné rozdelenie P ani cieľový koncept $f \in H$. Uvedená veta teda platí pre ľubovoľné P a f.

Čo ak nás zaujíma druhá otázka: "Ako závisí chyba algoritmu od počtu trénovacích príkladov?" Pri klasifikačných úlohách je táto otázka úzko spätá s predošlou otázkou, kde sme sa zaujímali o ε a δ.

Veta 2.2. *Platí*

$$\text{chyba algoritmu} \leq \frac{1}{t} \cdot (\ln |H| + \ln t + 1)\,.$$

Dôkaz. Ak nám algoritmus vráti dobrú hypotézu (s chybou nanajvýš ε), vieme jej chybu odhadnúť zhora ako ε. Ak nám algoritmus vráti zlú hypotézu, jej chyba je nanajvýš 1. Z toho dostávame nasledovný horný odhad na celkovú chybu algoritmu:

$$\text{chyba algoritmu} = \operatorname*{E}_T\left[\mathrm{err}(\hat{h})\right] \leq \operatorname*{P}_T(\mathrm{err}(\hat{h}) \leq \varepsilon) \cdot \varepsilon + \operatorname*{P}_T(\mathrm{err}(\hat{h}) > \varepsilon) \cdot 1$$

Pritom pravdepodobnosti na pravej strane vieme odhadnúť zhora: $\mathrm{P}_T(\mathrm{err}(\hat{h}) \leq \varepsilon) \leq 1$, a druhú vieme odhadnúť pomocou vety 2.1. Dostávame tak odhad

$$\text{chyba algoritmu} \leq 1 \cdot \varepsilon + |H| \cdot e^{-\varepsilon t}.$$

My sme si ale mohli zvoliť ε ľubovoľne. Ak teda chceme dostať čo najlepší odhad, nájdeme ε, pre ktoré je výraz na pravej strane čo najmenší. Zderivujme a položme rovné nule:

$$1 - |H| \cdot t \cdot e^{-\varepsilon t} = 0 \tag{2.1}$$

$$\varepsilon = \frac{1}{t} \cdot (\ln |H| + \ln t) \tag{2.2}$$

Odtiaľ dosadením dostaneme požadovaný odhad na chybu algoritmu. □

Na základe týchto dvoch viet vieme sformulovať postačujúce podmienky na t také, aby boli príslušné chyby (ε, δ a chyba algoritmu) dostatočne malé. Sformulujeme a dokážeme jednu z nich.

Dôsledok 2.3. *Množina hypotéz H PAC-naučiteľná: pre každé $\varepsilon > 0$, $\delta > 0$, ľubovoľné pravdepodobnostné rozdelenie P a ľubovoľný cieľový koncept $f \in H$ existuje počet trénovacích príkladov t taký, že platí*

$$\underset{T}{\mathrm{P}}(\mathrm{err}(\hat{h}) \leq \varepsilon) \geq 1 - \delta.$$

Ekvivalentne,

$$\underset{T}{\mathrm{P}}(\mathrm{err}(\hat{h}) > \varepsilon) \leq \delta.$$

Dôkaz. Podľa vety 2.1 platí

$$\underset{T}{\mathrm{P}}(\mathrm{err}(\hat{h}) > \varepsilon) < |H| \cdot e^{-\varepsilon t}.$$

Stačí nám teda zvoliť také t, aby bol výraz na pravej strane menší rovný δ. Odtiaľ dostaneme postačujúci počet trénovacích príkladov t:

$$|H| \cdot e^{-\varepsilon t} \leq \delta \tag{2.3}$$

$$\ln |H| - \varepsilon t \leq \ln \delta \tag{2.4}$$

$$\varepsilon t \geq \ln |H| - \ln \delta \tag{2.5}$$

$$t \geq \frac{1}{\varepsilon} \cdot \left(\ln |H| + \ln \frac{1}{\delta} \right) \tag{2.6}$$

□

2.1.2 Problém konjunkcie

Jedným príkladom problému, kde je množina hypotéz konečná, je *problém konjunkcie pozitívnych literálov.* Na ňom si ukážeme, že (aspoň v niektorých problémoch) sú vyššie uvedené odhady relatívne tesné.

Množina vstupov sú všetky možné priradenia boolovských hodnôt premenným $x_1, \ldots, x_n$, kde n je pevné. Napríklad $x_1 = 0$, $x_2 = 1$, $x_3 = 0$ je priradenie hodnôt. Priradenia vieme zapísať vo vektorovom tvare: vyššie uvedený príklad by sme zapísali ako $x = (0, 1, 0)$. Všetkých vstupov je zrejme 2^n.

Množina hypotéz (konceptov) sú všetky konjunkcie, v ktorých vystupujú iba vyššie uvedené premenné, nie nutne všetky ale vždy bez negácie. Tieto konjunkcie sú chápané ako funkcie, ktoré vracajú 1 iba ak dané priradenie hodnôt konjunkciu spĺňa. Napríklad $x_1 \wedge x_3 \wedge x_4$ vráti 1 na všetkých tých vstupoch, kde táto konjunkcia platí: $x_1 = 1$, $x_3 = 1$, $x_4 = 1$ a ostatné premenné môžu mať ľubovoľnú hodnotu. Všetkých hypotéz je tiež 2^n.

Uvedieme teraz niektoré výsledky z predchádzajúcej časti tak, ako platia pre problém konjunkcie.

Dôsledok 2.4. *Platí*

$$\text{chyba algoritmu} \leq \frac{1}{t} \cdot (n \ln 2 + \ln t + 1) = O\left(\frac{n + \ln t}{t} \right).$$

Dôsledok 2.5. *Aby sme mali zaručené (s pravdepodobnosťou aspoň $1-\delta$), že dostaneme hypotézu s chybou nanajvýš ε, stačí zvoliť veľkosť trénovacej množiny nasledovne:*

$$t \geq \frac{1}{\varepsilon} \cdot \left(n \ln 2 + \ln \frac{1}{\delta} \right) = \Omega\left(\frac{n + \ln \frac{1}{\delta}}{\varepsilon} \right)$$

Ďalej ukážeme, že tieto odhady sú relatívne tesné. Vo všeobecnom prípade je ťažké dostať nejaký dolný odhad, nakoľko pravdepodobnostné rozdelenie P a cieľový koncept $f \in H$ môžu byť degenerované a "uľahčiť algoritmu robotu". Uvidíme ale, že pre niektoré "ťažké" prípady vieme spraviť dolný odhad.

Veta 2.6. *Existuje pravdepodobnostné rozdelenie P a cieľový koncept $f \in H$ také, že nech je trénovací algoritmus ľubovoľný, pre jeho chybu platí nasledovný dolný odhad:*

$$\text{chyba algoritmu} \geq \frac{1}{2e} \cdot \frac{n-1}{t+1} = \Omega\left(\frac{n}{t}\right)$$

V znení vety je trochu obmedzujúce, že cieľový koncept musí byť pevne vybraný. Ukážeme teda najprv, že pokiaľ tvrdenie dokážeme pre náhodne vybrané f, bude z neho plynúť pôvodné tvrdenie.

Lemma 2.7. *Nech P_H je pravdepodobnostné rozdelenie nad množinou hypotéz H. Označme f cieľovú hypotézu, ktorý vyberieme náhodne z P_H. Ak platí*

$$\mathop{\mathrm{E}}_{f \sim P_H}[\text{chyba algoritmu}] \geq c,$$

tak existuje hypotéza $f \in H$ taká, že pre ňu tiež platí daný odhad:

$$\text{chyba algoritmu} \geq c.$$

Dôkaz. Vyplýva z toho, že ak priemer nejakých čísel je c', potom aspoň jedno z tých čísel musí byť väčšie alebo rovné c'. Na strednú hodnotu sa dá pozerať ako na priemer. Spolu s $c' \geq c$ dostávame požadovanú nerovnosť. □

Ďalej pokračujeme dôkazom vety 2.6. V ňom si už môžeme dovoliť vyberať cieľovú hypotézu náhodne.

Dôkaz. Budeme používať pravdepodobnostné rozdelenie, ktoré priradí nenulovú pravdepodobnosť iba určitej sade vstupov. To, aké konkrétne pravdepodobnosti im pridelí, vyplynie z výpočtov ďalej v dôkaze.

$$x^{(1)} = (0, 1, 1, \ldots, 1, 1) \tag{2.7}$$
$$x^{(2)} = (1, 0, 1, \ldots, 1, 1) \tag{2.8}$$
$$\vdots \tag{2.9}$$
$$x^{(n)} = (1, 1, 1, \ldots, 1, 0) \tag{2.10}$$

Tieto vstupy majú nasledujúcu vlastnosť. Pre ľubovoľnú cieľovú hypotézu f platí, že sa v nej nachádza konjunkcia x_i vtedy a len vtedy, keď $f(x^{(i)}) = 0$. Každý z našich vstupov sa dá teda chápať ako "test na niektorú premennú".

Trénovacie príklady teda môžeme chápať tak, že nám dávajú informáciu o jednotlivých premenných: "Je alebo nie je v cieľovej hypotéze?" Následne, po natrénovaní hypotéze kladieme tú istú otázku. Z toho je zrejmé, že pokiaľ nedostaneme niektoré $x^{(i)}$ ako trénovací príklad, môžeme si jedine tipnúť, akú hodnotu nadobúda cieľová hypotéza f na tomto vstupe. A pri tipovaní budeme mať úspešnosť $\frac{1}{2}$, pokiaľ bola cieľová hypotéza vybraná rovnomerne náhodne.

Označme si pravdepodobnosti pridelené jednotlivým vstupom $p_1, \ldots, p_n$. Aká je šanca, že pri trénovaní vstup $x^{(i)}$ nedostaneme, a potom si ho pri testovaní vytiahneme?

$$p_i \cdot (1 - p_i)^t$$

Celková pravdepodobnosť, že si vytiahneme pri testovaní vstup mimo trénovacej množiny, je potom súčet jednotlivých pravdepodobností (nakoľko sú jednotlivé udalosti dizjunktné):

$$\sum_{i=1}^{n} p_i \cdot (1-p_i)^t$$

V týchto prípadoch budeme mať chybu $\frac{1}{2}$. V ostatných prípadoch sme si vytiahli počas testovania nejaký príklad, ktorý bol aj v trénovacej množine. Pokiaľ sme si ho zapamätali, tak budeme mať chybu 0, v každom prípade bude ale chyba aspoň 0. Dostávame tak nasledovný dolný odhad na chybu algoritmu:

$$\mathop{\mathrm{E}}_{f\in H}[\text{chyba algoritmu}] \geq \frac{1}{2} \cdot \left(\sum_{i=1}^{n} p_i \cdot (1-p_i)^t\right)$$

Ako ale zvoliť $p_1, \dots, p_n$ tak, aby sme dostali dobrý dolný odhad? Môžeme napríklad skúsiť zvoliť p_i také, pre ktoré nadobúda výraz $p_i \cdot (1-p_i)^t$ maximum. Zderivovaním a položením rovné 0 dostaneme

$$p_i = \frac{1}{t+1}.$$

Pokiaľ ale $t \neq n-1$, nemôžeme zvoliť všetky pravdepodobnosti takéto: pre $t < n-1$ je súčet pravdepodobností priveľký, pre $t > n-1$ primalý. Prvý prípad nás nezaujíma, nakoľko je to "len konštanta" (v zmysle $t \to \infty$). V druhom prípade si vieme zvoliť jedného "obetného baránka" p_1, ktorému priradíme celú zvyšnú pravdepodobnosť.

$$p_1 = 1 - \frac{n-1}{t+1} \tag{2.11}$$

$$p_2 = \frac{1}{t+1} \tag{2.12}$$

$$\vdots \tag{2.13}$$

$$p_n = \frac{1}{t+1} \tag{2.14}$$

Dosadíme a dostaneme tak odhad:

$$\mathop{\mathrm{E}}_{f\in H}[\text{chyba algoritmu}] \geq \frac{1}{2} \cdot \left((n-1)\cdot\frac{1}{t+1}\cdot\left(1-\frac{1}{t+1}\right)^t + \frac{n-1}{t+1}\cdot\left(1-\frac{n-1}{t+1}\right)^t\right)$$

Odignorujeme druhý sčítanec, a použijeme odhad

$$\left(1-\frac{1}{t+1}\right)^t \geq \frac{1}{e}.$$

Dostávame tak:

$$\mathop{\mathrm{E}}_{f\in H}[\text{chyba algoritmu}] \geq \frac{1}{2e}\cdot\frac{n-1}{t+1}$$

□

Dolný odhad zodpovedajúci dôsledku 2.5 uvedieme bez dôkazu.

Veta 2.8. *Pre ľubovoľný trénovací algoritmus existuje pravdepodobnostné rozdelenie P a cieľová hypotéza $f \in H$, ktoré vynútia, že algoritmus bude potrebovať aspoň*

$$t = \Omega\left(\frac{1}{\varepsilon}\cdot\left(n + \ln\frac{1}{\delta}\right)\right),$$

aby platilo

$$\mathop{\mathrm{P}}_{T}(\mathrm{err}(\hat{h}) > \varepsilon) \leq \delta.$$

TODO dôkaz

2.2 Nekonečné množiny hypotéz