Obsah

Kapitola 1

Úvod do teórie strojového učenia

Predstavme si, že chceme vedieť na základe nejakých vstupných dát x predpovedať výstupné dáta y. Napríklad, môžme chcieť na základe rozlohy bytu vedieť predpovedať jeho cenu. Alebo vedieť z obrázku (vo formáte 32×32 čiernobielych pixelov) povedať, či v ňom nachádza mačka alebo nie.

Snažíme sa teda zachytiť nejaký vzťah medzi dátami x a y. To, akým spôsobom to budeme robiť, je nasledovné: nejakým procesom P získame t *trénovacích príkladov* $(x_1, y_1), \ldots, (x_t, y_t)$. Na základe týchto príkladov budeme chcieť navrhnút nejakú funkciu h, ktorá bude vedieť podľa vstupu x predpovedať výstup y s dostatočnou presnosťou.

Prirodzený spôsob, akým môžeme merať úspešnosť funkcie h, je podľa toho, ako sa jej darí na našich t príkladoch. Dobrá funkcia h ale bude schopná aj *zovšeobecňovať*: bude sa jej dariť dobre nielen na našich príkladoch, ale aj na ľubovoľných dátach, ktoré vieme získať tým istým procesom P.

Ilustrujeme dôležitosť zovšeobecňovania: predstavme si, že funkciu h zostrojíme tak, že si "zapamätáme" všetky trénovacie príklady. Ak je na vstupe x, ktoré bolo medzi trénovacími príkladmi, vrátime príslušné y. Takáto funkcia h má zrejme nulovú chybu na trénovacích príkladoch (pokiaľ pre jedno x je vždy len jedno správne y), ale mimo nich sa jej vôbec nebude dariť.

1.1 Matematický model

Proces, ktorým získavame dáta (x, y), vieme formalizovať ako pravdepodobnostné rozdelenie P. Každému možnému pozorovaniu teda priradíme nejakú pravdepodobnosť, že si ho vytiahneme. Trénovacie príklady získame ako t nezávislých vzoriek z rozdelenia P. Množinu všetkých možných vstupov x označíme X, množinu možných výstupov y označíme Y.

Postup, ktorým zostrojíme funkciu h, vieme formalizovať ako algoritmus, ktorý na vstupe dostane niekoľko trénovacích príkladov $(x_1, y_1), \ldots, (x_t, y_t)$, a na výstupe vráti funkciu $h : X \to Y$, ktorú budeme volať *hypotéza*. Množinu všetkých možných funkcii, ktoré môže náš algoritmus vrátiť, budeme volať *množina hypotéz* a budeme ju značiť H. V tejto kapitole sa nebudeme zaoberať výpočtovou stránkou strojového učenia, takže od detailov ako časová zložitosť, ..., abstrahujeme.

Chyba jednej hypotézy. Ako vyjadriť mieru toho, že sa hypotéze "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 výstupy y a y'.

Pomocou tejto funkcie vieme odmerať očakávanú chybu, ktorú spraví hypotéza h, ak jej na vstupe dáme náhodne vybraný príklad (x, y) z rozdelenia P. Túto chybu budeme tiež označovať err, a vypočítame ju 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 pravdepodobnostného rozdelenia P, teda $(x, y) \sim P$.

Niekedy nás bude zaujímať ale aj chyba, ktorú hypotéza nadobúda na trénovacích dátach T. Aby sme tieto dve chyby rozlíšili, prvú uvedenú chybu budeme volať *testovacia chyba* a druhú uvedenú *trénovacia chyba*. Trénovaciu chybu budeme označovať $\mathrm{err}_T(h)$ a vieme ju vypočítať nasledovne:

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

Algoritmus sa snaží nájsť takú hypotézu $\hat{h}$, ktorej testovacia chyba bude minimálna. Pretože ale nemá presný popis rozdelenia P ale iba niekoľko trénovacích príkladov, v istom zmysle najlepšie, čo môže spraviť, je minimalizovať trénovaciu chybu hypotézy.

Uvedieme ešte príklady používaných chybových funkcii. 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

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

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

Očakávaná testovacia a trénovacia chyba. Vyššie uvedeným spôsobom vieme porovnávať dve rôzne hypotézy. Nevieme ale týmto spôsobom porovnať dve rôzne trénovacie algoritmy. Výstupná hypotéza algoritmu, ktorú budeme označovať $\hat{h}$, totiž závisí od toho, aké trénovacie príklady sme si s rozdelenia P vytiahli, a tiež od prípadnej náhody ak je algoritmus randomizovaný.

Pre jednu trénovaciu množinu T_1 môže byť lepší jeden algoritmus, pre druhý množinu T_2 zas môže dávať lepší výsledok druhý algoritmus. Vieme ale vypočítať strednú hodnotu testovacej chyby (*očakávanú testovaciu chybu*), rátanú cez vyššie uvedené náhodné faktory:

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

Podobným spôsobom vieme vypočítať strednú hodnotu trénovacej chyby (*očakávanú trénovaciu chybu*):

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

V nasledujúcom texte si dovolíme 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 metriky (tj. očakávaná testovacia a očakávaná trénovacia chyba) od veľkosti trénovacej množiny $t = |T|$ 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 byť nejaká *funkcia*. 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 na to, aby sme vedeli predpovedať y. 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 testovacia chyba je

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

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

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

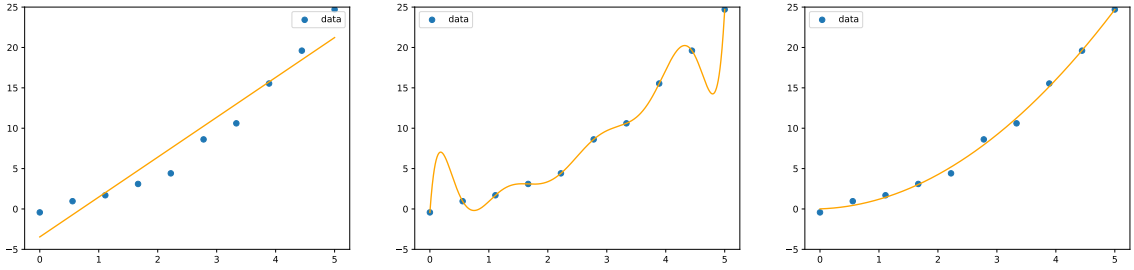
Budeme upravovať výraz reprezentujúci očakávanú testovaciu chybu.

$$\text{očakávaná testovacia chyba} = \mathop{\mathrm{E}}_{T} \left[\mathrm{err}(\hat{h})\right] = \mathop{\mathrm{E}}_{T} \left[\mathop{\mathrm{E}}_{x,y} \left[(\hat{h}(x) - y)^2\right] \right] \tag{1.3}$$

$$= \mathop{\mathrm{E}}_{T} \left[\mathop{\mathrm{E}}_{x,y} \left[\Big((\hat{h}(x) - h^\star(x)) + (h^\star(x) - y) \Big)^2 \right] \right] \tag{1.4}$$

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

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



Obr. 1.1: Podučenie, preučenie, akurát.

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

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

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.

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.

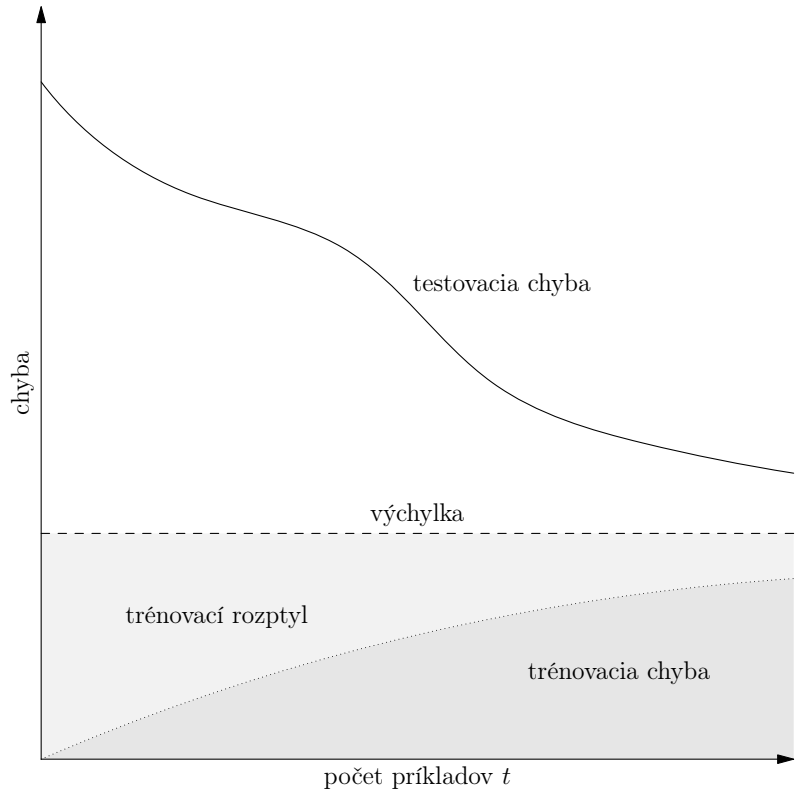
Výchylku vieme upraviť ďalej. Hypotéza $h^\star$ ani y nezávisia od trénovacej množiny T. Z ich pohľadu sú teda testovacie dáta x, y a trénovacie dáta x_i, y_i nerozoznateľné. Takže na meranie chyby $h^\star$ môžeme použiť trénovacie dáta, berúc v úvahu ich náhodný výber:

$$\text{výchylka} = \mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x_i,y_i}\left[(h^\star(x_i) - y_i)^2\right]\right] \tag{1.5}$$

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

Použitím ďalšieho technického kroku dostaneme:

$$\text{výchylka} = \underbrace{\mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x_i,y_i}\left[(h^\star(x_i) - \hat{h}(x_i))^2\right]\right]}_{\text{trénovací rozptyl}} + \underbrace{\mathop{\mathrm{E}}_{T}\left[\mathop{\mathrm{E}}_{x_i,y_i}\left[(\hat{h}(x_i) - y_i)^2\right]\right]}_{\text{očakávaná trénovacia chyba}} \tag{1.7}$$



Obr. 1.2: Závislosť chyby algoritmu od počtu trénovacích príkladov t.

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 je naša vzorka.

Druhý zo sčítancov je očakávaná trénovacia chyba. Je to stredná hodnota chyby, ktorej sa dopustí výstup z algoritmu $\hat{h}$ na tých istých dátach, pomocou ktorých sme $\hat{h}$ zostrojili.

Platí

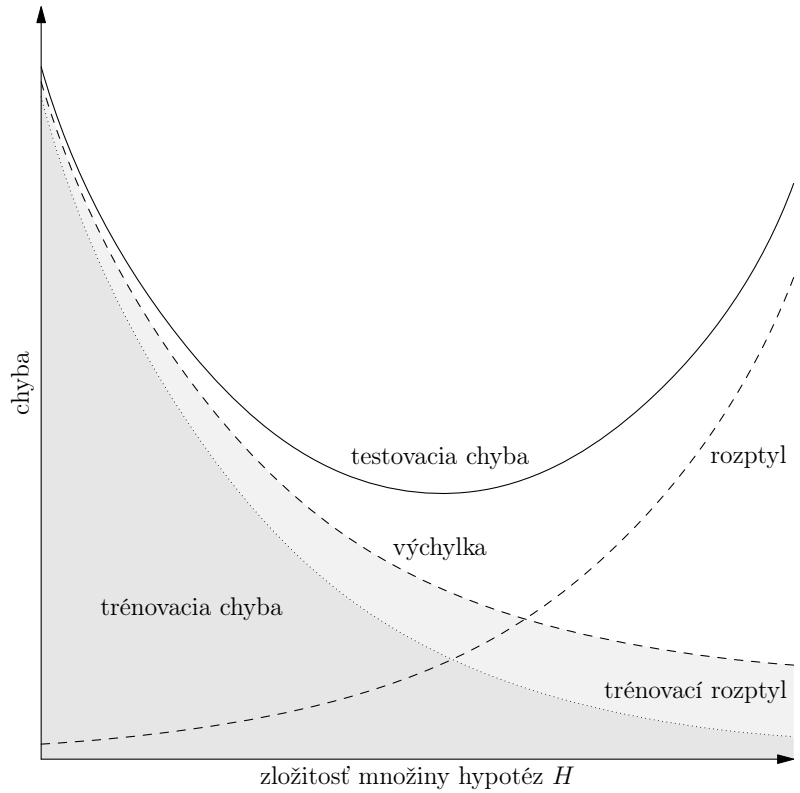
$$\text{očakávaná trénovacia chyba} \leq \text{výchylka} \leq \text{očakávaná testovacia chyba}.$$

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, na ktorých sa žiadnej hypotéze z H nedarí.

Na základe dosiaľ uvedeného vieme graficky znázorniť, ako sa zhruba správajú očakávaná testovacia chyba, rozptyl, výchylka, trénovací rozptyl a očakávaná trénovacia chyba, v závislosti od veľkosti trénovacej množiny (obrázok 1.2) a od zložitosti množiny hypotéz (obrázok 1.3).

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*



Obr. 1.3: Závislosť chyby algoritmu od zložitosti množiny hypotéz H.

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,

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

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, vo všeobecnosti 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 sú výstupné hodnoty y ohraničené). Ale taká je teória.

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

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

$$\langle f, g \rangle = \int f(x) \cdot g(x) \; d\rho x \tag{1.8}$$

$$= \mathop{\mathrm{E}}_{x} \left[f(x) \cdot g(x) \right], \tag{1.9}$$

kde ρ je hustota pravdepodobnosti rozdelenia 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"):

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

Takto definovaná norma spĺňa *trojuholníkovú nerovnosť*: pre ľubovoľné funkcie f, g platí

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

Podľa normy príslušne definujeme vzdialenosť dvoch funkcii:

$$d(f, g) = \|f - g\| .$$

Táto vzdialenosť nám definuje (euklidovskú) metriku nad funkciami, podľa ktorej vieme definovať 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:

$$\|f + g\|^2 = \langle f + g, f + g \rangle \tag{1.10}$$

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

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

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

Definícia 4. (Projekcia na množinu.) *Projekcia* funkcie f na množinu H je taká funkcia f_H, ktorá je spomedzi funkcii v H k našej funkcii f najbližšie:

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

Poznámka 1.6. Ako sa už spomínalo, nie je zrejmé, či je projekcia 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 $h \perp f - f_H$. Inými slovami, kolmica z f na H je kolmá na celé 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 spor s definíciou f_H.

Pozrime sa na všetky funkcie, ktoré ležia na priamke $f_H + \Delta \cdot h$, pre $\Delta \in \mathbb{R}$. Celá táto priamka leží v množine H, nakoľko množina H je uzavretá na lineárne kombinácie a obe funkcie $f_H, h \in H$. Každú bod priamky vieme asociovať s jedným reálnym číslom Δ. Pozrime sa na ich vzdialenosti od funkcie f v závislosti od Δ:

$$\begin{aligned} \operatorname{dist}(\Delta) &= d(f, f_H + \Delta \cdot h) && (1.13)\\ &= \langle (f - f_H) + \Delta \cdot h, (f - f_H) + \Delta \cdot h \rangle && (1.14)\\ &= \langle f - f_H, f - f_H \rangle + 2\Delta \cdot \langle h, f - f_H \rangle + \Delta^2 \cdot \langle h, h \rangle && (1.15) \end{aligned}$$

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:

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

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

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 sú obe kolmice $f - g$ aj $f - h$ kolmé na celé H. Špeciálne sú teda obe kolmé na g aj na h, odkiaľ dostaneme:

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

Tieto rovnosti upravíme pomocou linearity skalárneho súčinu. Dostaneme z nich

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

Nakoniec, pozrime sa na normu funkcie $g - h$. Pomocou predchádzajúceho výsledku ukážeme, že musí byť nulová, čo môže nastať jedine pre $g = h$:

$$\|g - h\| = \sqrt{\langle g - h, g - h\rangle} \tag{1.23}$$
$$= \sqrt{\langle g, g\rangle - \langle g, h\rangle - \langle h, g\rangle + \langle h, h\rangle} \tag{1.24}$$
$$= 0 \tag{1.25}$$

□

Lemma 1.5. *Hypotéza $h^\star$ je projekciou $h^\square$ na H, inak zapísané $h^\star = h^\square_H$. Tvrdíme teda, že najlepšia funkcia z množiny H je tá, ktorá sa (v euklidovskej metrike) najmenej líši od $h^\square$.*

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

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

Druhý sčítanec je konštanta, teda nás pri minimalizácii nemusí zaujímať. Môžeme ho teda ignorovať. Tretí sčítanec vieme upraviť nasledovne:

$$\mathop{\mathrm{E}}_{x,y}\left[(h(x) - h^\square(x)) \cdot (h^\square(x) - y)\right] = \mathop{\mathrm{E}}_{x}\left[\mathop{\mathrm{E}}_{y|x}\left[(h(x) - h^\square(x)) \cdot (h^\square(x) - y)\right]\right] \tag{1.29}$$
$$= \mathop{\mathrm{E}}_{x}\left[(h(x) - h^\square(x)) \cdot \mathop{\mathrm{E}}_{y|x}\left[h^\square(x) - y\right]\right] \tag{1.30}$$
$$= 0 \tag{1.31}$$

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. □

Vyzbrojení týmito znalosťami, môžeme sa vrhnúť na dôkaz vety 1.1. Pripomeňme si na začiatok 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 nasledovnej úvahy. Pretože $h^\star = h^\square_H$, funkcia $h^\star - h^\square$ je kolmicou z funkcie $h^\square$ na množinu H. Je teda kolmá na celú množinu H. Stačí preto ukázať, že funkcia $\hat{h} - h^\star$ je z H; to ale plynie z uzavretosti H na lineárne kombinácie. □

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:

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

Tvrdíme, že sa dá rozložiť na tri nasledovné časti:

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

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), teda $h^{\square}$.

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.

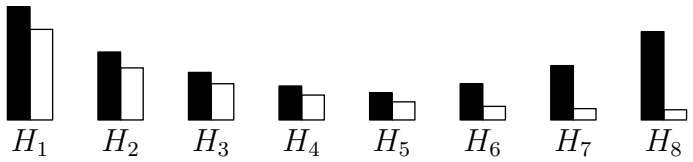
$$\begin{aligned}
\text{očakávaná testovacia chyba} &= \mathrm{E}\left[(\hat{f}(x) - y)^2\right] && (1.32)\\
&= \mathrm{E}\left[(\hat{f}(x) - f(x) - \varepsilon)^2\right] && (1.33)\\
&= \mathrm{E}\left[(\hat{f}(x) - f(x))^2\right] + \mathrm{E}\left[\varepsilon^2\right] - 2\cdot\mathrm{E}\left[\varepsilon\cdot(\hat{f}(x)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

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.

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



Obr. 1.4: Testovacie (čierne) a trénovacie (biele) chyby v čím ďalej, tým zložitejších množinách hypotéz H.

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

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

□

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:

$$H_1 \subseteq H_2 \subseteq H_3 \subseteq \ldots$$

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 1.4.

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. Výstupom algoritmu potom je

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

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.

Vo všetkých prípadoch je pokuta parametrizovaná reálnym parametrom λ hovoriacim, ako veľké pokuty chceme udeľovať. Veľké λ (v porovnaní s trénovacou chybou) nám hovorí, že sa snažíme hlavne dosiahnuť jednoduché hypotézy; s malým λ zas kladieme dôraz na hypotézy s menšou trénovacou chybou.

Uvedieme si niekoľko príkladov výrazov, ktoré môžu byť použité ako pokuta. 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*).

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

  Táto pokuta "tlačí" váhy nepotrebných atribútov do nuly, pričom väčšie váhy tlačí viac. Takže čím dôležitejší atribút, tým väčšiu váhu si môže dovoliť mať.

- L_1 regularizácia (známa aj ako *lasso*).

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

  Opäť "tlačíme" nepotrebné atribúty do nuly, pričom ale všetky váhy tlačíme rovnako. To nám vie vynulovať nepotrebné atribúty, čo nám vie znížiť 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 tej množiny hypotéz, ktorú sme identifikovali ako najvhodnejšiu. Výsledná hypotéza je výstupom algoritmu.

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.

k-fold evaluation. Pri holdout testovaní 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 1.11. "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

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

1.3. Dokážte korektnosť druhého technického kroku, v odvodení rozkladu výchylky na trénovací rozptyl a očakávanú 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*).

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.

Začneme definíciou PAC učenia, ktorá ešte nebude brať do úvahy výpočtovú stránku učenia.

Zameriame sa na binárne klasifikačné úlohy, v ktorých je cieľom rozlíšiť medzi reprezentantmi nejakého cieľového konceptu od nereprezentantov. Napríklad cieľovým konceptom môže byť "písmeno A". Hypotéza dostane na vstupe obrázok 32×32 a má povedať, či tento obrázok vyobrazuje písmeno A alebo nie.

Máme teda množinu konceptov C, z ktorej pochádza cieľový koncept c. Náš trénovací algoritmus ale nevie, ktorý z nich to je. Jeho úlohou je nájsť dobrú aproximáciu, ktorú bude hľadať v množine hypotéz H. Budeme predpokladať, že $H \supseteq C$, aby bolo zaručená existencia dobrej hypotézy.

Algoritmus na vstupe dostane niekoľko trénovacích príkladov v tvare dvojíc $(x, c(x))$, pričom x pochádza z pravdepodobnostného rozdelenia P. Uvažovať y v rozdelení P nie je potrebné, nakoľko je jednoznačne určené cez x a c.

Definícia 5. Nech C je množina konceptov nad množinou vstupov X. Hovoríme, že C je *PAC naučiteľná* ak existuje algoritmus L s nasledujúcou vlastnosťou: pre každý (cieľový) koncept $c \in C$, pre každé $\varepsilon > 0$, $\delta > 0$ a pre každé možné pravdepodobnostné rozdelenie P, ak algoritmu L dáme na vstupe t trénovacích príkladov $(x_i, c(x_i))$ náhodne vybraných podľa P a t je dostatočne veľké, tak nám algoritmus s pravdepodobnosťou nanajvýš δ vráti hypotézu $\hat{h} \in C$ spĺňajúcu

$\mathrm{err}(\hat{h}) \leq \varepsilon$. Táto pravdepodobnosť sa berie cez náhodnosť v ťahaní trénovacích príkladov a prípadnú náhodnosť algoritmu L.

Poznámka 2.1. Neskôr túto definíciu rozšírime tak, aby brala do úvahy aj výpočtovú stránku algoritmu. Vtedy sa budeme zaoberať aj tým, v akom čase náš algoritmus beží a koľko trénovacích príkladov potrebuje. Budeme hovoriť o *efektívnej PAC naučiteľnosti*.

Poznámka 2.2. Pripomíname, že chyba hypotézy h sa pre klasifikačné úlohy počíta ako

$$\mathrm{err}(h) = \mathop{\mathrm{E}}_{x\sim P}[h(x) \neq c(x)] = \mathop{\mathrm{P}}_{x\sim P}(h(x) \neq c(x)).$$

Budeme hovoriť, že trénovací príklad (x, y) je *pozitívny*, ak $y = 1$. V opačnom prípade budeme hovoriť, že príklad je *negatívny*.

2.1 Konečné množiny hypotéz

V tejto časti sa zameriame na konečné množiny hypotéz.

Budeme predpokladať, že algoritmus vždy vráti *konzistentný klasifikátor*: takú hypotézu, ktorá je konzistentná s trénovacími príkladmi, teda že pre ľubovoľnú trénovaciu množinu T platí $\mathrm{err}_T(\hat{h}) = 0$. Za predpokladu $H \supseteq C$ je to splniteľný predpoklad: jedným konzistentným klasifikátorom je samotný cieľový koncept c.

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:*

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

Dôkaz. Ak žiadna zlá hypotéza nie je konzistentná s príkladmi, tak výstupom algoritmu nemôže byť zlá hypotéza. Budeme sa teda snažiť zhora odhadnúť pravdepodobnosť, že aspoň jedna zlá hypotéza "prežila".

Nech h je ľubovoľná zlá hypotéza. Pravdepodobnosť, že je konzistentná s trénovacími príkladmi, 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 je všetkých hypotéz, teda $|H|$. Pravdepodobnosť, že aspoň jedna z nich bude konzistentná s príkladmi, sa dá odhadnúť zhora ako súčet ich pravdepodobností:

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

Odkiaľ dostávame požadovanú nerovnosť. □

Poznámka 2.3. Vo vyššie uvedenom dôkaze sme nepredpokladali nič o pravdepodobnostnom rozdelení P ani o cieľovom koncepte c. Uvedená veta teda platí pre ľubovoľné P a c.

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*

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

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:

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

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

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

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 je 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*

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

Ekvivalentne,

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

Dôkaz. Podľa vety 2.1 platí

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

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.

Nech je dané n. Množina vstupov sú všetky možné ohodnotenia boolovských premenných $x_1, \ldots, x_n$. 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 konceptov C sú všetky konjunkcie nad pozitívnymi literálmi $x_1, \ldots, x_n$. Tieto konjunkcie sú chápané ako funkcie, ktoré vracajú 1 iba ak dané ohodnotenie premenných spĺňa túto konjunkciu. Napríklad $x_1 \wedge x_3 \wedge x_4$ vráti 1 na všetkých tých vstupoch, kde táto konjunkcia platí: musí platiť $x_1 = 1$, $x_3 = 1$, $x_4 = 1$, ale ostatné premenné už môžu mať ľubovoľnú hodnotu. Všetkých konceptov je zrejme tiež 2^n. Množina hypotéz H bude rovnaká, ako množina konceptov.

Aby sme videli, že konzistentnosť s trénovacími príkladmi je realistická požiadavka, ukážeme si, ako sa dá na základe trénovacích príkladov zostrojiť nejaký konzistentný klasifikátor.

Veta 2.4. *Nech hypotéza $\hat{h}$ je konjunkciou všetkých tých premenných, ktoré sa vyskytujú vo všetkých pozitívnych trénovacích príkladoch. Potom $\hat{h}$ je konzistentná so všetkými trénovacími príkladmi.*

Dôkaz. Ak (x, y) je pozitívny príklad, v cieľovej hypotéze c môžu byť jedine tie premenné, ktoré majú v x priradenú hodnotu 1. Keď túto úvahu zopakujeme pre všetky pozitívne príklady, dostaneme niekoľko množín "povolených premenných". Cieľová premenná musí byť podmnožinou všetkých z nich, teda je podmnožinou ich prieniku. Pritom ich prienik je práve hypotéza $\hat{h}$.

Z toho vyplýva, že ak nie je splnené c, nemôže byť splnené ani h. Naša hypotéza totiž kladie ešte väčšie požiadavky na hodnoty premenných. Teda,

$$(c(x) = 0) \implies (h(x) = 0),$$

takže h je konzistentná s negatívnymi príkladmi.

Čo sa týka pozitívnych príkladov, v h sú všetky tie premenné, ktoré sme do nej mohli dať tak, aby bola konzistentná so všetkými pozitívnymi príkladmi. Teda jej konzistentnosť s pozitívnymi príkladmi vyplýva priamo z jej konštrukcie. □

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

Dôsledok 2.5. *Platí*

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

Dôsledok 2.6. *Aby sme mali zaručené (s pravdepodobnosťou aspoň $1-\delta$), že dostaneme hypotézu s chybou nanajvýš ε, stačí zvoliť veľkosť trénovacej množiny nasledovne:*

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

Tieto odhady sú relatívne tesné. Vo všeobecnom prípade je ťažké dostať nejaký dolný odhad, nakoľko pravdepodobnostné rozdelenie P a cieľový koncept c môžu byť degenerované a "uľahčiť algoritmu robotu". Uvidíme ale, že pre niektoré "ťažké" prípady vieme spraviť dolný odhad.

Veta 2.7. *Existuje pravdepodobnostné rozdelenie P a cieľový koncept $c \in C$ také, že nech je trénovací algoritmus ľubovoľný, pre jeho chybu platí nasledovný dolný odhad:*

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

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

Lemma 2.8. *Nech P_C je pravdepodobnostné rozdelenie nad množinou konceptov C. Cieľovú hypotézu vyberieme náhodne podľa P_C. Ak pre nejaké číslo k platí, že stredná hodnota chyby algoritmu je aspoň k, tj.*

$$\mathop{\mathrm{E}}_{c\sim P_C}[\text{očakávaná testovacia chyba}] \geq k,$$

tak existuje voľba cieľového konceptu $c \in C$ taká, že chyba algoritmu bude tiež aspoň k.

Dôkaz. Ak by na každom cieľovom koncepte bola chyba algoritmu menšia ako k, potom by aj ľubovoľný vážený priemer chýb (zodpovedajúci strednej hodnote s rozdelením P_C) bol menší ako k. To by bol spor. □

Ďalej pokračujeme dôkazom vety 2.7. V ňom si už môžeme dovoliť vyberať cieľový koncept náhodne.

Dôkaz. Zvolíme pravdepodobnostné rozdelenie, ktoré priradí nenulovú pravdepodobnosť iba nasledovnej sade vstupov. Konkrétne pravdepodobnosti určíme neskôr v dôkaze tak, ako sa nám to bude hodiť.

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

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

Trénovacie príklady potom môžeme chápať tak, že algoritmu dávajú informáciu o jednotlivých premenných: "Je alebo nie je v cieľovej hypotéze?" Pokiaľ ale niektorý zo vstupov $x^{(i)}$ nie je medzi trénovacími príkladmi, nemáme žiadnu informáciu o tom, či sa tam príslušná premenná nachádza. Ak si potom $\hat{h}$ vytiahne pri testovaní takýto vstup, môže jedine hádať, aká je správna odpoveď. Šanca úspechu pri hádaní bude $\frac{1}{2}$, pokiaľ vyberieme c rovnomerne náhodne z celej množiny konceptov.

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 bude mať $\hat{h}$ 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:

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

Ako ale zvoliť $p_1, \ldots, p_n$ tak, aby sme dostali dobrý dolný odhad? Skúsime 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 a 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:

$$\mathop{\mathrm{E}}_{c \in C}[\text{očakávaná testovacia chyba}] \geq \frac{1}{2} \cdot \left((n-1) \cdot \frac{1}{t+1} \cdot \left(1 - \frac{1}{t+1}\right)^t + \frac{n-1}{t+1} \cdot \left(1 - \frac{n-1}{t+1}\right)^t \right)$$

Odignorujeme druhý sčítanec, čím sa nám výraz na pravej strane môže len zmenšiť, takže nerovnosť sa zachová. Ďalej použijeme odhad

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

ktorý platí pre všetky prirodzené t. (Tento odhad sa dá dokázať napríklad tak, že sa dokáže, že daný výraz je rastúci od t. Odhad z toho už plynie ľahko, nakoľko jeho limita pre $t \to \infty$ je práve $\frac{1}{e}$.) Dostaneme tak požadovanú nerovnosť:

$$\mathop{\mathrm{E}}_{c \in C}[\text{očakávaná testovacia chyba}] \geq \frac{1}{2e} \cdot \frac{n-1}{t+1}$$

□

Dolný odhad zodpovedajúci dôsledku 2.6 uvedieme bez dôkazu.

Veta 2.9. *Pre ľubovoľný trénovací algoritmus existuje pravdepodobnostné rozdelenie P a cieľová hypotéza $f \in H$, ktoré vynútia, že algoritmus bude potrebovať aspoň*

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

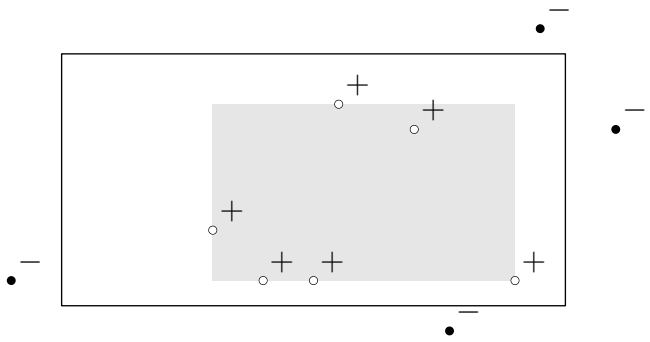
aby platilo

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

TODO dôkaz

2.2 Nekonečné množiny hypotéz

Vyššie uvedený postupy pre konečné množiny hypotéz zlyhajú, ak $|H| = \infty$: nedostaneme z nich žiaden odhad. Aj pre nekonečné množiny hypotéz je ale možné odvodiť horné odhady. Tie sa už nebudú odvíjať od veľkosti množiny H, ale od nejakej jej miery zložitosti. Touto mierou zložitosti bude tzv. *Vapnik-Chervonenkisova dimenzia*. Predtým sa ale pozrieme na jednoduchý príklad.



Obr. 2.1: Obdĺžniková hra: najmenší obdĺžnik konzistentný s príkladmi.

2.2.1 Obdĺžniková hra

Ide o klasifikačnú úlohu. Množina vstupov sú všetky body v rovine. Množina konceptov sú všetky obdĺžniky, ktorých strany sú rovnobežné so súradnicovými osami. Pre každý bod v rovine sa teda pýtame, či je vo vnútri cieľového obdĺžnika c alebo nie. Body na okraji obdĺžnika považujeme, že sú vnútri.

Pre jednoduchosť argumentu použijeme konkrétny trénovací algoritmus. Dôkaz by sa dal zovšeobecniť na prípad, kedy jediné, čo o algoritme predpokladáme je, že nám vracia konzistentný klasifikátor. S nasledovným konzistentným klasifikátorom to ale bude jednoduchšie.

Z trénovacích príkladov vezmime tie pozitívne. Ako hypotézu $\hat{h}$ zvolíme najmenší (vzhľadom na inklúziu) osovorovnobežný obdĺžnik, ktorý obsahuje všetky body vo vybraných príkladoch. Na obrázku 2.1 ilustrujeme konštrukciu $\hat{h}$ a porovnávame ho s c.

Lemma 2.10. *Každá množina bodov $(x_1, y_1), \ldots, (x_t, y_t)$ má unikátny* bounding box*: najmenší (vzhľadom na inklúziu) osovorovnobežný obdĺžnik, ktorý ich všetky obsahuje.*

Dôkaz. Každý obdĺžnik je jednoznačne určený x-ovými súradnicami ľavého a pravého okraja a y-ovými súradnicami dolného a horného okraja. Označme ich $x_\leftarrow, x_\rightarrow$ a $y_\downarrow, y_\uparrow$. Vo vnútri sú práve tie body (x, y), ktoré spĺňajú

$$x_\leftarrow \leq x \leq x_\rightarrow,\ y_\downarrow \leq y \leq y_\uparrow.$$

Ak teda majú byť všetky spomínané body vo vnútri, musí platiť

$$x_\leftarrow \leq \min x_i,\ x_\rightarrow \geq \max x_i,\ y_\downarrow \leq \min y_i,\ y_\uparrow \geq \max y_i$$

Najmenší osovorovnobežný obdĺžnik, ktorý toto spĺňa, je ten, pre ktorý nastávajú v jednotlivých nerovnostiach rovnosti. Teda $x_\leftarrow = \min x_i$, $x_\rightarrow = \max x_i$, $y_\downarrow = \min y_i$ a $y_\uparrow = \max y_i$. □

Dôsledok 2.11. *Obdĺžnik $\hat{h}$ je podmnožinou cieľového obdĺžnika c.*

Dôsledok 2.12. *Obdĺžnik $\hat{h}$ je konzistentný klasifikátor.*

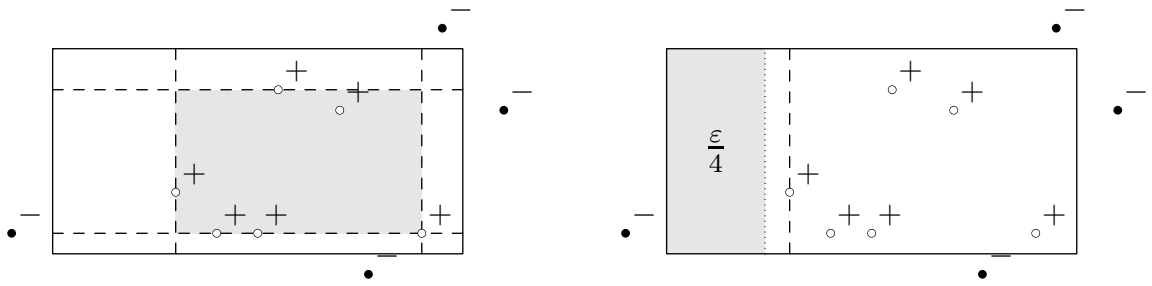
Dôkaz. Konzistentnosť na pozitívnych príkladoch vyplýva z konštrukcie $\hat{h}$. Každý negatívny príklad je mimo c a teda aj mimo $\hat{h}$, je teda konzistentný aj s negatívnymi príkladmi. □

Veta 2.13. *Obdĺžniková hra je PAC naučiteľná: pre každý cieľový obdĺžnik $c \in C$, $\varepsilon > 0$, $\delta > 0$ a pravdepodobnostné rozdelenie P, ak vyššie popísanému algoritmu dáme t trénovacích príkladov z P a platí*

$$t \geq \frac{4}{\varepsilon} \cdot \ln \frac{4}{\delta},$$

tak nám algoritmus s vysokou pravdepodobnosťou vráti hypotézu s nízkou chybou:

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



(a) Krajné pásiky.

(b) Sivou farbou je vyznačená množina všetkých bodov, pre ktoré má pásik naľavo od nich váhu nanajvýš $\frac{\varepsilon}{4}$.

Obr. 2.2: Obdĺžniková hra.

Dôkaz. Zle klasifikované body sú práve tie, ktoré sú vo vnútri c ale nie sú vo vnútri $\hat{h}$. Každý z týchto bodov padne do aspoň jedného z okrajových "pásikov" (nerátajúc jeden z okrajov), ako je zobrazené na obrázku 2.2a.

Pod *váhou* množiny budeme rozumieť pravdepodobnosť, že náhodný bod z rozdelenia P padne do tejto množiny. Označme $p_{\leftarrow}, p_{\rightarrow}, p_{\uparrow}, p_{\downarrow}$ postupne váhy ľavého, pravého, horného, resp. dolného pásika. Chyba $\hat{h}$ sa dá zhora odhadnúť ako súčet týchto váh:

$$\mathrm{err}(\hat{h}) \leq p_{\leftarrow} + p_{\rightarrow} + p_{\uparrow} + p_{\downarrow}.$$

Ak by sme voľbou dostatočne veľkého t vedeli zaručiť (s pravdepodobnosťou aspoň $1 - \delta$), že dostaneme takú hypotézu $\hat{h}$, pre ktorú je výraz na pravej strane nanajvýš ε, vyhrali by sme. Ukážeme, že vieme zaručiť

$$p_{\leftarrow}, p_{\rightarrow}, p_{\uparrow}, p_{\downarrow} \leq \frac{\varepsilon}{4}.$$

Postup bude vo všetkých štyroch prípadoch ten istý, ukážeme si to teda len na ľavom pásiku.
TODO dôkaz (eww, grc)

□

2.2.2 Vapnik-Chervonenkisova dimenzia

Vapnik-Chervonenkisova dimenzia (alebo skrátene VC dimenzia) je miera zložitosti množiny hypotéz, pomocou ktorej sme schopní tvoriť tvrdenia v štýle PAC učenia. Na jej definíciu budeme potrebovať niekoľko pomocných pojmov.

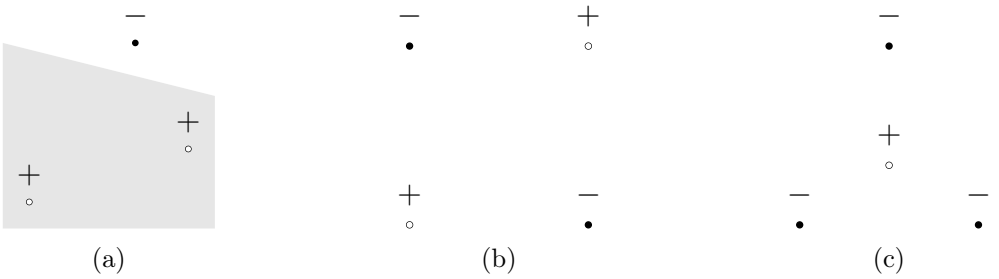
Definícia 6. Majme konečnú podmnožinu vstupov $S \subseteq X$. Hovoríme, že množina hypotéz H *rozbíja* množinu S, ak platí: nech označíme vstupy v S ako pozitívne alebo negatívne akokoľvek, v množine H existuje hypotéza konzistentná s týmto označením.

Intuitívne, množina S je rozbitá hypotézami H, ak nie je pre hypotézy v H príliš náročné modelovať príklady v S: sú schopné ich modelovať akokoľvek.

Definícia 7. Vapnik-Chervonenkisova dimenzia množiny hypotéz H, označovaná $VCD(H)$, je veľkosť najväčšej množiny $S \subseteq X$, ktorá sa dá rozbiť hypotézami v H. Ak sa dá rozbiť ľubovoľne veľká množina S, definujeme $VCD(H) = \infty$.

Teda na to, aby sme ukázali dolný odhad $VCD(H) \geq d$, stačí nám nájsť jednu množinu S veľkosti d, ktorá sa dá rozbiť. Aby sme ale ukázali horný odhad $VCD(H) \leq d$, musíme ukázať, že žiadna množina veľkosti $d + 1$ sa nedá rozbiť: teda že existuje také označenie vstupov, pre ktoré neexistuje v H konzistentný klasifikátor. Z tohto dôvodu je obvykle ťažšie dokázať horný odhad na VC dimenziu, ako dolný odhad.

Obr. 2.3: Žiaden interval nevytvorí takéto označenie bodov.



Obr. 2.4: Polroviny v rovine: v situácii (a) vieme nájsť deliacu priamku, v (b) a (c) nie.

Lemma 2.14. *Majme dve množiny hypotéz H a H' nad tou istou množinou vstupov x. Ak platí $H \supseteq H'$, tak potom platí $VCD(H) \succeq VCD(H')$ (kde $\succeq$ je relácia $\geq$ prirodzene rozšírená na $\mathbb{N} \cup \{\infty\}$).*

Dôkaz. Ak sa množina S dá rozbiť pomocou H', potom, pretože každá hypotéza v H' je aj v H, dá sa táto množina rozbiť aj pomocou H (rovnakým spôsobom). □

Než uvedieme vety hovoriace o PAC naučiteľnosti, uvedieme príklady niektorých nekonečných množín hypotéz, a neformálne zdôvodníme ich VC dimenzie. Väčšinou budú geometrického charakteru (čo je prirodzené, ak máme mať nekonečnú množinu hypotéz).

Intervaly na reálnych číslach. Nie je ťažké vidieť, že ľubovoľná dvojica bodov sa dá rozbiť. Na druhej strane, ak máme tri body, tak ich vieme označiť ako na obrázku 2.3, čo nie je konzistentné so žiadnym intervalom. Takže VC dimenzia je 2.

Polroviny v rovine. Každá trojica bodov tvoriacich nedegenerovaný trojuholník sa dá rozbiť, ako je ilustrované na obrázku 2.4. Na druhej strane, žiadna štvorica bodov sa rozbiť nedá: ak tvoria konvexný štvoruholník, tak jednu protiľahlú dvojicu označíme kladne a druhú záporne. Ak tvoria nekonvexný štvoruholník, jeden z bodov je vo vnútri trojuholníka tvoreného ostatnými tromi: tento bod označíme kladne a ostatné body záporne. Takže VC dimenzia je 3.

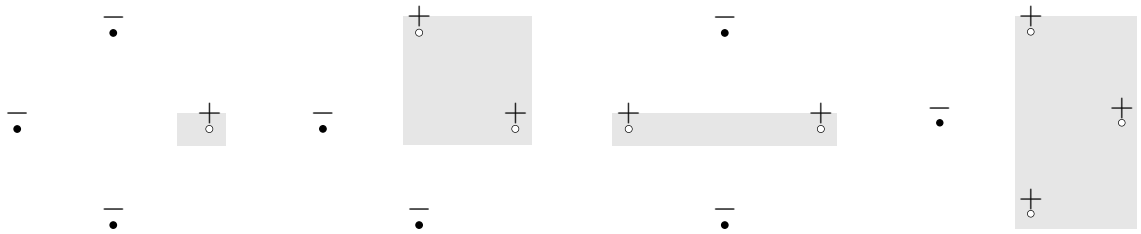
Osovorovnobežné obdĺžniky. Štvorica bodov, ktorá sa dá rozbiť, je ilustrovaná na obrázku 2.5, nie každá štvorica bodov sa ale dá rozbiť. Na druhej strane, žiadna pätica bodov sa nedá rozbiť. Rozoberieme dva prípady: ak je aspoň jeden z bodov vo vnútri bounding boxu (nie na okraji), tak ho označíme záporne a všetky body na okraji kladne. Ak sú všetky body na obvode bounding boxu, na jednej strane toho obdĺžnika musia ležať aspoň dva body. Jeden z nich označíme kladne a druhý záporne. Takže VC dimenzia je 4.

Nakoniec hlavný výsledok, kvôli ktorému je VC dimenzia zaujímavá.

Veta 2.15 (Blumer et. al., 1989 [1])**.** *Nech C je ľubovoľná množina konceptov. Nech H je množina hypotéz s VC dimenziou rovnou d. Nech L je ľubovoľný algoritmus, ktorý pre ľubovoľnú sadu t trénovacích príkladov vráti hypotézu $\hat{h}$ konzistentnú s príkladmi. Potom L je PAC trénovací algoritmus pre množinu konceptov C s použitím hypotéz v H, pokiaľ je t dostatočne veľké:*

$$t \geq c_0 \left(\frac{1}{\varepsilon} \ln \frac{1}{\delta} + \frac{d}{\varepsilon} \ln \frac{1}{\varepsilon} \right)$$

pre nejakú konštantu $c_0 > 0$.



Obr. 2.5: Osovorovnobežné obdĺžniky: štvorica bodov, ktorá sa dá rozbiť. Jednotlivé obrázky zobrazujú všetky netriviálne označenia bodov a príslušné konzistentné klasifikátory.

TODO dôkaz

Veta 2.16 (Haussler et. al., 1994 [3]). *Ak VC dimenzia je d, tak ľubovoľný trénovací algoritmus, ktorý vždy vracia hypotézy konzistentné s trénovacími príkladmi, má chybu nanajvýš*

$$\mathrm{E}\left[\mathrm{err}(\hat{h})\right] = O\left(\frac{d}{t}\ln\frac{t}{d}\right).$$

TODO dôkaz

Čo ale v prípade, že neexistuje konzistentný klasifikátor? Taká situácia nastane, keď buď cieľový koncept nie je v množine hypotéz, alebo keď v probléme vystupuje šum. Ukazuje sa, že aj v takom prípade sa dá odhadnúť chyba hypotézy.

Veta 2.17 (Vapnik & Chervonenkis, 1971 [2]). *Nech $h^\star$ je hypotéza v H s najmenšou (testovacou) chybou a nech $\hat{h}$ je hypotéza v H s najmenšou trénovacou chybou. Ak $VCD(H) = d$, tak pre počet trénovacích príkladov*

$$t \geq c_0\left(\frac{1}{\varepsilon}\ln\frac{1}{\delta} + \frac{d}{\varepsilon}\ln\frac{1}{\varepsilon}\right),$$

kde c_0 je nejaká konštanta, platí, že s veľkou pravdepodobnosťou je chyba hypotézy $\hat{h}$ dostatočne malá (berúc v úvahu najmenšiu dosiahnuteľnú chybu):

$$\mathrm{P}(\mathrm{err}(\hat{h}) > \mathrm{err}(h^\star) + \varepsilon) \leq \delta.$$

TODO dôkaz

Literatúra

[1] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.

[2] AIA Chervonenkis and VN Vapnik. Theory of uniform convergence of frequencies of events to their probabilities and problems of search for an optimal solution from empirical data(average risk minimization based on empirical data, showing relationship of problem to uniform convergence of averages toward expectation value). *Automation and Remote Control*, 32:207–217, 1971.

[3] David Haussler, Nick Littlestone, and Manfred K Warmuth. Predicting {0, 1}-functions on randomly drawn points. *Information and Computation*, 115(2):248–292, 1994.