
Ohodnocení úspěšnosti klasifikace

Základní statistiky uváděné při ohodnocování modelů pro klasifikaci (např. ve Weka) vycházejí z tzv. matice záměn (confusion matrix):

správná třída \ klasifikace	+	-
+	TP – true positive	FN – false negative
-	FP – false positive	TN true negative

Česky se říká správně/falešně pozitivní/negativní.

správná třída \ klasifikace	+	-
+	TP – true positive	FN – false negative
-	FP – false positive	TN true negative

Základní míry ohodnocení modelu

celková správnost	accuracy	$Acc = \frac{TP+TN}{TP+TN+FP+FN}$
chyba	error	$Err = \frac{FP+FN}{TP+TN+FP+FN}$
přesnost	precision	$Prec = \frac{TP}{TP+FP}$
úplnost, sensitivita	recall, sensitivity	$Sensit = Rec = \frac{TP}{TP+FN}$
specificita	specificity	$Specificity = \frac{TN}{TN+FP}$

Různá cena chyby

Raději zařadím spam do normální pošty než normální email do spamu.

$L_{kk|}$ matice ceny za chybnou klasifikaci k jakožto $k|$.

- Predikce na listech se změní tak (minimalizuje cenu chyb), v listu klasifikujeme $k(m) = \operatorname{argmin}_k \sum_l L_{lk} \hat{p}_{ml}$.
- Můžeme tvořit strom uzpůsobený na ceny $L_{kk|}$.
 - Pro vícekategoriální klasifikaci modifikujeme $Gini = \sum_{k \neq k|} L_{kk|} \hat{p}_{mk} \hat{p}_{mk|}$.
 - Pro dvouhodnotovou vážíme prvky třídy k $L_{kk|}$ krát.
- Pro porovnání modelů s proměnnou cenou chyb slouží křivka ROC (obrázek).

Cena za pozorování

$$\frac{Gain^2(S, X_j)}{Cost(X_j)}$$

učení robota, nakolik objekty mohou být uchopeny nebo

$$\frac{2^{Gain(S, X_j) - 1}}{(Cost(X_j) + 1)^w}$$

Medicínská diagnostika.

Pravidla ze stromů

- Ze stromu můžeme vytvořit pravidla napsáním pravidla pro každý list.
- Ta pravidla ale můžeme ještě zlepšit tím, že uvažujeme každý atribut v každém pravidle a zjistíme, jestli jeho vynecháním pravidlo nezlepšíme (tj. nezlepšíme chybu na validačních datech resp. pesimistický odhad chyby na trénovacích datech).
- To funguje celkem dobře, ale učí se dlouho, protože pro každý atribut musíme projít všechny trénovací příklady. Algoritmy tvořící pravidla přímo bývají rychlejší.
- Výsledná pravidla setřídíme podle klesající úspěšnosti.

Stromy pro numerickou predikci

- Listy stromů obsahují numerické hodnoty, rovné **průměru** **trénovacích příkladů v listu**.
- pro výběr atributu k dělení zkusíme všechny řezy, vybíráme **maximální zlepšení střední kvadratické chyby**.
- Tyto stromy se nazývají také **regresní stromy**, protože statistici nazývají regrese proces pro predikci numerických hodnot.
- Můžeme i kombinovat regresní přímku a rozhodovací strom tím, že v každém uzlu bude regresní přímka – takový strom se nazývá **model tree**.

CART

Hledá proměnnou j a bod řezu s na oblasti $R_1(j, s) = \{X|X_j \leq s\}$ a $R_2(j, s) = \{X|X_j > s\}$

takové, aby minimalizovaly

$$\min_{j,s} = \left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right]$$

vnitřní minima jsou průměry, tj. $\hat{c}_1 = \text{avg}(y_i | x_i \in R_1(j, s))$.

Prořezávání v CART

- Naučme strom T_0 až k listům s málo (5–ti) záznamy.
- Vytvoříme hierarchii podstromů $T \subset T_0$ postupným sléváním listů dohromady.
- Listy stromu T o velikosti $|T|$ indexujeme m , list m pokrývá oblast R_m , definujeme:

$$\hat{c}_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i,$$
$$Q_m(T) = \frac{1}{N_m} \sum_{x_i \in R_m} (y_i - \hat{c}_m)^2.$$

-
- Definujeme kritérium k optimalizaci:

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha |T|.$$

- Vyjdeme z T_0 , postupně slijeme vždy dva listy, které způsobí nejmenší nárůst $\sum_m N_m Q_m(T)$, až nám zbyde jen kořen.
- Dá se ukázat, že tato posloupnost obsahuje T_α optimální stromy pro daná α .
- $\alpha = 0$ – neprořezáváme, necháme T_0 , pro $\alpha = \infty$ necháme jen kořen, vhodné α zvolíme na základě krosvalidace, $\hat{\alpha}$ minimalizuje krosvalidační chybu RSS, vydáme model $T_{\hat{\alpha}}$.

Krosvalidace (Crossvalidation)

Snažím se odhadnout chybu z více, než jednoho testu, tím dostat stabilnější odhad.

- Rozdělím (stratifikovaně) data na daný počet stejných částí.
- Jednu část zadržím pro učení, na ostatních naučím model a otestuji schovanou částí.
- Tohle provedu s každou částí, jednotlivé chyby zprůměruji.
- Zkušenost velí dělit na 10 částí, **tenfold** crossvalidation.

Slabší stránky CART

- nestabilita stromů: pro trochu jiná data mohu dostat naprosto jiný strom; lze zmírnit průměrováním přes více stromů (bagging)
- řezy pouze kolmo na osy; dalo by se rozšířit, ale pak se hůře optimalizuje
- výsledek není hladký, ale schodovitý. Pokud předpokládám hladkou cílovou funkci, je to divné. MARS bude hladký (Multivariate Adaptive Regression Splines)
- špatně podchytí aditivní strukturu

$$Y = c_1 I(X_1 < t_1) + c_2 I(X_2 < t_2) + \dots + c_k I(X_k < t_k) + \epsilon$$

po prvním dělení bude v různých větvích dělit různě, globální vzorec z toho nikdo nevykuká a nejspíš ani strom neodhalí (pro nedostatek dat u listů). MARS toto zvládne.

Pravidla pro regresi

“Hon na maxima” PRIM = Bump Hunting Patient Rule Induction Method

- iterativně hledáme oblasti, kde je Y velké; pro každou oblast vytvoříme pravidlo
- CART po cca. $\log_2(N) - 1$ řezech přijde o data, PRIM si může dovolit cca. $-\frac{\log(N)}{\log(1-\alpha)}$.
Pro $N = 128$ a $\alpha = 0.1$ to je 6 a 46 resp. 29, protože počty pozorování musí být celé.
- Např. XOR s matoucími nezávislými dimenzemi PRIM hravě zvládne, CART má problém.

PRIM Pravidla pro regresi

1. Vezmi všechna data a prostor obsahující všechna data, $\alpha = 0.05$ nebo 0.10
2. Najdi X_j a jeho horní či dolní okraj, jehož oříznutím o $\alpha \cdot 100$ procent pozorování vede k největší střední hodnotě zbytku.
3. Opakuj 2. dokud zbývá aspoň 10 pozorování.
4. Rozšiř oblast v libovolném směru, pokud to zvýší střední hodnotu.
5. Vyber z oblastí generovaných 1 až 4 tu (ten počet pozorování), který je nejlepší při krosvalidaci. Nazvěme odpovídající oblast B_1 .
6. Odstraníme data v B_1 z databáze a opakujeme 2 až 5, vytvoříme B_2 , atd., dokud je libo.

Lineární regrese

- Cíl: aproximovat funkci $f(x)$, kde x je n -rozměrný vektor, pomocí lineární funkce

$$\hat{y} = \hat{\beta}_0 + \sum_{j=1}^n x_j \hat{\beta}_j$$

- Není-li $X^T X$ singulární, dostaneme jednoznačné řešení

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

- a odhad \hat{y} pro dané x_i je $\hat{y}(x_i) = x_i^T \hat{\beta}$.

MARS Multivariate Adaptive Regression Splines

- Zobecnění postupné lineární regrese i zobecnění CART
- model je tvaru

$$f(X) = \beta_0 + \sum_{m=1}^M \beta_m h_m(X)$$

kde $h_m(X)$ je funkce z \mathcal{C} nebo součin libovolného počtu funkcí z \mathcal{C}

- pro pevně zvolenou h_m spočteme koeficienty β_m standardní lineární regrese (minimalizujeme součet kvadrátů reziduí)
- funkce h_m volíme postupně.

MARS – pokračování

- Pro každou vstupní proměnnou a každý datový bod vytvoříme dvojici funkcí báze
- $(x - t)_+$ a $(t - x)_+$, kde to $+$ značí nezápornou část, zápornou ořízneme nulou. Tuto dvojici nazýváme zrcadlový pár.
- máme tedy množinu funkcí

$$\mathcal{C} = \{(X_j - t)_+, (t - X_j)_+\}_{t \in \{x_{1,j}, x_{2,j}, \dots, x_{N,j}\}, j=1,2,\dots,p}$$

- tedy $2Np$ funkcí, jsou-li všechny vstupní hodnoty různé.

MARS – volba báze

- Začneme s konstantní $h_0 = 1$, funkci přidáme do modelu $\mathcal{M} = \{h_0\}$.
- Uvažujeme součin každé funkce z modelu \mathcal{M} s každou dvojicí v \mathcal{C}

$$\hat{\beta}_{M+1}h_\ell(X)(X_j - t)_+ + \hat{\beta}_{M+2}h_\ell(X)(t - X_j)_+, h_\ell \in \mathcal{M}$$

vybereme ten, co nejvíce sníží trénovací chybu (vždy dopočteme regresní koeficienty $\hat{\beta}$).

- Opakujeme, dokud \mathcal{M} nemá předem daný počet členů
- protože je model často přeučtený, zas ubíráme, vždy tu, co nejméně zvýší trénovací chybu. Máme tak posloupnost modelů \hat{f}_λ pro různé počty parametrů λ .

-
- vybereme tu, co minimalizuje zobecněnou krosvalidaci (abychom se nemuseli namáhat krosvalidaci počítat)

$$GCV(\lambda) = \frac{\sum_{i=1}^N (y_i - \hat{f}_\lambda(x_i))^2}{(1 - M(\lambda)/N)^2}.$$

- $M(\lambda)$ je počet efektivních parametrů v modelu, tj. počet funkcí h_m (označím r) plus počet uzlů K (tj. použitých datových bodů t), zkušenost radí násobit počet uzlů třikrát, tj. $M(\lambda) = r + 3K$.

Z MARSu CART

- I když se nezdá, souvisí.
- místo po částech lineární zvolíme po částech konstantní funkce $I(x - t > 0)$ a $I(x - t \leq 0)$
- Pokud funkci modelu h_m použijeme pro násobení, tak jí z modelu vymažeme, aby nešla znovu použít. Tím zajistíme, že se použije maximálně pro jedno násobení, tj. jen pro jedno dělení – a máme binární strukturu stromu.
- a máme CART.

Strukturované regresní modely

- Minimalizaci RSS splňuje nekonečně mnoho funkcí interpolujících naměřené hodnoty
- to ale nebývá vhodné pro predikci (velká očekávaná a testovací chyba).
- Omezíme přípustné funkce, ve zvolené třídě jednoznačné řešení.
- Chceme "jednoduché" funkce, které nejsou divoké na malých okolích ve vstupním prostoru.
- Velikost okolí bývá parametrem, čím větší, tím větší restriktce.

Penalizace za složitost

- Míru chyby RSS přímo upravíme penaltou $J(f)$ za složitost modelu

$$PRSS(f; \lambda) = RSS(f) + \lambda J(f)$$

- např. Hřebenová (ridge) regrese penalizuje nenulové složky β

$$\hat{\beta}^{ridge} = \operatorname{argmin}_{\beta} \sum_{i=1}^N (y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j)^2 + \lambda \sum_{j=1}^p \beta_j^2.$$

- kubický vyhlazující splajn pro jednorozměrný vstup

$$PRSS(f; \lambda) = \sum_{i=1}^N (y_i - f(x_i))^2 + \lambda \int [f''(x)]^2 dx.$$

- $\lambda \geq 0$ řídí velikost penalty, $\lambda = 0$ vede k interpolaci, $\lambda = \infty$ dovolí jen lineární funkce x .

Jádrové funkce a lokální regrese

- Nejbližší sousedi tvoří "skokovou" aproximující funkci, není to hezké a je to zbytečné
- specifikujme okolí jádrovou funkcí $K_\lambda(x_0, x)$ určující, nakolik je x v okolí x_0 , např. gausovské jádro

$$K_\lambda(x_0, x) = \frac{1}{\lambda} \exp \left[-\frac{\|x - x_0\|^2}{2\lambda} \right]$$

- y pak odhadneme

$$\hat{f}(x_0) = \frac{\sum_{i=1}^N K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^N K_\lambda(x_0, x_i)}$$

-
- nebo "naučíme" parametrickou funkci f_θ minimalizací RSS

$$RSS(f_\theta, x_0) = \sum_{i=1}^N K_\lambda(x_0, x_i) (y_i - f_\theta(x_i))^2$$

- $f_\theta = \theta_0$ vede na odhad viz výše (Nadaraya–Watson)
- $f_\theta = \theta_0 + \theta_1 x$ dává lokálně lineární regresní model.
- Nejbližší sousedi mají jádrovou funkci závislou na datech, je 1 ve vzdálenosti menší či rovné vzdálenosti *ktého* souseda, jinak nula.

Báze funkcí a "slovníkové" metody

- vezmeme bázi funkcí $\{h_m(x)\}$ a modelujeme f jako jejich lineární kombinaci, θ značí parametry:

$$f_{\theta}(x) = \sum_{m=1}^M \theta_m h_m(x)$$

- vejde se sem lineární a polynomiální expanze
- splajny a součiny tenzorů
- báze radiálních funkcí
- jednovrstvé dopředné neuronové sítě
- a další.