Vypracoval: Ondřej Přibyl
e-mail: xpribyl1@lab.felk.cvut.cz

Otázka 4/2; Zadání:

Adaptace a učení, jednoduché algoritmy učení klasifikátoru a trénování neuronových sítí. Metody strojového učení, indukce rozhodovacích stromů, případové usuzování (CBR), usuzování na základě analogií. Učení se zapomínáním.

1.Adaptace a učení

Učení: Proces samočinné optimalizace. Dlouhodobý proces, který se skládá z postupných kroků (adaptace). Jedná se o analogii živých organismů.

1.1. Nastavování klasifikátoru učením

Základním předpokladem pro nastavování klasifikátorů je znalost apriorních pravděpodobností a hustot pravděpodobností. Většinou je však neznáme úplně. Máme však k dispozici **trénovací množinu,** tj. soubor vektoru příznaků se známou příslušností ke třídě. Na jejím základě není problém apriorní pravděpodobnost určit jako četnost výskytu vektoru příznaků v jednotlivých třídách. Při odhadu hustoty pravděpodobnosti můžeme rozlišit dva případy:

- známe tvar hustoty pravděpodobnosti, ale neznáme parametry – parametrické metody
- neznáme tvar hustoty pravděpodobnosti – neparametrické metody.

V obou případech využíváme vektory z trénovací množiny s předem známou klasifikací. Jedná se proto o **učení s učitelem**. (Viz. kapitola 1.1.1)

Někdy se stane, že informaci o správné klasifikaci nemáme. Metody, které vedou k nastavení klasifikátoru v tomto případě se nazývají **učení bez učitele**. Nejčastěji se používají *metody shlukové analýzy.* Ty umožňují nastavení klasifikátoru bez znalostí o správné klasifikaci, ale případně i bez znalosti o počtu tříd.(Viz. kapitola 1.1.2)

1.1.1. Matematická formulace učení

Trénovací množina $\{X_K, \Omega_K\}_{k=1}^N$, obsahuje vektory příznaků X_K a jim odpovídající správnou klasifikaci Ω_K (informaci od učitele). Trénovací množina by měla být náhodným a nezávislým výběrem a měla by zcela popisovat prostor příznaků.

Důležitými vlastnostmi učících systémů je:

- *Sekvenčnost* – parametr q se vypočítá pomocí rekurentních vzorců. V případě, že trénovací množinu doplníme novými vzorky, není potřeba celý proces učení opakovat: $q(K+1) = f_{K+1}(q(K), x(K+1), \Omega(K+1))$
- *Induktivnost* – posloupnost $q(K)$ by měla konvergovat k q^* (optimum).

Činnost učitele se dá popsat následujícím algoritmem. Po předložení $\{[X_K, \Omega_K]\}_{k=1}^N$ je nalezen parametr q^* tak, že

Vypracoval: Ondřej Přibyl
e-mail: xpribyl1@lab.felk.cvut.cz

$$J(q^*) = \min J(q)$$

Naučený systém je takový systém, který nejlépe aproximuje učitele ve smyslu minima střední ztráty.

1.1.2. Shluková analýza

Úkolem shlukové analýzy je nalézt shluky vektorů příznaků, tj. skupiny jejichž prvky jsou si vzájemně blízké. Shluková analýza se používá v případech, že potřebujeme nahlédnout do struktury dat.

Výchozí neklasifikovaná data tvoří množinu *T*. Tu se snažíme rozložit na co možná „nejkompaktnější“ navzájem disjunktní podmnožiny. Úspěšného výsledku tedy může být dosaženo jen v případě vhodných dat, která v příznakovém prostoru tvoří shluky.

Před prováděním shlukové analýzy je nezbytné zvolit některou z měr. Tyto míry ve skutečnosti určují „podobnost“ vzorků.

- *Míra podobnosti mezi dvěma obrazy*, která je nezbytná jako základní míra v obrazovém prostoru. Obvykle se volí Euklidovská metrika, či Mahalanobisova vzdálenost.
- *Míra vzdálenosti mezi dvěma shluky*, která je důležitá při postupném vzájemném slučování shluků, např. při hierarchickém shlukování. Nejčastěji se používá průměrná vzdálenost mezi obrazy shluku, vzdálenost ideálních obrazů středů shluků (etalonů), minimální či maximální vzdálenost mezi dvěma obrazy různých shluků.
- *Míra pro ohodnocování kvality rozdělení obrazů do shluků* (Kriteriální funkce *J*). Kriteriální funkce musí dosahovat svého extrému v případě, že vytvořené shluky jsou shodné s přirozenými shluky. Nejčastěji se používá kritérium minima kvadrátu odchylky.

Metody shlukové analýzy je možné rozdělit do dvou skupin:

1. *Metody iterativní optimalizace.*
   Základní myšlenka iterování vychází z předpokladu, že je dáno nějaké počáteční nastavení. Jednotlivé obrazy jsou potom zkusmo přesouvány z jedné třídy do druhé, přičemž se sleduje zda při přesunu došlo ke zlepšení kriteriální funkce. V takovém případě dojde k přerozdělení. Tato metoda vede jen k lokální optimalizace rozdělení neboť je závislá na počátečním nastavení.
2. *Metody hierarchického shlukování.*
   Na počátku shlukování považujeme každý obraz za samostatný shluk, existuje tedy *K* shluků. V jednotlivých krocích se spojují vždy dva nejbližší shluky. (ve smyslu příslušné míry podobnosti). Tak se postupně vytvoří shluk jediný, či příslušný počet.Tato metoda je optimální a je vhodná na případy kdy není znám optimální počet tříd.

Vypracoval: Ondřej Přibyl
e-mail: xpribyl1@lab.felk.cvut.cz

1.2. Učení neuronových sítí

Parametry neuronových sítí se obvykle na počátku nastaví náhodně. Cílem procesu učení je změnit jednotlivé parametry tak, aby aktuální odezva na daný vzor odpovídala požadované odezvě.

Gradientní metody

Tyto metody vycházejí z vhodné definice kriteriální (chybová) funkce *J*. Jedná se o hledání extrému této funkce pomocí výpočtu gradientů podle jednotlivých parametrů. Typickým příkladem je minimalizace globální chyby sítě *GE* definované jako:

$$GE = \sum_{T} \sum_{i=1}^{n} \left(y_i - d_i \right)^2 ,$$

kde d_i je požadovaná odezva na daný vstupní vektor *x* v i-tém výstupním neuronu a y_i je jeho aktuální odezva. Vnitřní suma se bere přes všechny výstupní neurony, vnější přes všechny příklady z trénovací množiny.

Po předložení vzoru vznikne obecně na i-tém výstupním neuronu chyba y_i-d_i, úkolem adaptace je v jednom kroku je upravit hodnoty všech vah aby se tato chyba zmenšila. K tomu se využívá parciální derivace globální chyby GE podle zvolené váhy *w* a tato váha se upraví o přírůstek

$$\Delta w = -\eta \frac{\partial GE}{\partial w}$$

pro jistou hodnotu parametru η. Takto se postupně upraví všechny váhy sítě.

Zatímco se aktivita sítě šíří zdola nahoru, chyba a její důsledky pro úpravu vah se šíří shora dolů. Proto tento algoritmus dostal jméno **back-propagation.** Nevýhodou metody Back-propagation je, že hledá lokální extrém chybové funkce GE.

Další metody

Abychom se vyhnuli sklouznutí do lokálního extrému, je třeba použít některé jiné metody. Jsou to většinou metody založené na stochastickém základě.

Na příkladu z přírody vznikly takzvané **genetické algoritmy**. Je-li možné smysluplně popsat jakýkoli objekt třídy vhodným deskripčním řetězcem a přiřadit mu jistou kriteriální funkci (fitness function) pak je možné použít právě tyto algoritmy. Stručně lze tento algoritmus popsat několika kroky:

1. Vytvoříme počáteční generaci.
2. Podle zvolených pravděpodobností následuje křížení a mutace jednotlivých jedinců.
3. Z takto vytvořené množiny vybereme novou generaci (selekce) a celou proceduru opakujeme.

Vypracoval: Ondřej Přibyl
e-mail: xpribyl1@lab.felk.cvut.cz

Aby se zabránilo sklouznutí do lokálního extrému, není vhodné vybírat pouze nejlepší jedince. S jistou nenulovou pravděpodobností můžeme vybrat i jedince méně vhodné, kteří po zkřížení mohou vygenerovat lepší potomky.

Dalším algoritmem je algoritmus **simulovaného žíhání**. Ten vznikl jako analogie procesu chlazení roztaveného křemene. Tato minimalizační metoda je založena na parametru *simulovaná teplota* τ . Hodnota τ se v průběhu procesu snižuje a tím se snižuje i pravděpodobnost vyskočení ze zatím dosaženého minima a hledání jinde.

2. Strojové učení

2.1. Popis základních pojmů

Vycházejme z příkladu popsaného v [1] str. 168, kdy se snažíme vysvětlit robotu význam slova „pták“. K dispozici máme příklady kos, foxteriér, moucha, vrána.

V případě strojového učení jsme v roli učitele, který předkládá žáku příklady ilustrující určitý pojem (koncept). Žák je nadán inteligencí, ale chybí mu základní znalosti v dané problematice. Příklady jsou popsány pomocí predikátů, například : ma_kridla, snasi_vejce, ... Žák se snaží najít takový popis pojmu „pták“, který se hodí na všechny exempláře tohoto živočicha, ale na žádného jiného.

Na počátku máme jediný pozitivní příklad, a to *kos*a. Výskyt negativního příkladu (*foxteriér*) umožní nalézt predikáty, které odlišují ptáka od „ne-ptáka“ (*ma_kridla, ma_zluty_zobak, zpiva, snasi_vejce*). Tyto predikáty nazveme **diskriminačními predikáty**. Jelikož je tento popis příliš podrobný, žák vybere jen jeho část – *ma_kridla*. Tento krok se obecně nazývá **generalizací**, čili zobecněním. Následující negativní příklad – *moucha* – však ukáže špatnou volbu predikátu. Žák v dalším kroku provede **specializaci** popisu na konjunkci *ma_kridla* a *ma_zluty_zobak*. Po přidání dalšího pozitivního příkladu – *vrány* – se žák dostane do dlepé uličky a není možné udělat žádnou generalizaciči specifikaci , která by umožnila vránu do popisu zahrnout. Proto žák sáhne k novému popisu pomocí predikátu *snasi_vajicka_se_skorapkou.* Tento popis již vyhovuje všem příkladům.

Podstata operátorů generalizace a specifikace může být buď deduktivní nebo induktivní. **Dedukce** je takový způsob inference (odvozování), který zachovává pravdivost, **indukce** uchovává nepravdivost.

Příkladem induktivního úsudku je například věta: “Jestliže pan A pije, bude opilý“. Jestliže levá strana výroku není pravdivá, není pravdivá ani strana pravá. Na druhé straně samotný fakt pití ještě nemusí vést k opití. Pravdivost levé strany nevede nutně k pravdivosti strany levé.

Častějším příkladem však bývá deduktivní usuzování: “Rozpojený vypínač má vždy za následek zhasnutí žárovky.“ Ta ovšem může zhasnout i z jiného důvodu. Pravdivost levé strany se zachovává, nicméně její nepravdivost nemusí mít na hodnotu na pravé straně vliv.

Dalšími pojmy jsou **abstrakce** a **konkretizace**. O abstrakci hovoříme tehdy, když z původního popisu ubereme část informace. Například místo „pan A měří 183 cm“

Vypracoval: Ondřej Přibyl
e-mail: xpribyl1@lab.felk.cvut.cz

použijeme popis „pan A je vysoký“. Opakem abstrakce je konkretizace, znamená tedy doplnění informací.

Není vždy možné nalezení přesného popisu. Jazyk nemusí být natolik bohatý aby odlišil pozitivní příklady od negativních. V takovém případě si musíme vybrat jaký typ nepřesností jsme ochotni akceptovat. Zda chceme popis konzistentní či kompletní.

Konzistentní popis jednoznačně diskriminuje pozitivní příklady od negativních, i když není schopen pokrýt všechny pozitivní příklady.

Kompletní popis pokryje všechny pozitivní příklady i za cenu zahrnutí některých příkladů negativních.

Problém učení z příkladů může být považován za problém prohledávání stavového prostoru všech možných popisů daného pojmu. Zadání je tedy formulováno takto:

je dáno:
- množina příkladů,
- popis příkladů v daném jazyku,
- klasifikace příkladů na pozitivní a negativní,
- znalosti omezující prostor prohledávání,

najdi:
- popis daného pojmu.

2.2. Induktivní tvorba rozhodovacích stromů

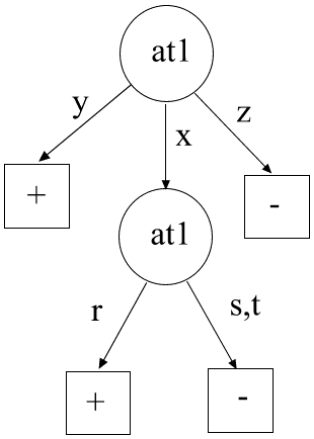
Existují algoritmy, které usnadňují učení z příkladů. Nejčastěji se používá metoda TDIDT (*Top-Down Induction of Decision Trees*). Výsledný popis má tvar rozhodovacího stromu, kde listy obsahují ohodnocení, zatímco ostatní uzly obsahují testy hodnot atributů. Při klasifikaci procházíme stromem od shora dolů a provádíme postupně testy předepsané v uzlech.

	at1	**at2**	**at3**	**klasifikace**
o1	x	r	m	+
o2	y	r	n	+
o3	y	s	n	+
o4	x	s	m	-
o5	z	t	n	-
o6	z	r	n	-

tabulka 1:Vzorová množina příkladů popsaných pomocí tří atributů

V tabulce 1 vydíme možné zadání vzorové množiny. Výsledný rozhodovací strom je na obrázku 1. Metoda TDIDT má dvě fáze: vytvoření rozhodovacího stromu a zjednodušování (prořezávání) rozhodovacího stromu.

Vypracoval: Ondřej Přibyl
e-mail: xpribyl1@lab.felk.cvut.cz



obrázek 1: Rozhodovací strom vytvořený z příkladů uvedených v tabulce 1.

2.2.1. Tvorba rozhodovacího stromu

Nejprve je potřeba najít atribut, který obsahuje největší množství informace. Tento atribut se stane kořenem stromu.

V dalším kroku si rozdělíme množinu příkladů na tolik podmnožin, kolik je hodnot kořenového atributu. Poté v každé z těchto podmnožin vyhledáme opět nejvýznamnější atribut a takto rekurzivně pokračujeme než vyčerpáme atributy či příklady, či pokud není splněno předem dané kritérium.

Pro výběr nejvýznamnějšího atributu se nejčastěji používá měření množství energie pomocí entropie. Podle Shannonovy věty platí pro entropii j-té podmnožiny vztah

$$H_j = -p_1 \log_2 p_1 - p_2 \log_2 p_2 ,$$

kde p_1, resp. p_2 je poměr pozitivních, resp. negativních příkladů v *j-té* podmnožině k celkovému počtu prvků v této podmnožině. Celková velikost entropie je dána váženým součtem entropií jednotlivých podmnožin

$$H = \sum_{j=1}^{K} P_j H_j ,$$

kde K je počet podmnožin indukovaných daným atributem, H_j je entropie *j-té* podmnožiny a P_j je poměr velikosti *j-té* podmnožiny k množině všech příkladů.

Je zřejmé, že zvolíme takový atribut, jehož entropie H je minimální. Při konstrukci rozhodovacího stromu (obr. 1) z dat obsažených v tabulce 1 byly použity tyto výpočty:

$$H(at1) = \frac{1}{3} H_X + \frac{1}{3} H_Y + \frac{1}{3} H_Z = \frac{1}{3} + 0 + 0 = \frac{1}{3},$$

$$H(at2) = \frac{1}{2} H_R + \frac{1}{6} H_T + \frac{1}{3} H_S = 0{,}46 + 0 + 0{,}33 \approx 0{,}8 ,$$

Vypracoval: Ondřej Přibyl
e-mail: xpribyl1@lab.felk.cvut.cz

$$H(at3) = \frac{2}{3} H_N + \frac{1}{3} H_M = \frac{2}{3} + \frac{1}{3} = 1,$$

kde například

$$H_X = -[\frac{1}{2}\log_2 \frac{1}{2} + \frac{1}{2}\log_2 \frac{1}{2}] = 1.$$

Atribut *at1* má nejmenší entropii (rovnou 1/3) a proto byl vybrán za kořen rozhodovacího stromu.

2.2.2. Prořezávání rozhodovacích stromů

V praxi máme snahu výsledný strom nějak zjednodušit. Takové zjednodušení se nazývá **prořezávání**. Jedním důvodem této snahy je zjednodušení interpretace, druhým je možnost zlepšení přesnosti. Příklady často obsahují i šum. Učící systém má snahu přidat větve i pro chybné příklady. Tento problém se nazývá neúměrná přesnost čili *overfitting*.

2.3. Učení z klasifikovaných příkladů

Algoritmus TDIDT je velmi jednoduchý, postrádá však možnost využití již dříve nabytých vědomostí. Typičtější metodou je proto **algoritmus AQ**, který je opět určen pro učení z příkladů popsaných pomocí hodnot atributů. Lze jej shrnout do pěti bodů:

1. Rozděl množinu příkladů na dvě podmnožiny: množinu PE obsahující pouze pozitivní příklady a množinu NE obsahující příklady negativní.
2. Vyber z množiny PE náhodně jeden prvek a označ jej *s* (jádro).
3. Nalezni všechny maximální generalizace popisu jádra *s*, přičemž limitem je množina NE. Generalizace popisu nesmí pokrýt žádný negativní příklad.
4. Podle zvoleného preferenčního kriteria vyber nejlepší z těchto popisů a zařaď jej do množiny popisů.
5. Pokud množina popisů pokrývá všechny prvky PE, ukonči práci.

Výsledkem je pak disjunkce všech nalezených popisů. V opačném případě vyber nové jádro z dosud nepokrytých pozitivních příkladů.

Maximální generalizace – pokrývá co největší počet příkladů.

Preferenční kritérium – přednost je dána těm atributům, které jsou lépe dostupné („levnější“). Například u lékařských dat je dostupnější údaj o teplotě než údaj o sedimentaci.

2.4. Další metody

2.4.1. Učení na základě analogií.

Prohledávání neúměrně velkých bází dat se promítne do rychlosti algoritmu. Způsobem omezení prohledávácího prostoru je využívání analogií. Třeba při studiu elektrického pole

Vypracoval: Ondřej Přibyl
e-mail: xpribyl1@lab.felk.cvut.cz

nám pomůže, když víme, že se elektrické pole chová analogicky k magnetickému poli. Schopnost analogií zmenšit prohledávací prostor je veliká a proto se mnoho vědců domnívá, že jejich nalézání a využívání je klíčem k podstatě umělé inteligence.

2.4.2. Učení založené na příkladech

Metody, které zde byly uvedeny, předpokládají, že z příkladů se odvodí rozhodovací pravidla a příklady se poté zapomenou. Někdy je však vhodné alespoň některé příklady v paměti uchovat, například pro pozdější snazší aktualizaci znalostí. Krajním přístupem je proto učení založené na příkladech (Case Based Reasoning), kdy se v paměti uchovávají pouze rozsáhlé případy a při usuzování se hledají analogie mezi příklady v paměti a studovanými vnějšími příklady. Učení spočívá v rozpoznání obzvlášť typických příkladů, které se uloží.

Literatura:

[1] Umělá inteligence 1, Mařík, Štěpánková, Lažanský; Academia, Praha 1993
[2] Umělé neuronové sítě, Mirko Novák a kol., C.H.Beck, Praha 1998
[3] přednášky RPZ
[4] přednášky SOC