

1 UVOD V UMETNO INTELIGENCO

1.1 TURINGOV TEST

Opazovalec po pogovru ne more lociti racunalnika od cloveka.

2 STROJNO UCENJE

2.1 PROBLEMSKI PROSTOR, OCENJEVANJE ZNANJA

2.2 EVALVIRANJE HIPOTEZ

Pomembni kriteriji:

- **konsistentnost** hipotez z primeri (ucnimi)
- **splosnost** (tocnost za nevidene primere)
- **razumljivost** hipotez

Ocenjevanje uspesnosti pri klasifikaciji na podlagi njihove **tocnosti**:

TP - true positive, FP - false positive, FN - false negative, TN - true negative

Klasifikacijska tocnost = $\frac{TP + TN}{TP + TN + FP + FN} = \frac{TP + TN}{N}$

Napaka 1. tipa = FP, napaka 2. tipa = FN

Obcutljivost/senzitivnost = $TPR = \frac{TP}{TP + FN}$

2.3 GRADNJA ODLOCITVENIH DREVES

Informacijski prispevek Gain(A) = I - I_{res}(A), I=H(C)

$I_{res} = - \sum_{v_i \in A} p_{v_i} \sum_c p(c|v_i) \log_2 p(c|v_i)$

Za koliko se entropija zmanjsa po delitvi z Atributom A.

Razmerje inofracijskega prispevka atributa A:

IGR(A) = $\frac{Gain(A)}{H(A)}$

2.3.1 TDIDT (TOP DOWN INDUCTION DECISION TREE) ALGORITHM

Pozresen algoritem, ki **lokalno** izbira najbolsi atribut.

- kratkoviden algoritem

2.3.2 BINARIZACIJA ATRIBUTOV

Aletarnativa za reševanje problematike z vecvrednostnimi atributi:

Primer: barve ∈ rdeca, rumena, zelena, modra

Strategije, razbijemo v dve množici:

- rdeca, rumena, zelena, modra

- rdeca, rumena, zelena, modra

Prednost: manjše vejanje drevesa.

2.4 UCENJE IZ SUMNIH PODATKOV (REZANJE)

tocnost t...verjetnost pravilnosti klasifikacije

napaka e ... 1 - t

relativna frekvenca $p = \frac{n}{N}$

m-ocena $p = \frac{n + p_a \cdot m}{N + m}$

m... koliko zaupam apriorni verjetnosti

p_a apriorna verjetnost (domenski ekspert lahko pove)

Laplacova ocena verjetnosti $p = \frac{n+1}{N+k}$

k...stevilo vseh moznih razredov

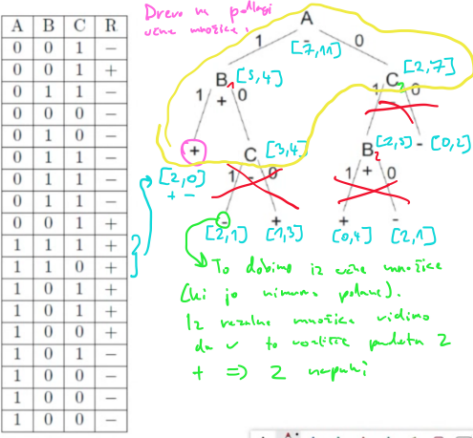
2.4.1 REP (REDUCED ERROR PRUNNING)

Dela dobro ce imamo veliko rezalno množico.

Obicajno uporabljamo relativno frekvenco za ocenjevanje verjetnosti.

G(v) = #napak_T - #napak_v

G(v) ≥ 0 ⇒ rezemo podrevo



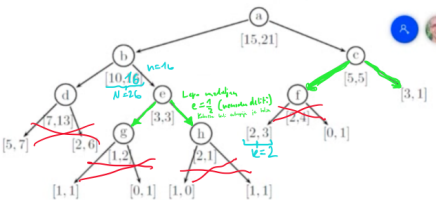
e(C) = 3

e_T = 2 + 3 = 5

G(C) = 5 - 3 = 2 ≥ 0 → rezemo

2.4.2 MEP (MINIMAL ERROR PRUNNING)

e...staticna napaka, E...vzvrtna napaka, e ≤ E → rezemo podrevo



(Laplace)

e_L(d) = 1 - t = 1 - $\frac{13+1}{20+2} = 0.363$

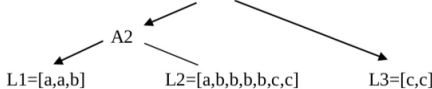
E_L(d) = 12/20 · e_L(d₁) + 8/20 · e_L(d₂) = $\frac{12}{20} \cdot (1 - \frac{7+1}{12+2}) + \frac{8}{20} \cdot (1 - \frac{13+1}{20+2})$

2.5 OCENJEVANJE USPEŠNOSTI MODELOV

tocnost t ... verjetnost pravilnosti klasifikacije

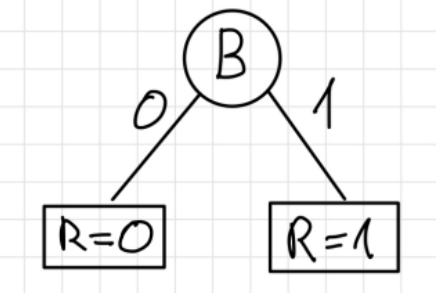
Laplacova ocena verjetnosti $p = \frac{n+1}{N+k}$

k...stevilo vseh moznih razredov



t_{L1} = $\frac{2+1}{3+3} = 0.5$, t_{L2} = $\frac{4+1}{7+3} = 0.5$, t_{L3} = $\frac{2+1}{2+3} = 0.6$

tocnost drevesa: t_D = 3/12 · 0.5 + 7/12 · 0.5 + 2/12 · 0.6 = 0.5167



e = 1 - (P(B=0)P(R=0|B=0) + P(B=1)P(R=1|B=1))

2.6 OBRAVNANVA MANKAJOCIH ATRIBUTOV, NAVINI BAYESOV KLASIFIKATOR

2.6.1 NAIVNI BAYES

Ce poznamo razred, kam klasificiramo ce nepoznamo atributov:

Klasifikator: $\text{argmax}_{c \in C} P(c) \prod_{i=1}^n P(x_i|c)$

c...razred, x_i...atributi

Verjetnost::

$P(C = c|x_1, \dots, x_n) = \frac{P(C = c)P(X_1 = x_1|C = c)P(X_2 = x_2|C = c) \dots}{P(X_1 = x_1)P(X_2 = x_2) \dots}$

Primer moski: visina ≥ 175, teza ≥ 65, spol = M

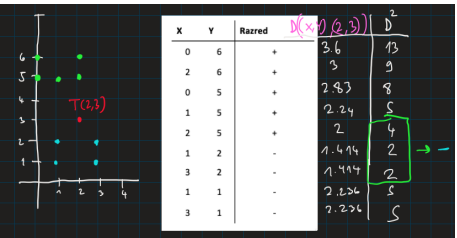
X\Y	Razred A	Razred B
p _a	$P(A) = \frac{2}{3}$	$P(B) = \frac{1}{3}$
spol	$P(M A)$	$P(M B)$
visina	$P(V \geq 175 A)$	$P(V \geq 175 B)$
teza	$P(T \geq 65 A)$	$P(T \geq 65 B)$
$P(y) \prod_{i=1}^n P(x_i y)$

2.6.2 NOMOGRAMMI

Ciljni razred C = c_T

$X_{X_i=x_j} = \ln \left(\frac{P(X_i = x_j|C = c_T)}{P(X_i = x_j|C = \bar{c}_T)} \right)$

2.7 K-NAJBILIZJIH SOSEDOV



3 VRSTE UCENJA

3.1 NADZOROVANO UCENJE (SUPERVISED LEARNING)

Ucni primeri so podani/oznaceni kot vrednosti vhodov in izhodov.

(x₁, y₁), (x₂, y₂), ..., (x_N, y_N)

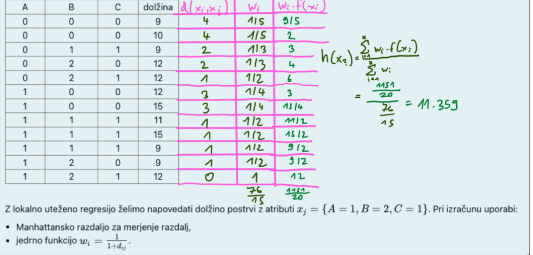
x_i... atributi, y_i... ciljna spremenljivka

Locimo dve vrsti problemov:

1. **Klasifikacijski problemi** - y_j **diskretna**
2. **Regresijski problemi** - y_j **zvezna**

3.1.1 LOKALNO UTEZENA REGRESIJA

$$h(\vec{x}_i) = \frac{\sum_{i=1}^k w_i \cdot f(\vec{x}_i)}{\sum_{i=1}^k w_i}, w_i(d) \dots \text{utez}$$



3.1.2 REGRESIJSKA DREVEZA

Linearna regresija je poseben primer regresijskega drevesa.

V listih regresijskega drevesa vcasih napovemo kar povprečno vrednost.

3.2 NENADZOROVANO UCENJE (UNSUPERVISED LEARNING)

Ucni primeri niso oznaceni (nimajo ciljne spremenljivke), **ucimo se vzorcev v podatkih**, (npr. grucenje)

3.2.1 HIERARHICNO GRUCENJE

Poveze po podobnosti med primeri, primer zacne kot samostojna gruca, na koncu vsi primeri pripadajo eni gruci

Dendrogram: drevo, ki predstavlja grucenje.

Single-linkage: povezava med grucami je najkrajše razdalje med primeroma iz različnih gruc.

Complete-linkage: povezava med grucami je najdaljša razdalja med primeroma iz različnih gruc.

Average-linkage: povezava med grucami je povprečna razdalja med primeroma iz različnih gruc.

3.2.2 K-MEANS

1. V prostor dodamo k centroidov, ki predstavljajo gruce.
2. Izracunamo ketri centroid je najblizji vsakemu primeru.
3. Izracunamo nove centre gruc = $\frac{1}{|G|} \sum_{i \in G} x_i$
4. Ponovimo korake 2 in 3 dokler se centri ne premaknejo.

3.3 SPODBUJEVALNO UCENJE - REINFORCEMENT LEARNING

Intelligentni agent se uci iz zaporedja **nagrad in kazni**

3.4 OCENJEVANJE UCENJA

3.4.1 PRECNO PVERJANJE

Poseben primer **vekratnega ucenja** in testiranja

k-kratno precno preverjanje

- celo ucno množico razbij na k disjunktnih podmnozic
- za vsako od k podmnozic:
 - uporabi **mnozico** kot **testno množico**
 - uporabi **preostalih k-1 množic** kot **ucno množico**
- povpreci dobljenih k ocen tocnosti v koncno oceno

Pri precnem preverjanju uporabimo vse podatke za testiranje in vse za ucenje

Metoda **leave one out** je poseben primer precnega preverjanja Imamo dve hipotezi A in B. Izkase se, da A bolje napoveduje na uc-nih podatkih B pa na testnih. Potem je B verjetno boljše hipoteza.

4 PREISKOVANJE

4.1 NEINFORMIRANI PREISKOVALNI ALGORITMI

4.1.1 ISKANJE V SIRINO

4.1.2 ISKANJE V GLOBINO

Izboljsave:

- Iskanje s sestopanjem

- **depth-limited-search** (vnaprej definiramo globino l (dolocimo preko domenskega znanja))

4.1.3 ITERATIVNO POGLABLJANJE

problem gobinsko omejenega iskanja -> nastavitve meje l Mejo l postopoma povečujemo za 1, dokler ne najdemo rešitve.

- **popolnost:** Da
- **optimalnost:** Da
- **casovna zahtevnost** $O(b^d)$
- **prostorska zahtevnost** $O(bd)$

Bolje od iskanja v globino/sirino

4.1.4 DVOSMERNO ISKANJE

Ideja: pognati vzporedni iskanji od zacetka do cilja in od cilja do zacetka.

Motivacija:

Implementacija dvosmernega iskanja

- ciljno vozlisce mora biti znano
- originalni problemski prostor preslikamo v dvosmerni prosto stanj E1, E2 dosegljiv iz E in S1,S2,S3 dosegljiv iz S (S,E) -> (S1,E1), (S1,E2), (S2,E1), (S2,E2)... Vozlisce (Si, Ei) je v dvosmernem prostur ciljo vozlisce ce velja E=S (soda dolzina na isto mesto pridemo iz obeh strani) ali S->E (liha pot sosednja)

4.1.5 CENOVNO - OPTIMALNO ISKANJE

- posplositev iskanja v sirino (iskanje v sirino je optimalno, ce so cene vseh povezav enake 1)

- dijkstra basically (sam do zadnga noda)
- <https://stackoverflow.com/a/14587449>

4.1.6 PRIMERJAVA ALGORITMOV

Kriterij	sirino	globino	omejitvijo globine	iterativno poglabljanje	generirani vozlisce
----------	--------	---------	--------------------	-------------------------	---------------------

4.2 INFORMIRANI PREISKOVALNI ALGORITMI

4.2.1 HEVRISTICNO PREISKOVANJE

ideja: preiskovanje usmerjamo z dodatnim znanjem (ocenitven funkcija za obetavnost vozlisca)

hevrstica je ocenitvena funkcija za obetavnost vozlisca

- **optimisticna/dopustna:** $h(n) \leq h^*(n)$ (h^* je optimalna ocena)
- **optimalna:** $h(n) = h^*(n)$
- **pesimisticna:** $h(n) \geq h^*(n)$

4.2.2 POZRESNO PREISKOVANJE/ GREEDY BEST-FIRST SEARCH

$h(n)$ hevrsticna ocena

vrednotenje vozlisca $f(n) = h(n)$ hevrsticna ocena ... npr manhattan distance (zracna razdalja)

- **popolnost** (ali najde vedno resitev): Ne
- **optimalnost:** Ne
- **casovna zahtevnost** $O(b^m)$, kjer je m največja globina drevesa

4.2.3 A*

A* is informed version of **dijkstra** (uses heuristics and pq)

Vozlisca vrednotimo: $f(n) = g(n) + h(n)$

$g(n)$ cena poti do n (znano),

$h(n)$ cena od n do najblizjega cilja (ocena)

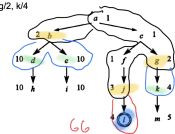
prioritetna vrsta (max glede na $f(n)$) Basically dijkstra + $h(n)$ (A* is basically an informed variation of Dijkstra.)

- **popolnost:** Da (ce ustreza pogoju dopustnosti)
- **optimalnost:** Da (ce ustreza pogoju dopustnosti)
- **casovna zahtevnost** $O(b^m)$, kjer je m največja globina drevesa

4.2.4 IDA* (ITERATIVE DEEPENING A*)

DFS with heuristics and iterative bound (value)

- primer:
 - podane so vrednost $f(n)$ ($= g(n) + h(n)$) vozlišč
 - simuliraj preiskovanje z IDA*
- generirana vozlišča
 - 1. iteracija, meja=1: a/1, b/2, c/1, f/1, j/3, g/2
 - 2. iteracija, meja=2: a/1, b/2, d/10, e/10, c/1, f/1, j/3, g/2, k/4
 - 3. iteracija, meja=3: a/1, b/2, d/10, e/10, c/1, f/1, j/3, i/4, g/2, k/4
 - 4. iteracija, meja=4: a/1, b/2, d/10, e/10, c/1, f/1, j/3, i/4



Učinkovitost

- neucinkovit ce vozlisca raznolika $f(n)$
- prednost: ne hrani vec vseh vozlisce kot A*
- optimalen: ce razvija v prioritetenem vrsntem redu, $h(n)$ mora biti **monotona|konsistentna** ($h(n)$ skos pada) (posledicno tudi dopustna)

$$h(n) \leq c(n, n') + h(n')$$

(h naslednjega vozlisca manjsi ker je blizji cilja)

- monotona \rightarrow dopustna (proti primer $h(n) = 0$)

4.2.5 KAKOVOST HEVRISTICNIH FUNKCIJ

7	2	4
5		6
8	3	1

Primer igra 8 ploscic

$-h_1$: stevilo ploscic ki niso na pravem mestu (8)

$-h_2$: vsota manhattanskih razdalj ploscic do pravega mesta ($3+1+2+2+3+3+2=18$)

Kakovost h ocenimo z:

Stevilo generiranih vozlisce

- z efektivnim faktorjem vejanja (koliko vozlisce N je algoritem general da je na globini d nasel resitev)

število generiranih vozlišč		efektivni faktor vejanja		
		IDS	A*(h1)	A*(h2)
2	10	6	3	2,45
4	112	13	12	2,87
6	680	20	18	2,73
8	6384	39	25	2,80
10	47127	93	39	2,79
12	3644035	227	73	2,78
14	?	539	113	?
16	?	1301	211	?
18	?	3056	363	?
20	?	7276	676	?
22	?	18094	1219	?
24	?	39135	1641	?

Vidimo $h_2(n) \geq h_1(n) \forall n$ pravimo h_2 **dominira** h_1

4.3 LOKALNO PREISKOVALNI ALGORITMI

4.3.1 PLEZANJE NA HRIB

Ne pomnemo poti do cilja, ampak samo trenutno stanje

Koristni v primerih:

- ce nas zanima samo kakovost resitve (in ne pot do cilja)
- reševanje optimizacijskih problemov (kjer je podana **kriterijska funkcija** za oceno kakovosti resitve)

Prednosti:

- majhna poraba prostora

Primer 4 kraljice na sahovnici - kriterijska funkcija: maksimiziramo - (minus) stevilo kraljic, ki se medsebojno napadajo
Tezave:

- lokalni maksimumi
- "rame, plaote" (kriterijska funkcija konstantna vrednost)
- grebeni (za plezanje navzgor je potreben sestop po pobocju grebena)
- Resevanje iz lokalnih maksimumov:
- **koraki vstran:** ce ima naslednje stanje isto vrednost kriterijske funkcije, dovolimo premik v to stanje
- **stochasticno** plezanje na hrib: iz mnozice boljsih stanj, verjetnostno izberemo naslednje stanje (pri cemer upostevamo da imajo boljsa stanja vecjo verjetnost izbora)
- **nakljucni ponovni zagon:** veckrat pozeni plezanje na hrib iz nakljucnih stanj dokler ne najdes resitve

4.3.2 SIMULIRANO OHLAJANJE

algoritem ki izvira iz metalurgije (ko je jeklo tekoce, so molekule v njem bolj gibljive; ko se ohlaja se strjuje in molekule se umirjajo) Analogija:

- generiramo nakljucne sosedne trenutnega stanja
- ce najdemo **boljše stanje ga izberemo**
- ce najdemo **slabše stanje, ga izberemo z določeno verjetnostjo**
- verjetnost izbire neoptimalnega stanja s casom pada (nizanje temperature)

4.3.3 LOKALNO ISKANJE V SNOPU

Algoritem:

- v spominu hrani k aktualnih stanj namesto enega
- izberi k optimalnih stanj od sosedov aktualnih stanj
- ponavljaj do ustavitnega pogoja

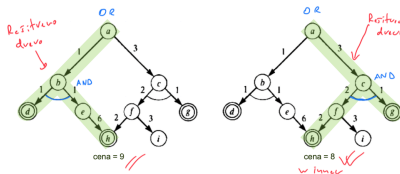
4.4 PREISKOVANJE GRAFOV AND/OR, NEDETERMINISTICNO OKOLJE

Pomagajo reševati probleme z **dekompozicijo na manjše probleme** Uporabnost:

- princip deli in vladaj
- iskanje v nedeterministicnih okoljih
- igre med dvema nasprotnikoma s popolno informacijo (sah, dama)
- ekspertno reševanje problem

Primer graf dekompozicija v dva manjša problema skozi g in f

Resitveno drevo je resitev AND/OR grafov



4.4.1 AO*

Vozlisce **OR**: Razvijamo najbolj obetavno poddrevo, dokler njegova cena ne preseže alternativnega poddrevesa Konec (AND vozlisca imajo v vsaki veji resitev, OR pa vsaj v eni)

- posplositev A* na grafe AND/OR
- **popoln in optimalen** $\Leftrightarrow h(n)$ ne precenjuje dejanske cene do cilja

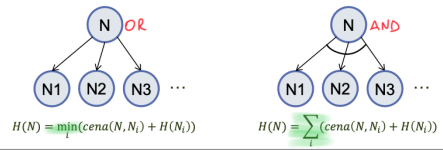
Vsako vozlisce N ima:

- lokalno (dinamicno) **hevrsticno oceno** $H(N)$
- lokalno (dinamico) vrednost **kriterijske funkcije** $F(N)$ $G(N)$ cena od predhodnika do trenutnega vozlisca

$$F(N_i) = G(N_i) + H(N_i) = \text{cena}(N_{i-1}, N_i) + H(N_i)$$

Dinamicna hevrsticna ocena $H(N)$ je **odvisna od tipa vozlisca**:

- za **liste**:
- $H(N) = h(n)$
- $F(N) = G(N) + H(N) = \text{cena}(\text{stars}, N) + h(n)$
- za **notranja vozlišca**:



4.4.2 PREISKOVANJE V NEDETERMINISTICNEM OKOLJU:

Nedeterministicnacija - ista akcija lahko obrodi razlicna ciljna stanja

Do resitve ni vec **poti** temvec **drevesa** (uporbljamo AND/OR grafe) Vozsilca **OR mozne akcije**, vozlisca **AND vejanja v mozna stanja**, ki so rezultat nedeterministicnih akcij

4.5 PREISKOVANJE BREZ INFORMACIJ O STANJU

Okolja smo razdelili na **transparent** (agent lahko zazna popolno informacija) in **netransparentna** (brez informacije o stanju) Kej ce imamo opravka z netraspranetim okoljem?

- izvajamo preiskovanje prostora **verjetnih** stanj in ne prostora **dejanskih** stanj
- izvajamo s postokopom omejevanja moznostmi kandidatnih stanj

4.6 IGRANJE IGER

4.6.1 PREDSTAVITEV PROBLEMA

4.6.2 ALGORITEM MINIMAX

- m globina - b

4.6.3 REZANJE ALFA-BETA

5 PLANIRANJE

plan zaporedje akcij, ki pripelje od zacetnega do koncnega stanja

5.1 STRIPS

Agentu opisemo svet in postavimo fizikalne omejitve.

Ne zagotavljamo optimalne resitve, obravnavamo le en cilj naenkrat (ko ga dosežemo, se lahko ostali izgubijo) = Sussmanova anomalija **Akcija** move(X, From, To)

- **pogoj:** con=[clr(X), on(X,F), clr(T)] \rightarrow pogoji za izvajanje akcije,
- **poz. ucinki:** adds=[on(X, T), clr(F)] \rightarrow nova stanja,
- **neg. ucinki:** dels=[on(X, F), clr(T)] \rightarrow izbrisana stanja,
- **omejitve:** constr=[$F \neq T$, $X \neq F$, $X \neq T$, block(X)] \rightarrow omejitve akcij (fizikalne omejitve),

Algoritem:

1. Izberi se neresen cilj iz mnozice CILJEV
2. Izberi akcijo, ki izbrani cilj doda v stanje
3. Omogoci izbrano akcijo (izpolni pogoje)
4. Izvedi akcijo (ki izpolni največ pogojev)
5. Ce obstajajo nereseni cilji \Rightarrow 1.

Primer dfs, zlaganje kock

DFS:

start $s = [on(a,1), on(b,2), on(c,3), c(c), c(2), c(2), c(4)]$
 child $g = [on(a,b), on(b,c)]$ (child se mora učitati v s)

1. child $on(a,b)$
 $move(a, F, b)$
 $cond = [c(a), c(b), on(a,1)]$

1.1. child $c(a)$
 $move(X, a, T)$
 $cond = [c(2), c(c), on(c,3)]$

Poslednja stanja (po koraku)
 $s = [on(a,1), c(a), on(b,2), c(c), c(b), c(4), on(c,2), c(a)]$

1. child $on(a,b)$
 $move(a, F, b)$
 $cond = [c(a), c(b), on(a,1)]$

Poslednja stanja (po koraku)
 $s = [on(b,3), c(c), c(4), on(b,1), on(c,2), c(a), on(c,1)]$

2. child $on(b,c)$
 $move(b, F, c)$
 $cond = [c(b), c(a), on(b,3)]$

2.2. $c(b)$
 $move(X, b, T)$
 $cond = [c(a), c(a), on(a,1)]$

novi stanje: $s =$

You got the idea

5.2 PLANIRANJE Z REGRESIRANJEM CILJEV

Resitev za sussmanovo anomalijo

5.3 RAZPOREJANJE OPRAVIL

6 SKLEPANJE

6.1 BAYESOVSKE MREZE