

Učenje v stavčni logiki

Ljupčo Todorovski

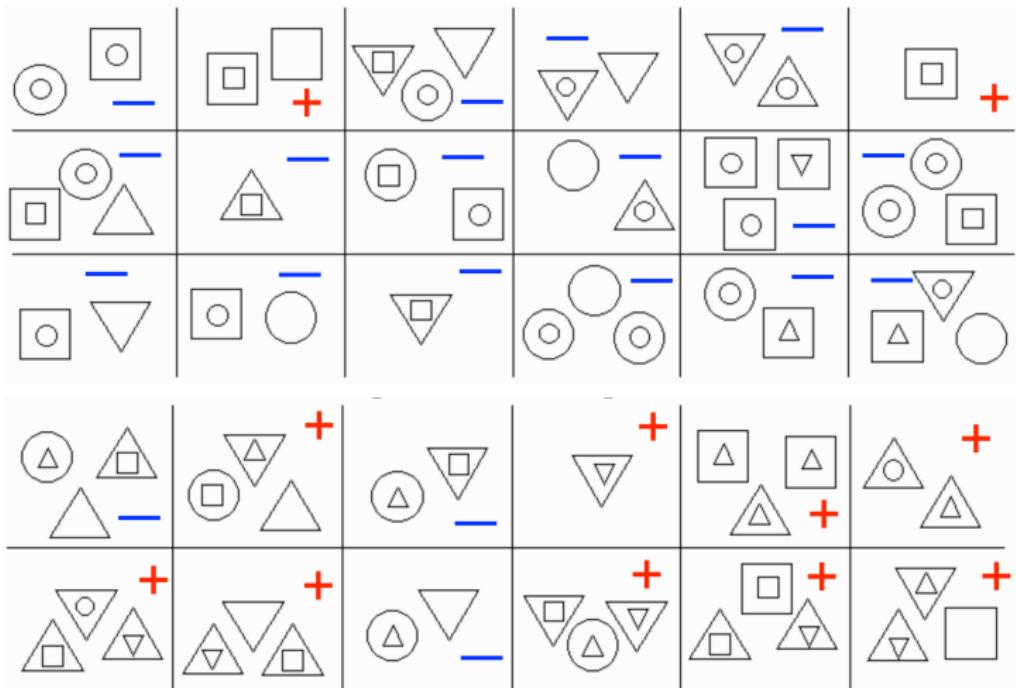
Univerza v Ljubljani, Fakulteta za matematiko in fiziko
Institut Jožef Stefan, Odsek za tehnologije znanja (E8)

April 2023

Pregled predavanja

- 1 Diagrami BONGARD
- 2 Neposredno učenje iz predikatov
 - Naštevanje možnih poizvedb
 - Izbera optimalne poizvedbe
- 3 Propozicionalizacija in agregati
 - Poizvedbe kot Boolovi atributi
 - Poizvedbe kot agregati

Pozitivni in negativni primeri



Štirje predikati

Trije enomestni predikati za oblike

- krog/1
- kvadrat/1
- trikotnik/1

Dvomesten predikat

vsebuje/2

Predikatni opis primerov

Prvi negativni primer	$\text{krog}(k1), \text{krog}(k2), \text{krog}(k3),$ $\text{kvadrat}(v1),$ $\text{vsebuje}(k1, k2), \text{vsebuje}(v1, k3)$
Prvi pozitivni primer	$\text{kvadrat}(v1), \text{kvadrat}(v2), \text{kvadrat}(v3),$ $\text{vsebuje}(v1, v2)$
Tretji pozitivni primer	$\text{krog}(k1),$ $\text{kvadrat}(v1),$ $\text{trikotnik}(t1), \text{trikotnik}(t2), \text{trikotnik}(t3),$ $\text{vsebuje}(k1, v1), \text{vsebuje}(t1, t2)$

Razlika od običajnega strojnega učenja

Primeri

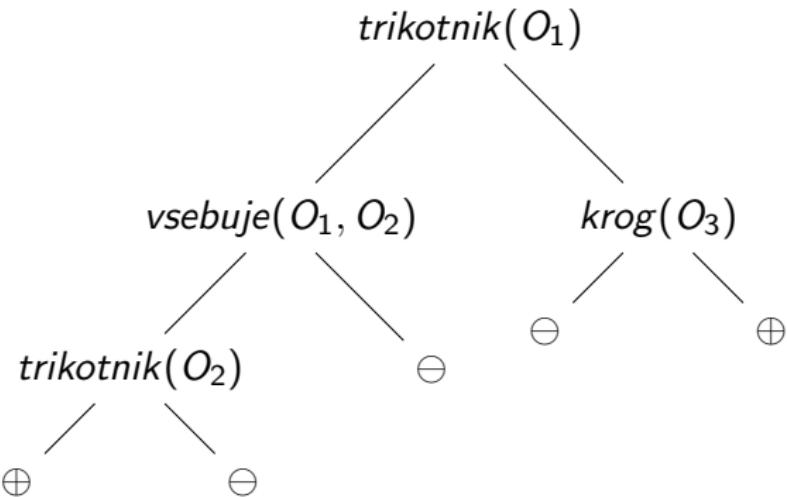
- So množice namesto vektorjev oz. p -teric
- Vsebujejo strukturirane izraze (predikate)

Model za razvrščanje: Pozitivni primer

- Ima vsaj en trikotnik vsebovan v drugem trikotniku
- Sicer, če nima trikotnika, ne sme imeti niti kroga

Kako formalno zapišemo tak model?

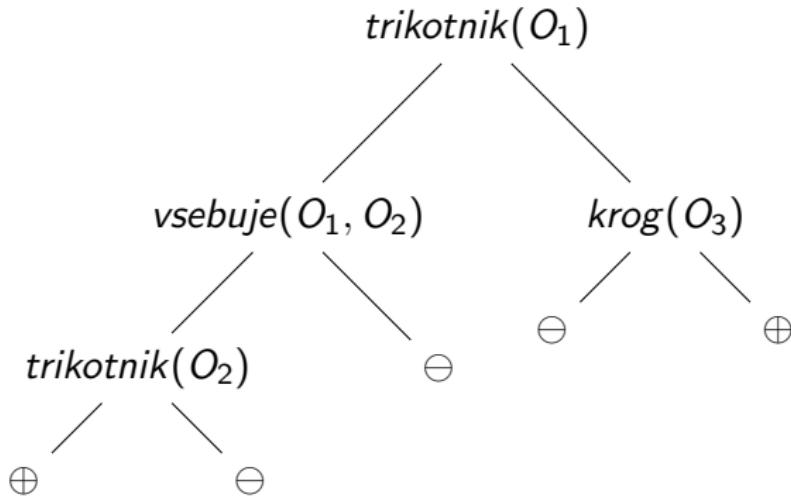
Relacijsko, lahko tudi predikatno drevo



Poglejmo še enkrat semantiko tega drevesa.

→ če je stavek nerezničen poravdba uspe in se premikamo proti levu

Relacijska odločitvena drevesa: vozlišča



Pomen vozlišč

- Notranja vozlišča: poizvedbe, npr. $\leftarrow \text{trikotnik}(O_1)$
- Končna vozlišča: napovedi

Relacijska odločitvena drevesa: veje

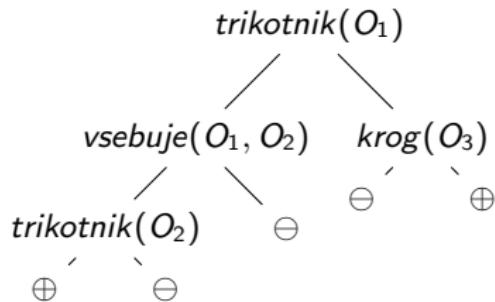
Leva veja

- Poizvedba v vozlišču uspešna
- Spremenljivke dobijo vrednosti iz rešitev poizvedbe
- Vrednosti spremenljivk **veljavne le v levi veji**

Desna veja

- Poizvedba v vozlišču ni uspešna
- Vrednosti spremenljivk iz poizvedbe neznane
- Zato tudi spremenljivke nimajo vrednosti v desni veji

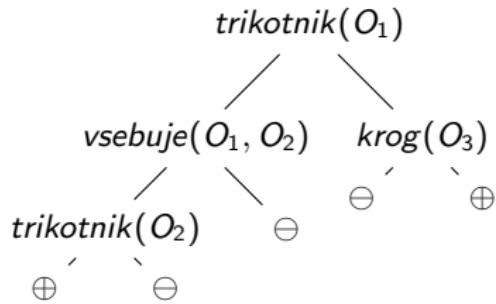
Relacijska odločitvena drevesa: napovedi



Končna vozlišča od leve proti desni

- ① $\exists O_1, O_2 : \text{trikotnik}(O_1) \wedge \text{vsebuje}(O_1, O_2) \wedge \text{trikotnik}(O_2) \Rightarrow \oplus$
- ② $\exists O_1, O_2 : \text{trikotnik}(O_1) \wedge \text{vsebuje}(O_1, O_2) \wedge \neg \text{trikotnik}(O_2) \Rightarrow \ominus$
- ③ $\exists O_1 : \text{trikotnik}(O_1) \wedge \nexists O_2 : \text{vsebuje}(O_1, O_2) \Rightarrow \ominus$
- ④ $\nexists O_1 : \text{trikotnik}(O_1) \wedge \exists O_3 : \text{krog}(O_3) \Rightarrow \ominus$
- ⑤ $\nexists O_1 : \text{trikotnik}(O_1) \wedge \nexists O_3 : \text{krog}(O_3) \Rightarrow \oplus$

Relacijska odločitvena drevesa: IF-THEN-ELSE



IF $\text{trikotnik}(O_1) \wedge vsebuje(O_1, O_2) \wedge \text{trikotnik}(O_2)$ THEN \oplus
 ELIF $\text{trikotnik}(O_1) \wedge vsebuje(O_1, O_2)$ THEN \ominus
 ELIF $\text{trikotnik}(O_1)$ THEN \ominus
 ELIF $krog(O_3)$ THEN \ominus
 ELSE \oplus

Dva pristopa k učenju v stavčni logiki

Neposredno učenje

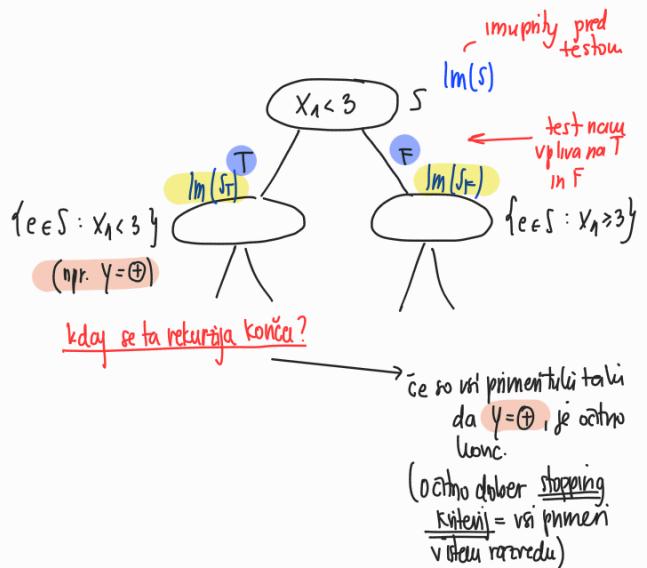
- Učenje relacijskih dreves iz predikatov in poizvedb
- Naštevanje vseh možnih poizvedb
- Izbera optimalnih poizvedb za napovedovanje

Propozicionalizacija in agregati

- Pretvorba kompleksnih podatkov v vektorje fiksne dolžine
- Pretvorba poizvedb v atributе
- Dva tipa atributov: Boolovi pogoji in numerični agregati

TDIDT = top - Down Induction
of Decision trees

→ vodíčou priedmo množico prímerov
(koreňovemu dreveru celotno množico S)



Variancia luka pravomo NECERTOČA / IMPURITY:
 $Im(S)$

Enkrat kez zbereme test, znamo impurity zmerte

Na nek nahneželku zmanjšati varianciu (a upokojimo varianciu ciljne spremenljivke)
(varianca lahko merimo za numerične spremeni.)

$$IR(t) = Im(S) - \left(\frac{|S_T|}{|S|} Im(S_T) + \frac{|S_F|}{|S|} Im(S_F) \right)$$

Impurity reduction: $IR(t) = Im(S) - \left(\frac{|S_T|}{|S|} Im(S_T) + \frac{|S_F|}{|S|} Im(S_F) \right)$

znamena max t

naš optimizacijski problem

to je zberemo t (test) s katerim bomo množico S razbil na S_T in S_F

(tak optimizacijski problem rešujemo na tříkrokou)

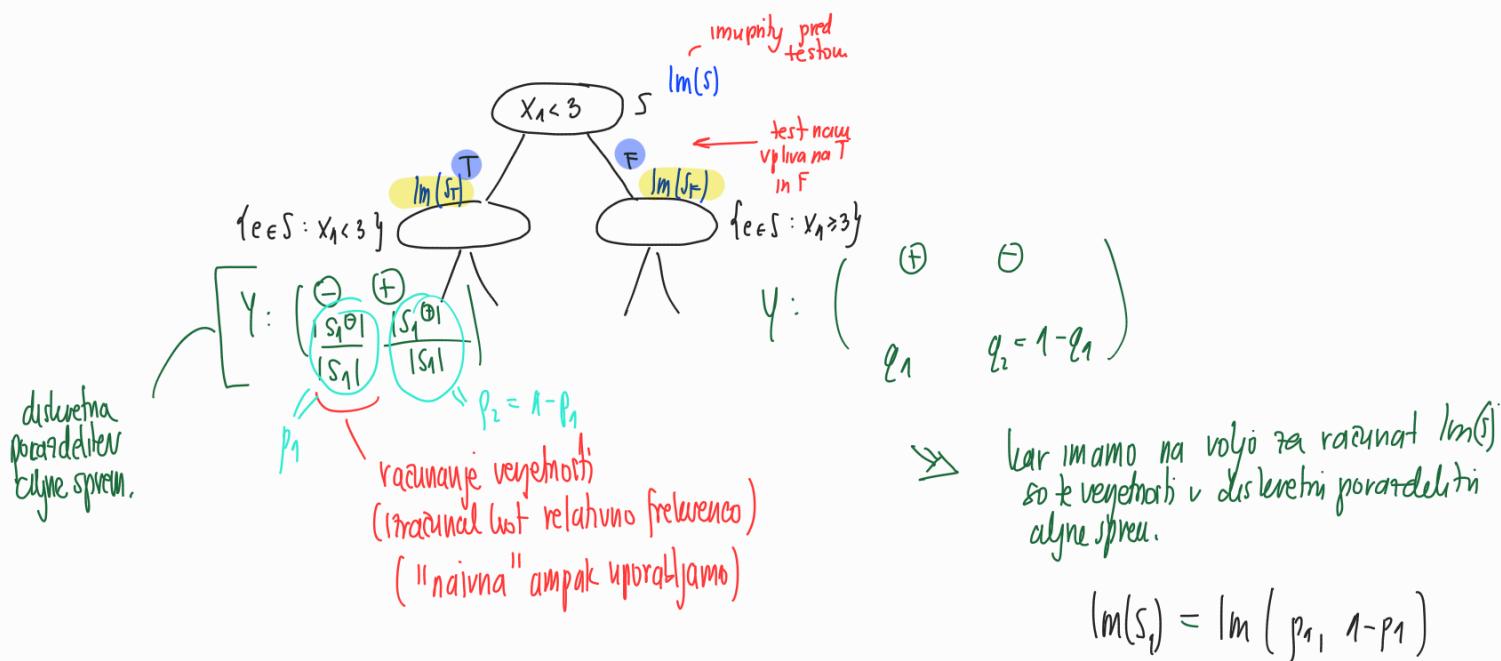
TDIDT ... požrešen algoritmu → dří problem pogleda samu cestu krokou napřej pa řeče da je fáil dobr (takže fáil naročuje všechno)

naredí neoptimalnou cestu

\Rightarrow zakaj se pol vseh zavetkov k tem pozremim algoritmom?

\hookrightarrow ko ne moremo nulti redi v normalnem času (ekspponentnemu)
se kot kompromis zatečemo k tem algoritmom.

($\text{Var} = 0$ super,
 $\text{Var} = \text{velika...}$ to hčemo izmanjšati)



pogledimo ekstreme te funkcije (Im). Kdaj dosežemo max nečistost (min entropijo)?

$\boxed{\min Im}$, kadar $p_1=1$ ali pa $1-p_1=1$ (torej ena od vrednosti je 1)

Kdaj pa $\boxed{\max Im}$: $p_1=1-p_1$

A je pomemben vrsti red argumentov pri Im funkciji? Nebi smel biti (ker so arbitrarни)

\Rightarrow torej ena od strani ki jo lahko uporabimo kot Impurity function je Entropija

$$Im(S) = - \sum_i p_i \cdot \log_2 p_i$$

merska entota entropije je $[bit]$

(kuščka kolikana info je helje)

zakaj ne maramo? log je treba računat log

\Rightarrow druga opäťa : Gini index

$$Im(S) = 1 - \sum_i p_i^2$$

Pokritost in pokritje

Pokritost $\text{covers}(p, e)$

- p je poizvedba, e je Herbrandova interpretacija
- Pokritost je enaka uspešnosti poizvedbe

Pokritje $\text{coverage}(p)$

$$\text{coverage}(p) = \{e : \text{covers}(p, e)\}$$

Množica vseh Herbrandnovih interpretacij, za katere je poizvedba uspešna.

Zakaj je to pomembno?

Ker nam lahko pomaga strukturirati prostor poizvedb.

Primer pokritja za poizvedbo $p = \text{trikotnik}(O_1)$



Pokritost primera e_{20}

- Herbrandova interpretacija za primer: $krog(k1)$, $kvadrat(v1)$, $\text{trikotnik}(t1)$, $\text{trikotnik}(t2)$, $\text{trikotnik}(t3)$, $vsebuje(k1, v1)$, $vsebuje(t1, t2)$
- Uporabimo zamenjavo $O_1 = t1$: dobimo poizvedbo $\leftarrow \text{trikotnik}(t1)$
- Ekvivalentni logični izraz je $\neg \text{trikotnik}(t1)$: očitno protislovje
- Torej je poizvedba uspešna, primer pa **pokrit**, velja $\text{covers}(p, e_{20})$
- Rezultat poizvedbe je množica možnih zamenjav: $O_1 \in \{t1, t2, t3\}$

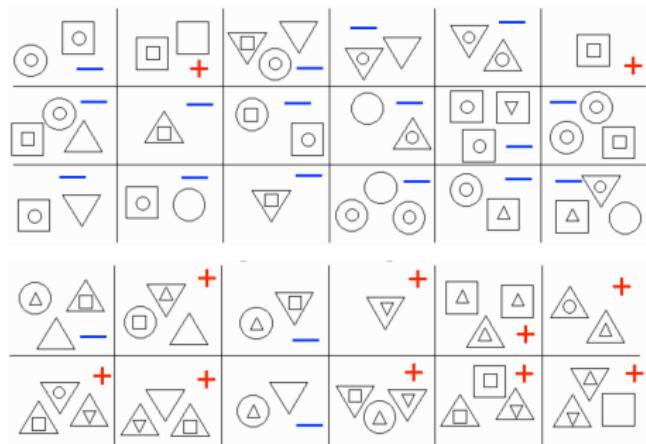
Primer pokritja za poizvedbo $p = \text{trikotnik}(O_1)$



Pokritost primera e_{14}

- Herbrandova interpretacija za primer: $\text{kvadrat}(v1)$, $\text{krog}(k1)$, $\text{krog}(k2)$, $\text{vsebuje}(v1, k1)$
- Uporabimo zamenjavo $O_1 = v1$: dobimo poizvedbo $\leftarrow \text{trikotnik}(v1)$
- Ekvivalentni logični izraz je $\neg \text{trikotnik}(v1)$: ni protislovja
- Ni nobenega protislovja, poizvedba je neuspešna, primer **ni pokrit**
- Torej ne velja $\text{covers}(p, e_{14})$
- Rezultat poizvedbe je množica možnih zamenjav: $O_1 \in \emptyset$

Primer pokritja za poizvedbo $p = \text{trikotnik}(O_1)$



$$\text{coverage}(p) = \{e_3, e_4, e_5, e_7, e_8, e_{10}, e_{11}, e_{13}, e_{15}, e_{17}, e_{18}, e_{19}, e_{20}, \dots, e_{30}\}$$

Relacija posploševanja

p_1 je bolj splošna od poizvedbe p_2 , $p_1 \preceq p_2$:

$$\text{coverage}(p_2) \subseteq \text{coverage}(p_1)$$

Rečemo tudi, da je p_1 manj specifična od p_2 .

ko naredimo poravdbo p_2 manj
splošnu od p_1 smo
zmanjšal coverage.

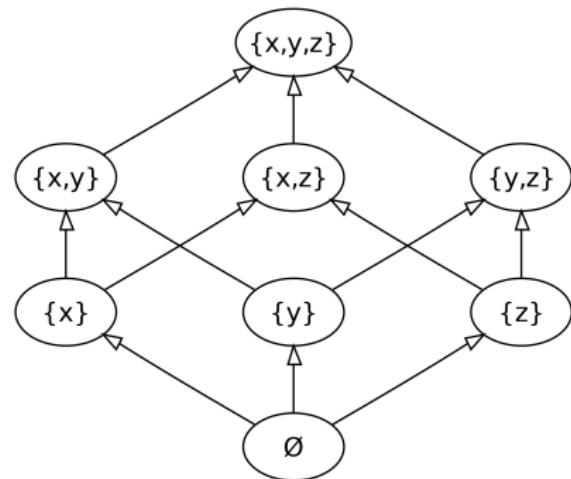
najbolj splošna poizvedba: TRUE (poizvedba, ki ima prazno telo)
najmanj -||- : FALSE

Robni poizvedbi

- $\leftarrow \top$: pokriva vse Herbrandove interpretacije
- $\leftarrow \perp$: ne pokriva nobene interpretacije, $\text{coverage}(\perp) = \emptyset$

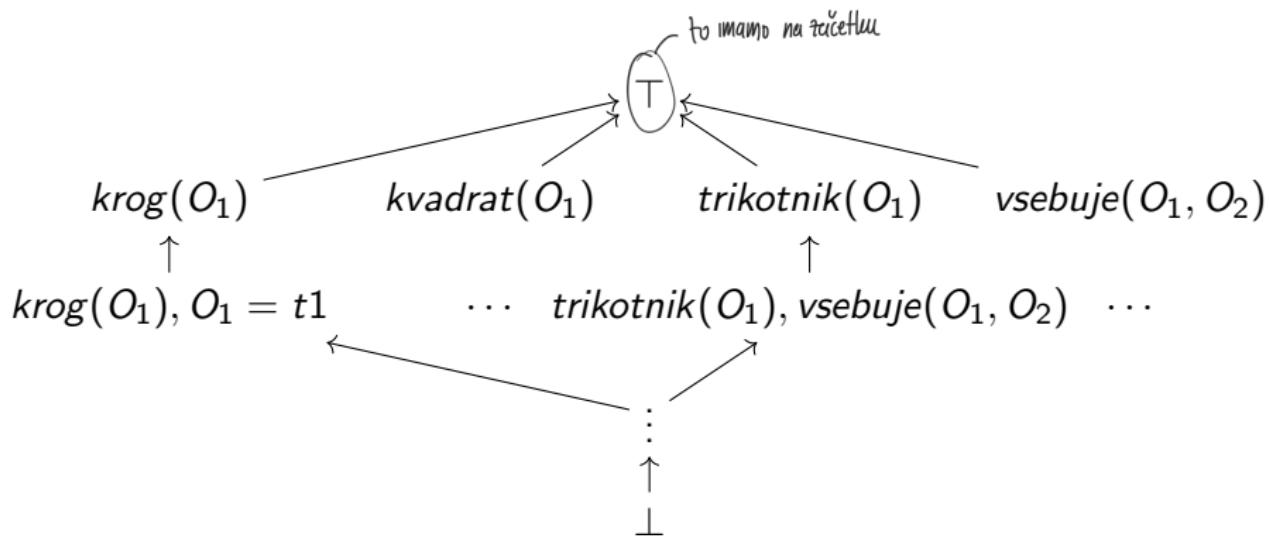
Hassejev diagram

Hassejev diagram je podmnožic množice
(+ operatorjem inširanja enega elementa
pridelavo do vseh možnosti)



Hassejev diagram za Bongard

Relacija posploševanja delno uredi prostor poizvedb



Algoritem za naštevanje: operator izostritve

pove naslednji vri q so v relaciji s p (kateri q so nasledniki od p)

Operator izostritve ρ

$$\rho(p) = \{q : p \preceq q\}$$

Od splošnega k specifičnemu (lahko tudi od specifičnega k splošnemu)

Lastnosti operatorjev izostritve

če bi brisal po dva elementa na enkrat ni idealno! ker idealen operator izostritve je le tak ki dela popoloma (tako ne preklaplja nobene poizvedbe)

- **Idealnost:** nasledniki so najbolj splošne poizvedbe
- **Popolnost:** iz najbolj splošne poizvedbe z zaporedjem izostritev pridemo do katerokoli druge poizvedbe *vsaj na en način*
- **Optimalnost:** isto kot zgoraj, a na točno en način

Operator izostritve: θ -subsumpcija, θ -subsumption

p_1 θ -subsuma p_2 , $p_1 \preceq_{\theta} p_2$:

$$\exists \theta : L(p_1 | \theta) \subseteq L(p_2)$$

\checkmark \exists zamenjava (θ)
literali v p_2
literali, ki se pojavijo pod to zamenjavo θ v p_1

- θ je zamenjava
- $p_1 | \theta$ je stavek p_1 po zamenjavi θ
- $L(p)$ je množica literalov iz disjunktne oblike p

Ustrezen operator izostritve ρ_{θ}

$$\rho_{\theta}(p) = \{q : p \preceq_{\theta} q\}$$

Lastnosti operatorja θ -subsumpcija

kaj bomo počeli? Dodajil bomo nove literale, poleg tega bomo pa še zamenjave spremenljivke delal
 poljubni konjunkčni dodamo literal \Rightarrow s tem zaznamo povez (ker zdi zahtevalno več)

Enostaven za implementacijo

- Stavku dodamo poljuben literal
- Zamenjavo lahko izvedemo z literalom oblike $X = Y$

dvo mesni predikat

Težave s semantiko

- $p \preceq_{\theta} q \Rightarrow p \preceq q$, ampak $p \preceq q \not\Rightarrow p \preceq_{\theta} q$
- Zato je možno $p \preceq_{\theta} q$ in $q \preceq_{\theta} p$ tudi za $p \neq q$
- Takrat rečemo, da sta p in q sintaktični varianti istega stavka
- Posledica sintaktičnih variant so težave z optimalnostjo izostritve

TO KAR BONA NASEVEDNIH SLAJDIH SHO ZGORI DEJAL (za Impurity)

Funkcija nečistoče $Impurity(S)$

Dohi: da kajko podredba razdeli na mn. podnih primerov in mn. nepodnih primerov zato je kajko računati Impurity.

Funkcija nečistoče meri varianco vrednosti ciljne spr. Y v množici S .

Regresija, $D_Y \subseteq \mathbb{R}$

$$Impurity(S) = \frac{1}{|S|} \sum_{(x,y) \in S} (y - \bar{y})^2$$

Klasifikacija, $D_Y = \{v_1, v_2, \dots, v_c\}$

$$Impurity(S) = \phi(p_1, p_2, \dots, p_c)$$

Verjetnosti $p_i = P(Y = v_i | S)$

Zaželene lastnosti funkcije nečistoče $\phi(p_1, p_2, \dots, p_c)$

- Doseže maksimalno vrednost pri enakomerni porazdelitvi $\forall i : p_i = 1/c$
- Doseže minimalno vrednost pri porazdelitvah, kjer $\exists ! i : p_i = 1$
- Simetrična: neobčutljiva na vrstni red parametrov
- Konkavna, zvezna in zvezno odvedljiva

vse večjeneake

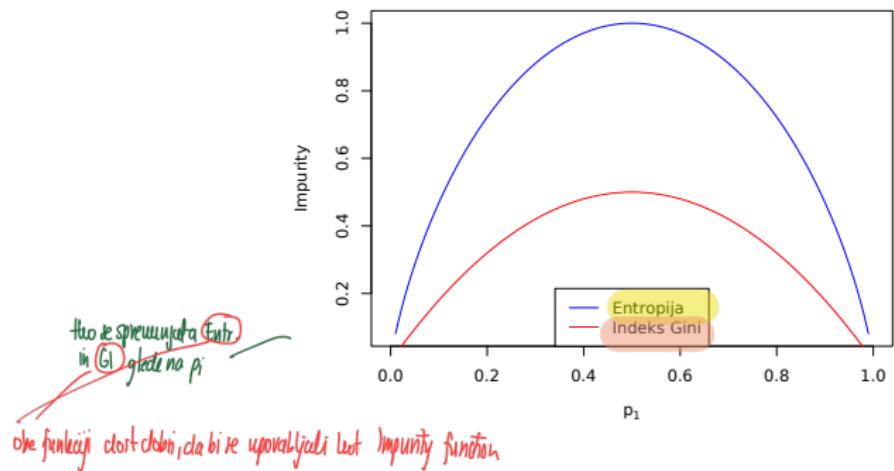
✓

eden od $p=1$, ostali = 0

ker ni pravega vrstnega reda

Dve pogosto uporabljeni funkciji

- ① Entropija $\phi(p_1, p_2, \dots, p_c) = -\sum_{i=1}^c p_i \log_2 p_i$
- ② Indeks Gini $\phi(p_1, p_2, \dots, p_c) = 1 - \sum_{i=1}^c p_i^2$



Izbira optimalne poizvedbe $Split = SelectOptimal(S)$

Funkcija zmanjšanja nečistoče IR , Impurity Reduction

$$IR(Split, S) = Impurity(S) - \left(\frac{|S_c|}{|S|} Impurity(S_1) + \frac{|S_{nc}|}{|S|} Impurity(S_2) \right)$$

covered
 non-covered

- $S_c \subseteq S$ je množica primerov, ki jih poizvedba **pokriva**
- $S_{nc} \subseteq S$ je množica primerov, ki jih poizvedba **ne pokriva**

Test izberemo z optimizacijo ciljne funkcije IR

$$\max_{Split} IR(Split, S) = \min_{Split} \frac{|S_1|}{|S|} Impurity(S_1) + \frac{|S_2|}{|S|} Impurity(S_2)$$

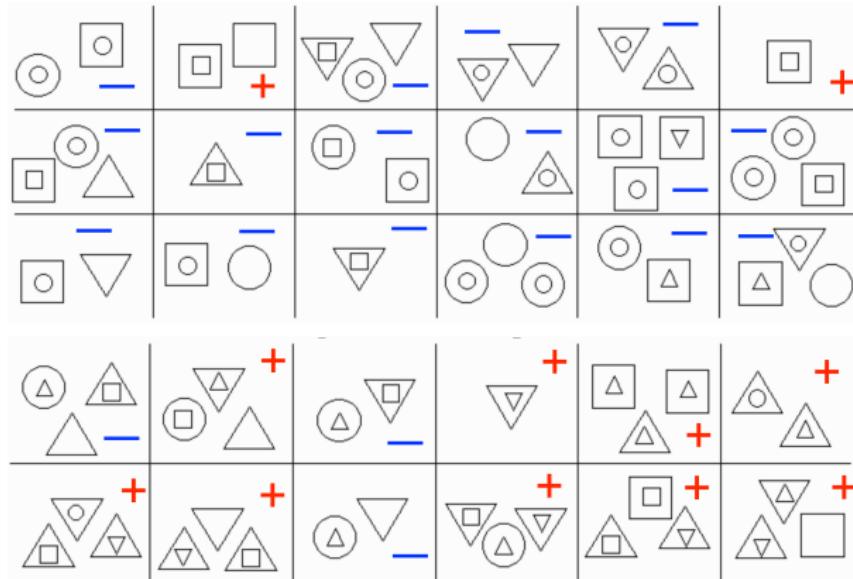
covered
 non-covered

(ponovitev prof prih in z)

Možne teste $Split$ naštevamo z operatorjem izostritve ρ .

BONGARD: 30 primerov za dvojiško razvrščanje

Primere oštrevljčimo od leve proti desni, nato navzdol



BONGARD: Herbrandove interpretacije primerov

e_1, \ominus	krog(k1), krog(k2), krog(k3), kvadrat(v1), vsebuje(k1, k2), vsebuje(v1, k3)
e_2, \oplus	kvadrat(v1), kvadrat(v2), kvadrat(v3), vsebuje(v1, v2)
e_{20}, \oplus	krog(k1), kvadrat(v1), trikotnik(t1), trikotnik(t2), trikotnik(t3), vsebuje(k1, v1), vsebuje(t1, t2)

BONGARD: Nečistost učne množice in testi

$Impurity(S), S = \{e_1, e_2, \dots, e_{30}\}$

11 poz. primerov 19 neg. primerov

- $p(\oplus) = 11/30, p(\ominus) = 19/30$
- $Gini(S) = 1 - ((11/30)^2 + (19/30)^2) \doteq 0.46$

Možne poizvedbe v korenskem vozlišču, izostritve T (operator rostnike pravidači literal)

- 1 $\leftarrow krog(O_1)$
 - 2 $\leftarrow kvadrat(O_1)$
 - 3 $\leftarrow trikotnik(O_1)$
 - 4 $\leftarrow vsebuje(O_1, O_2)$
- ali moj diagramu vsebuje levo
- možni literali na 1. koraku

ali moj diagramu vsebuje neki kar vsebuje neki drugega

BONGARD: Test $trikotnik(O_1)$

Pozor: $trikotnik(O_1) \equiv \exists O_1 : trikotnik(O_1)$... zde k vsemu datu výplní lze k delšímu mít dve rady.

$Impurity(S_c), |S_c| = 23$, vzorci s trikotníkem

- $p(\oplus) = 9/23$, $p(\ominus) = 14/23$
 - $Gini(S_c) = 1 - ((9/23)^2 + (14/23)^2) \doteq 0.48$
- 9 pozitivních 9 neg.

$Impurity(S_{nc}), |S_{nc}| = 7$, vzorci bez trikotníka

- $p(\oplus) = 2/7$, $p(\ominus) = 5/7$
- $Gini(S_{nc}) = 1 - ((2/7)^2 + (5/7)^2) \doteq 0.41$

$$IR(trikotnik(O_1)) \doteq 0.46 - \left(\frac{23}{30} 0.48 + \frac{7}{30} 0.41 \right) \doteq 0.004$$

BONGARD: Test krog(O_1)

ko za lung porazinimo dobimo boljši impurity reduction

Impurity(S_c), $|S_c| = 20$, Bongard vzorci s krogom

- $p(\oplus) = 4/20 = 1/5, p(\ominus) = 16/20 = 4/5$
- $Gini(S_c) = 1 - ((1/5)^2 + (4/5)^2) = 0.32$

Impurity(S_{nc}), $|S_{nc}| = 10$, Bongard vzorci brez kroga

- $p(\oplus) = 7/10, p(\ominus) = 3/10$
- $Gini(S_{nc}) = 1 - ((7/10)^2 + (3/10)^2) = 0.42$

$$IR(krog(O_1)) \doteq 0.46 - \left(\frac{20}{30} \cdot 0.32 + \frac{10}{30} \cdot 0.42 \right) \doteq 0.11$$

Kaj potrebujejo narediti tisto da nameri da sledijo samo en krok na naprej naredita 2 in pol triradenja K pa vidijo da trikotnik izmaga

Takoj naredi sledimo samo en krok? ker več zara name

Izbor nadaljnjih testov

V levih vejih

- Poizvedba $\leftarrow \text{trikotnik}(O_1)$ je uspela
- V vseh podrejenih vozliščih velja torej $\exists O_1 : \text{trikotnik}(O_1)$
- O_1 lahko zavzame eno od vrednosti, ki ustreza enemu izmed trikotnikov v vzorcu

V desnih vejih

- Poizvedba $\leftarrow \text{trikotnik}(O_1)$ ni uspela
- V vseh podrejenih vozliščih velja torej $\nexists O_1 : \text{trikotnik}(O_1)$
- O_1 nima smiselne vrednosti (O_1 ni havaljo)

operator izostitve se mora izvedeti kro delu izostitve za desno izraz, da se ne more zmanjšati na igrecu. O_1 [fam je ne bo]

BONGARD: Test $trikotnik(O_1), vsebuje(O_1, O_2)$

Impurity(S_c), $|S_c| = 18$, vzorci s trikotnikom z vsebino

- $p(\oplus) = 9/18, p(\ominus) = 9/18$
- $Gini(S_c) = 1 - ((9/18)^2 + (9/18)^2) = 0.5$

Impurity(S_{nc}), $|S_{nc}| = 5$, vzorci s praznim trikotnikom

- $p(\oplus) = 0/5, p(\ominus) = 5/5$
- $Gini(S_{nc}) = 1 - ((0/5)^2 + (5/5)^2) = 0$
- Čisto vozlišče, ki napoveduje \ominus

prišli smo do čstega vortlita
ker vsi primeni neg

$$IR(trikotnik(O_1), vsebuje(O_1, O_2)) \doteq 0.48 - \left(\frac{18}{23}0.5 + \frac{5}{23}0 \right) \doteq 0.089$$

BONGARD: $\text{trikotnik}(O_1)$, $\text{vsebuje}(O_1, O_2)$, $\text{trikotnik}(O_2)$

$\text{Impurity}(S_c), |S_c| = 9$, vzorci s trikotnikom, ki vsebuje trikotnik

- $p(\oplus) = 9/9, p(\ominus) = 0/9$
- $Gini(S_c) = 1 - ((9/9)^2 + (0/9)^2) = 0$
- Čisto vozlišče, ki napoveduje \oplus

$\text{Impurity}(S_{nc}), |S_{nc}| = 9$, vzorci s trikotniki, ki vsebujejo le kvadrate ali kroge

- $p(\oplus) = 0/9, p(\ominus) = 9/9$
- $Gini(S_{nc}) = 1 - ((0/9)^2 + (9/9)^2) = 0$
- Čisto vozlišče, ki napoveduje \ominus

$$IR(\text{trikotnik}(O_1), \text{vsebuje}(O_1, O_2), \text{trikotnik}(O_2)) = 0.5 - \left(\frac{9}{18}0 + \frac{9}{18}0 \right) = 0.5$$

BONGARD: Test $\neg trikotnik(O_1), krog(O_3)$

Pozor: $\neg trikotnik(O_1) \equiv \nexists O_1 : trikotnik(O_1)$

Impurity(S_c), $|S_c| = 5$, vzorci brez trikotnika in s krogom

- $p(\oplus) = 0/5, p(\ominus) = 5/5$
- $Gini(S_c) = 1 - ((0/5)^2 + (5/5)^2) = 0$
- Čisto vozlišče, ki napoveduje \ominus

Impurity(S_{nc}), $|S_{nc}| = 2$, vzorci brez trikotnika in brez kroga

- $p(\oplus) = 2/2, p(\ominus) = 0/2$
- $Gini(S_{nc}) = 1 - ((2/2)^2 + (0/2)^2) = 0$
- Čisto vozlišče, ki napoveduje \oplus

$$IR(\neg trikotnik(O_1), krog(O_3)) \doteq 0.41 - \left(\frac{5}{7}0 + \frac{2}{7}0 \right) = 0.41$$

Algoritem za učenje relacijskih dreves

Učenje relacijskih dreves iz množice Herbrandovih interpretacij S , pri začetnem klicu je $Q = \top$

```
function TDIRDT( $S, Q$ )
   $Q_b = SelectOptimal(\rho_\theta(Q), S)$ 
  if  $Stop(Q_b, S)$  then
    return  $leaf(S)$ 
  else
     $Split = Q_b - Q$ 
     $S_c = \{e \in S : covers(Q_b, e)\}$ 
     $S_{nc} = S \setminus S_c$ 
    return  $tree(Split, TDIRDT(S_c, Q_b), TDIRDT(S_{nc}, Q))$ 
```

Viri in implementacije

Viri

- Učbenik (De Raedt 2008): Logical and Relational Learning
- Doktorska disertacija (Blockeel 1998)
- Članek (Petković 2022): Re3Py, glej spodaj

Implementacije

- Veliko implementacij zahteva Prolog, npr. Alph
www.cs.ox.ac.uk/activities/programinduction/Aleph/aleph.html
- Primer izjeme: github.com/re3py/re3py

Diagrami Bongard zapisani malo drugače

definira čilno spremenljivko

Ciljni predikat *diagram/2*: $\text{diagram}(ID.d, Class)$

$\text{diagram}(e_1, neg), \text{diagram}(e_2, pos), \dots, \text{diagram}(e_{30}, pos)$

Določi razred $Class \in \{pos, neg\}$ diagrama (ciljnega objekta) $ID.d$.

Diagrami Bongard zapisani malo drugače



Predikat *element/3*: $\text{element}(ID.e, ID.d, \text{Oblika})$

identifikatorji elementov, ki jih pot uporabimo v predikatu vsebuje

$\text{element}(v_1, e_{14}, \text{kvadrat}), \text{element}(k_1, e_{14}, \text{krog}), \text{element}(k_2, e_{14}, \text{krog})$

Določi obliko *Oblika* elementa *ID.e* v diagramu *ID.d*.

Predikat *vsebuje/2*: $\text{vsebuje}(ID.e.1, ID.e.2)$

$vsebuje(v_1, k_1)$

Oznani dejstvo da element *ID.e.1* vsebuje element *ID.e.2*.

Predikatne relacije

Predikatna relacija $r_X(ID.\text{objekt}, \dots)$ poveže

- Nek ciljni objekt $ID.\text{objekt}$
- z drugimi objekti in lastnostmi teh objektov

Primer trimestne relacije $r_{vsebuje.\text{obliko}}(ID.d, ID.e, O)$

$$r_{vsebuje.\text{obliko}}(ID.d, ID.e, O) \leftarrow \text{diagram}(ID.d, C), \text{element}(ID.e, ID.d, O)$$

diagram d
vsebuje obliko
element e

- Zgornji stavek definira relacijo, ki poveže
- Diagram $ID.d$ z obliko O enega izmed elementov $ID.e$ diagrama

Predikatne relacije, poizvedbe in atributi

Stavčna definicija relacije $r_X(ID.\text{objekt}, \dots) \leftarrow p$

- Ima na desni strani poizvedbo p
- Poizvedba p je lahko uspešna ali ne

na desni mamo različne poizvedbe, te bi lahko nasteval in s tem bi najel vse možne spremembe.

Relacija $r_X(ID, \dots)$ je lahko Boolov atribut X ciljnih objektov

$$X(ID) = \begin{cases} \text{True} & ; \text{poizvedba } p \text{ je uspela za ciljni objekt } ID \\ \text{False} & ; \text{sicer} \end{cases}$$

Predikatna relacija $r_{vsebuje.obliko}(ID.d, ID.e, O)$ in atributi



Predikatna relacija in vrednost atributa

- $r_{vsebuje.obliko}(e_{14}, E, Oblika)$ velja, npr. $E = k_1$, $Oblika = krog$
- Torej atribut $X_{vsebuje.obliko}(e_{14}) = True$.

Alternativne uporabe

- $r_{vsebuje.obliko}(e_{14}, E, krog)$ velja, $X_{vsebuje.obliko.krog}(e_{14}) = True$
- $r_{vsebuje.obliko}(e_{14}, E, kvadrat)$ velja, $X_{vsebuje.obliko.kvadrat}(e_{14}) = True$
- $r_{vsebuje.obliko}(e_{14}, E, trikoktnik)$ ne velja, $X_{vsebuje.obliko.trik}(e_{14}) = False$

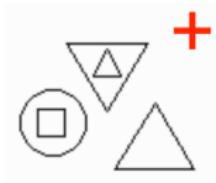
Tabela za običajno strojno učenje

Objekt	Diagram	$X_{V.O.krog}$	$X_{V.O.kvadrat}$	$X_{V.O.trik}$...
⋮	⋮	⋮	⋮	⋮	⋮
e_{14}		True	True	False	...
⋮	⋮	⋮	⋮	⋮	⋮
e_{20}		True	True	True	...
⋮	⋮	⋮	⋮	⋮	⋮

V.O okrajšava za vsebuje.obliko.

↑ ko imam takvo tabelo lahko uporabil
njen algortitem za strojno učenje

Predikatna relacija $r_{vsebovani.oblik}$



$r_{vsebovani.oblik}(ID.d, ID.e1, O1, ID.e2, O2) \leftarrow$
 $diagram(ID.d, C), element(ID.d, ID.e1, O1), element(ID.d, ID.e2, O2),$
 $vsebuje(ID.e1, ID.e2)$

$r_{vsebovani.oblik}(_)(e_{20}, ID.e1, trikotnik, ID.e2, trikotnik)$

- Ali je element diagrama trikotnik v trikotniku?
- $X_{V.O.trik.trik}(e_{20}) = True$

skratka na koncu
vse rata spremenljivka

Kako do celotnega nabora možnih atributov

lahko ga uporabljamo da prideemo do vseh možnih definicij spremenljivki

Uporabimo operator izostritve ρ, ρ_θ

- Tako kot smo ga uporabili za naštevanje testov v drevesih
- Od splošne, \top oz. $diagram(ID.d, C)$, k bolj specifičnim poizvedbam
- Ustavitevni kriterij: poizvedba je enaka False za vse ciljne objekte
- Dodatni kriterij: maksimalno število vključenih literalov

Vsako poizvedbo p pretvorimo v relacijo in nato v atribut

- $r_X(ID.d, \dots) \leftarrow p$
- Izračunamo vrednosti X na osnovi stavka za r_X

Rešitev poizvedbe pove več

 \exists

~~Štiri~~ rešitve poizvedbe $r_{vsebuje.obliko}(e_{14}, E, O)$

- $E, O = v1, kvadrat$
- $E, O = k1, krog$
- $E, O = k2, krog$

(mi smo se spričeval ali vsebuje kvadrat ali krog,
torej se počut po številu kvadratov in številih
krogov)
 \models Bodene spremi lahko prečitamo v numerične in se gremo
razbirna zletja.

Lahko bi aggregate rešitev poizvedbe uporabili kot numerične atribute.

Agregati rešitev poizvedbe

Vpeljava agregatnih funkcij in s tem dobiemo numerične rešitve

Primeri aggregatov osnovne relacije

- $\text{count}_{(E)} r_{vsebuje.obliko}(e_{14}, E, O) = 3$
- $\text{count}_{(O)} r_{vsebuje.obliko}(e_{14}, E, O) = 3$
- $\text{count.unique}_{(O)} r_{vsebuje.obliko}(e_{14}, E, O) = 2$

Primeri aggregatov izvedenih oblik relacije

- $\text{count}_{(ID.e)} r_{vsebuje.obliko}(e_{14}, ID.e, krog) = 2$
- $\text{count}_{(ID.e)} r_{vsebuje.obliko}(e_{14}, ID.e, kvadrat) = 1$
- $\text{count}_{(ID.e)} r_{vsebuje.obliko}(e_{14}, ID.e, trikotnik) = 0$

Tabela za običajno strojno učenje

Enaka kot prej, le da atributi so sedaj numerični in ustrezajo agregatom.

Pozor: za numerične lastnosti možni nadaljnji agregati

- mean: povprečna vrednost numerične lastnosti
- median: mediana, srednja vrednost numerične lastnosti
- minimumi, maksimumi, kvantili in podobno

Matej's Alternative: Tests with Aggregates

$X = \frac{\text{agg}_1}{X_1} \frac{\text{agg}_2}{X_2} \dots \frac{\text{agg}_m}{X_m} \ell_1 \wedge \ell_2 \wedge \dots \wedge \ell_r$, tests of the form $X \leq v$

Literals ℓ_i :

- Involve variables X_1, X_2, \dots, X_m
- ℓ_1 involves a variable from the target predicate

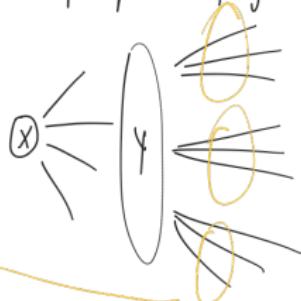
agg_i are aggregation functions

- sum, mean, min or max for numeric variables
- mode or count for discrete variables

modus (tisti kateri največkrat pojavi)

Aggregate Examples

X se povezuje s vseimi, ki jih univerzalno omogoča $\Rightarrow Y$



Stevilo kopije vodič Y

prvič vodec Y je

lubo vodilico med temi

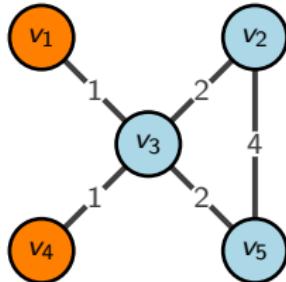
predikat aggregata edge

- $\underset{Y}{\text{count edge}}(X, Y, T)$; $\boxed{\text{degree}}$ of X

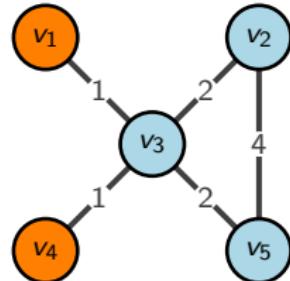
- $\underset{Y}{\text{count edge}}(X, Y, T) \wedge T = 4$: number of edges of type 4 involving X

- $\underset{Y}{\text{mean count edge}}(X, Y, T_1) \wedge \underset{Z}{\text{edge}}(X, Z, T_2)$: mean degree of the neighbors of X

delam povprečno kopijo vodič, ki so povezana z Y



Simple Graph and Its Predicate Representation



graf bi lahko definiran z enim predikativno edge
(tukaj smo že z degree)

Input predicates

$edge(v_1, v_3, 1)$
 $edge(v_2, v_3, 2)$
 $edge(v_2, v_5, 4)$
 $edge(v_3, v_4, 1)$
 $edge(v_3, v_5, 2)$

$degree(v_1, 1)$
 $degree(v_2, 2)$
 $degree(v_3, 4)$
 $degree(v_4, 1)$
 $degree(v_5, 2)$

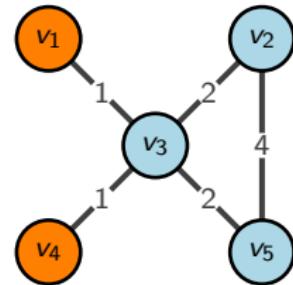
$class(v_1, \ominus)$
 $class(v_2, \oplus)$
 $class(v_3, \oplus)$
 $class(v_4, \ominus)$
 $class(v_5, \oplus)$

st. povratak iz nekega Target

vrednica

v_1, v_4 negativni vrednosti

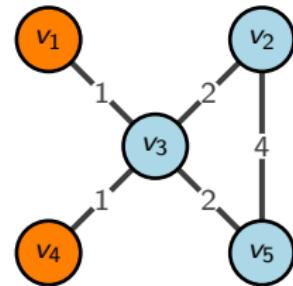
Test Example (1)



Query $\text{edge}(X, Y, T) \wedge T = 4 \equiv \text{edge}(X, Y, 4)$ successful

- Result $X = v_2$, $Y = v_5$
- Test outcome is *True*

Test Example (2)



Query $\text{edge}(X, Y, T) \wedge X = v_3 \wedge T = 4 \equiv \text{edge}(v_3, Y, 4)$ fails

- Result: there is no such value of Y , Y is not defined
- Test outcome is *False*, Y can not be used in the right subtree

Ordering the Set of Tests

Vašo uređuju teste? (kar smo prej povedali sas + drugimi besedami)
↓

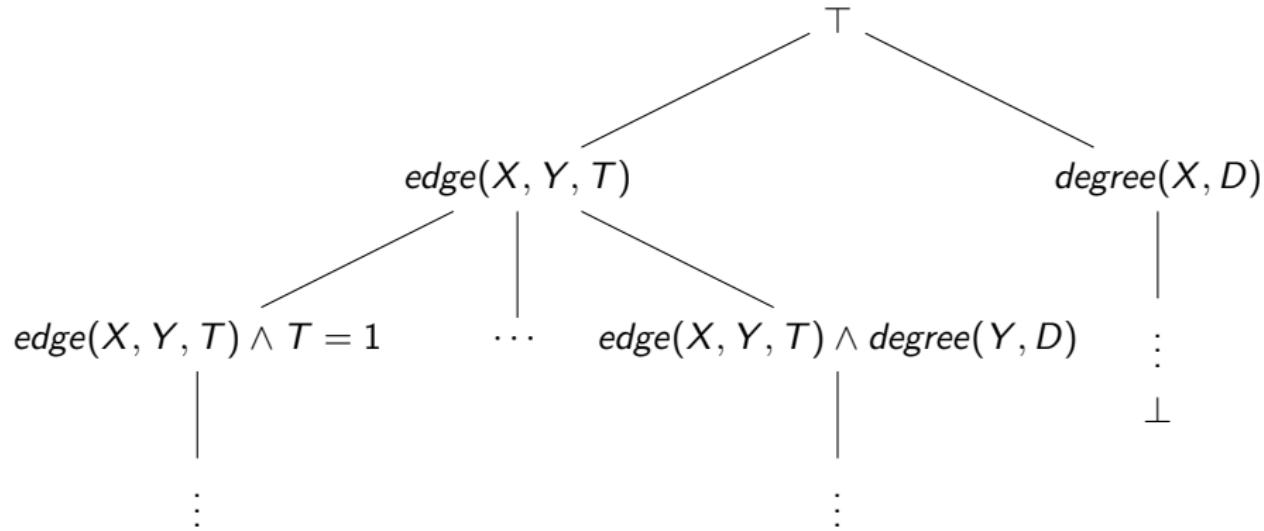
Start with an empty conjunction

- The query succeeds for all training objects
- Most **general** query

Continue in general to specific manner

- Add a literal ℓ to T , $T \wedge \ell$ is at least as **specific** as T
- The corresponding query succeeds for less training objects
- ℓ can be based on an input predicate, will typically introduce variables

An Example of A General-to-Specific Ordering Tree



Enumerates the tests from \mathcal{T} in a general-to-specific order.

Timeline and References

Top-Down Induction of Decision Trees, TDIDT

- Hunt et al 1966: Concept Learning System, inception
- Hyafil and Rivest 1976: Finding optimal tree is NP-complete
- Breiman 1984, Quinlan 1986: CART, ID3 (later C4.5)
- Ho 1995, Breiman 2001: ensembles of trees, random forests

First-order logic and relational decision trees

- Blockeel and De Raedt 1998: TILDE, Prolog implementation
- Assche et al 2006: relational random forests
- Petković et al 2022: Re3py efficient implementation in Python