Uvod u stabla odlučivanja

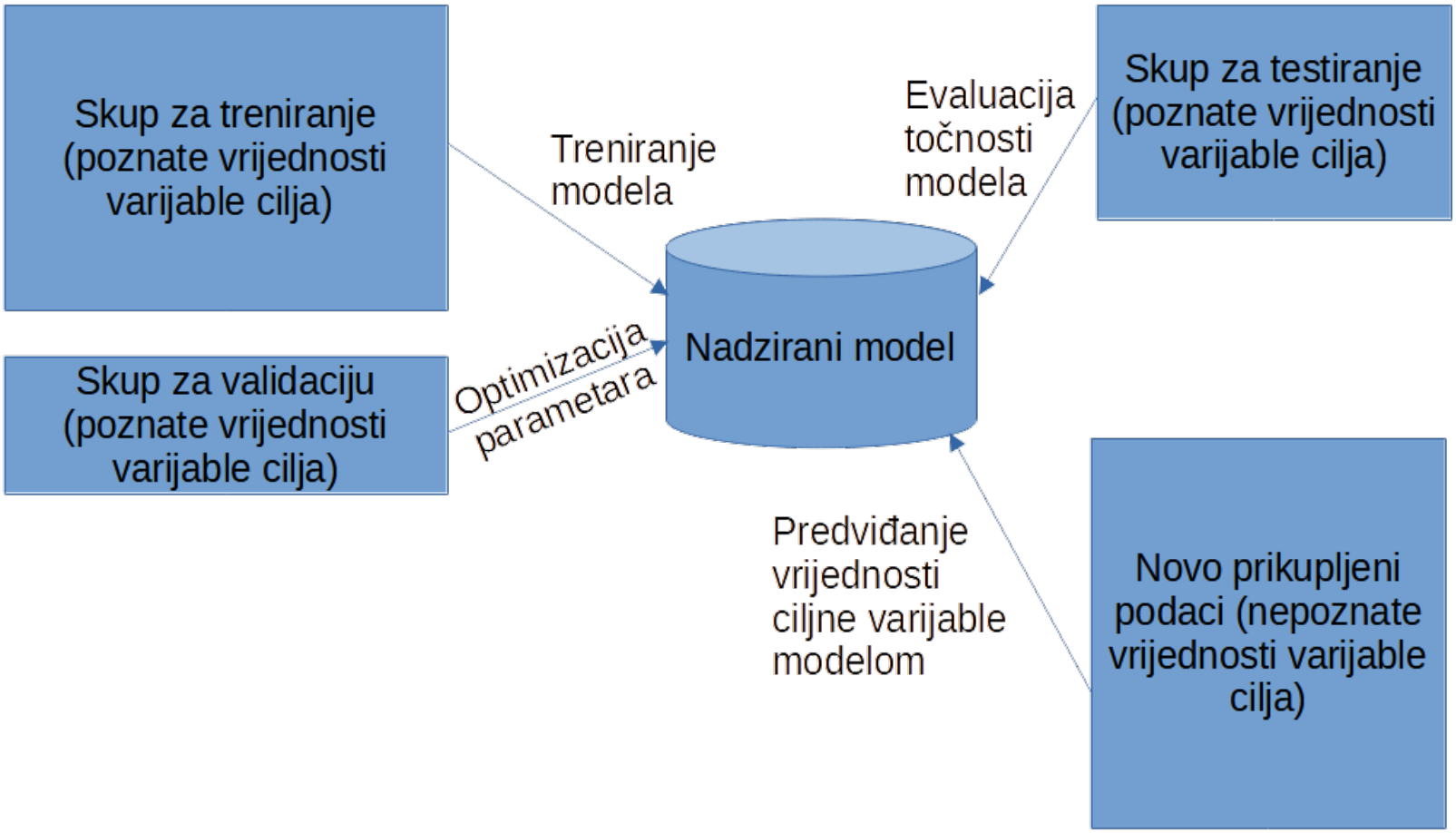
Matej Mihelčić

Prirodoslovno-matematički fakultet, Sveučilište u Zagrebu

matmih@math.hr

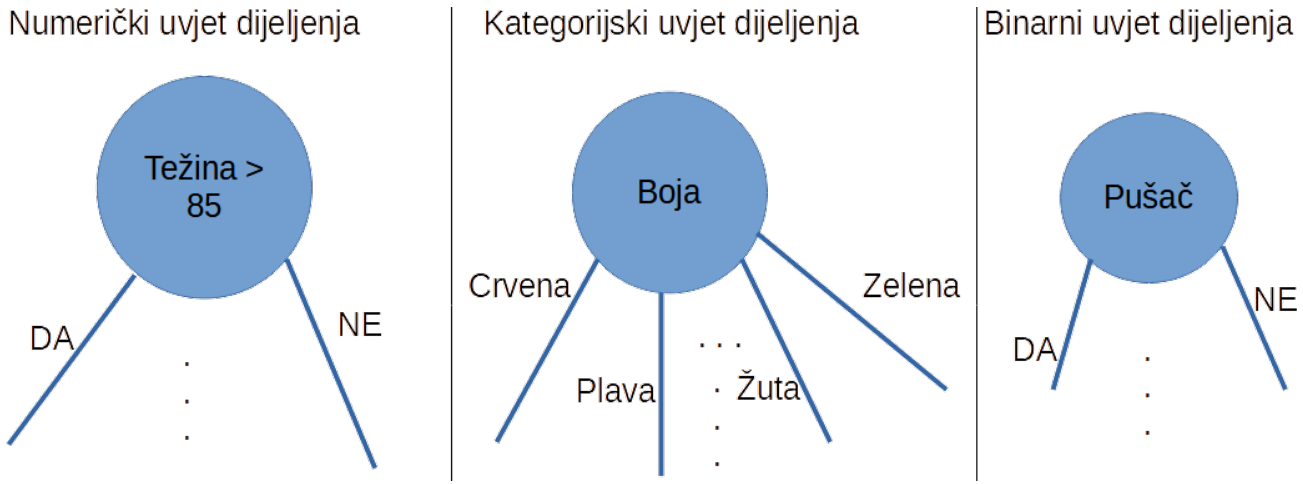
06. ožujka, 2026.

Model nadziranog učenja



Stablo odlučivanja

- struktura podataka stabla (može biti generalno ili binarno)
- uz informacije o djeci i roditeljima, dodatno sadrži informacije o **uvjetu dijeljenja**, sadržanim **entitetima** iz skupa podataka za treniranje, te **vrijednostima** njihove **varijable cilja**.



Atribut koji koristimo za podjelu podataka se zove i **atribut podjele**, a par (atribut, vrijednost) se zove i **točkom podjele**.

Stablo odlučivanja

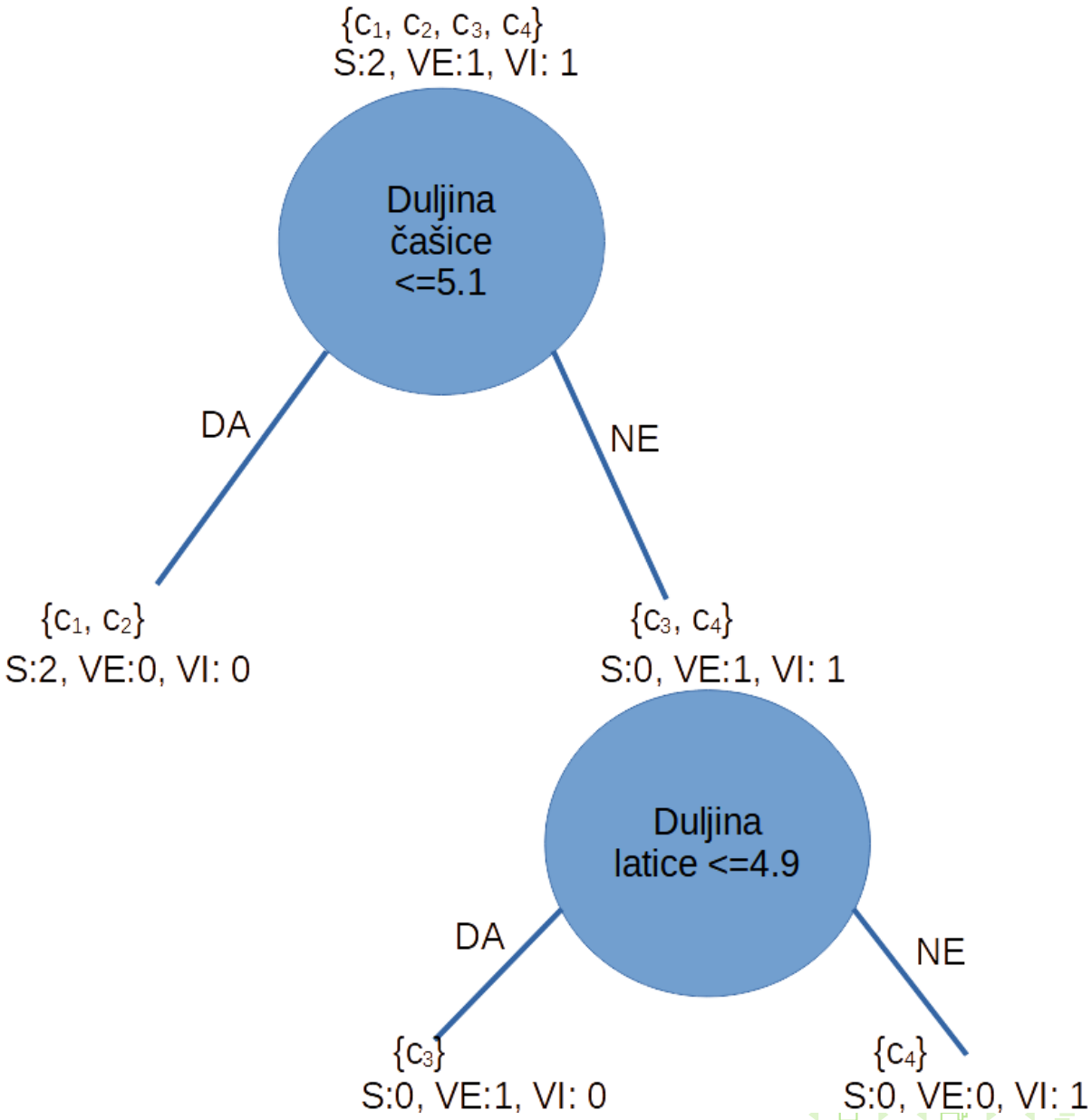
Entitet	Duljina čašice	Širina čašice	Duljina latice	Širina latice	Vrsta
c_1	5.1	3.5	1.4	0.2	Iris-setosa
c_2	5.0	3.3	1.4	0.2	Iris-setosa
c_3	6.9	3.1	4.9	1.5	Iris-versicolor
c_4	7.7	3.8	6.7	2.2	Iris-virginica

IRIS SETOSA

IRIS VERSICOLOR

IRIS VIRGINICA

Stablo odlučivanja



Stablo odlučivanja

Pretpostavimo da želimo predvidjeti koje je klase cvijet c_5 s vrijednostima:

`6.9,3.1,5.4,2.1,?`

Prema gornjem modelu ispitujemo:

- $6.9 \leq 5.1$? Odgovor je **NE**.
- Primjer proslijedimo u **desno** dijete korjena.
- $5.4 \leq 4.9$? Odgovor je **NE**.
- Primjer proslijedimo u desno dijete čvora.
- Desno dijete je **list** (nema djece), sadrži $\{c_4\}$ koji je klase `Iris-virginica`.
- **Predviđamo:** $c(c_5) =$ `Iris-virginica`.

Predviđanje je u ovom slučaju točno!
Kako pronaći uvjete dijeljenja?

Algoritam C4.5

- Podržava isključivo klasifikacijske probleme.

Algoritam C4.5

Ulaz: Skup za treniranje $D = \{(\vec{a_1}, y_1), \ldots (\vec{a_n}, y_n)\}$, skup atributa $A = \{a_1, \ldots a_m\}$

Izlaz: trenirano stablo C4.5

if svi entiteti iz D imaju istu vrijednost klase C **then**

T je stablo s jednim listom labeliranim klasom C

return T

end if

if $A = \emptyset$ ili svi entiteti imaju identične vrijednosti atributa **then**

T je stablo s jednim listom labeliranim klasom koju sadrži najveći broj entiteta iz skupa podataka

return T

end if

izaberi najbolji atribut za podjelu $a_* \in A$ ▷ Nastavak slijedi

---

Algoritam C4.5

---

for svaku kategorijsku vrijednost $a_*^v \in a_*$ **do** ▷ Nastavak
generiraj granu za vrijednost a_*^v čvora, podskup elemenata označavamo $D^v \in D$
if $D^v = \emptyset$ **then**
stvorimo list N_v i labeliramo ga s klasom koju sadrži najveći broj primjera u čvoru roditelju
$T \leftarrow T \cup N_v$
else
$T \leftarrow T \cup C4.5(D^v, A \setminus \{a_*\})$
end if
end for
return T

Algoritam C4.5

- C4.5 koristi **entropiju** i **porast informacije** za određivanje atributa za podjelu podataka.
- Označimo s Y ciljnu klasu problema.
- $H(D) = \sum_{y \in Y} p(y) \log_2 p(y)$. $p(y)$ je vjerojatnost pojavljivanja entiteta klase y u skupu D ($p(y) = \frac{|D_y|}{|D|}$).
- $PI(D, a) = H(D) - \sum_{v=1}^{V} \frac{|D^v|}{|D|} H(D^v)$.
- $a_* = \arg\max_{a \in A} PI(D, a)$

Algoritam C4.5 - dijeljenje numeričkih atributa

- Neka je $a_k \in A$ neki numerički atribut.
- Sortiramo vrijednosti od a_k uzlazno. Označimo ih s $a_{k,1},\ a_{k,2},\ \ldots\ , a_{k,n}$.
- Određujemo pragove $t_{a_{k,i}} = \dfrac{a_{k,i} + a_{k,i+1}}{2},\ i = 1, 2, \ldots, n-1.$
- Svaki prag dijeli skup entiteta na dva podskupa: $D^+(t)$ - skup entiteta koji imaju vrijednost veću od t i $D^-(t)$ - skup entiteta koji imaju vrijednost $\leq t$.
- $PI(D, a_k) = max_{t \in t_{a_k}} H(D) - \sum_{\lambda \in \{+,-\}} \dfrac{|D^\lambda(t)|}{|D|} H(D^\lambda(t)).$

Algoritam C4.5 - rad s nedostajućim vrijednostima

U prisustvu nedostajućih vrijednosti, javljaju se dva problema:

- Kako izabrati atribut dijeljenja u prisustvu nedostajućih vrijednosti?
- Kako dodijeliti entitet čvoru ukoliko ima nedostajuću vrijednost za neki atribut dijeljenja?

Izabir atributa dijeljenja ćemo raditi na sljedeći način:

- Promotrimo skup za treniranje D i atribut a. $\tilde{D}$ će označavati podskup entiteta iz skupa za treniranje takvih da oni nemaju nepoznatu vrijednost za atribut a. $\tilde{D}_k \subseteq \tilde{D}$ označava podskup elemenata koji imaju vrijednost ciljne labele y_k.
- Svi entiteti e_k, na početku izgradnje stabla, imaju težinu $w_{e_k} = 1$.
- Računamo:
  - $\rho = \dfrac{\sum_{e \in \tilde{D}} w_e}{\sum_{e' \in D} w'_e}$
  - $\tilde{r}_v = \dfrac{\sum_{e \in \tilde{D}^v} w_e}{\sum_{e' \in \tilde{D}} w'_e}, \ v = 1, \ldots, V$
  - $\tilde{p}_k = \dfrac{\sum_{e \in \tilde{D}_k} w_e}{\sum_{e' \in D} w'_e}, \ k = 1, \ldots, |Y|$

Algoritam C4.5 - rad s nedostajućim vrijednostima

- $PI(D, a) = \rho \cdot PI(\tilde{D}, a) = \rho \cdot (H(\tilde{D}) - \sum_{v=1}^{V} \tilde{r}_v H(\tilde{D}^v))$.
- $H(\tilde{D}) = -\sum_{k=1}^{|Y|} \tilde{p}_k \log_2 \tilde{p}_k$.

Dodjeljivanje entiteta čvorovima radimo:

- Ukoliko entitet w_e ima definiranu vrijednost za atribut dijeljenja, entitet dodijelimo čvoru prema vrijednosti (na standardan način) uz nepromijenjenu težinu w_e.
- Ukoliko entitet w_e ima nedostajuću vrijednost za atribut po kojem dijelimo primjere, taj entitet ćemo dodijeliti svakom od V čvorova djece. Težina entiteta u tim čvorovima će biti jednaka $\tilde{r}_v \cdot w_e$, gdje w_e označava težinu entiteta u čvoru roditelja.
- Kod predviđanja možemo izračunati težinsku ocjenu da primjer pripada svakoj klasi. Primjer konačno dodijelimo klasi s najvećom težinskom ocjenom.

Algoritam C4.5 - podrezivanje stabla

Podrezivanje stabla unutar algoritma C4.5 slijedi korake:

- Za svaki list stabla promatramo broj sadržanih primjera iz skupa za treniranje N i broj primjera za treniranje s krivom vrijednosti ciljne klase (vrijednost drugačije od one koju ima većina primjera lista), takve primjere označimo s E.
- Definiramo razinu pouzdanosti CF (obično 0.25).
- Računamo vjerojatnost p, takvu da $CF = \sum_{i=0}^{E} \binom{N}{i} p^i \cdot (1-p)^{N-i}$, za zadani CF.
- Koristimo izračunati p, u oznaci $U_{CF}(E, N)$ za procjenu greške u listu. Greška se računa kao $N \cdot U_{CF}(E, N)$.
- Procijenjena greška podstabla je suma procijenjenih grešaka listova tog podstabla.
- Ukoliko je procijenjena greška stabla bez nekog podstabla manja nego s njime, to podstablo izrežemo (odnosno korijen tog podstabla proglasimo listom).

Algoritam CART

- Podržava klasifikacijske i regresijske probleme.

Algoritam CART

Ulaz: Skup za treniranje $D = \{(\vec{a_1}, y_1), \ldots (\vec{a_n}, y_n)\}$, skup atributa $A = \{a_1, \ldots a_m\}$

Izlaz: trenirano stablo CART

Dodijeli sve entitete skupa za treniranje korjenu (T je jednak korjenu).
if svi entiteti iz D imaju istu vrijednost klase C ili je broj entiteta u čvoru premali **then**
    return T
end if
izaberi najbolji atribut za podjelu $a_* \in A$ na dva podstabla
$D_{lijevo} \leftarrow$ uzorci koji pripadaju lijevom pod-stablu
$D_{desno} \leftarrow$ uzorci koji pripadaju desnom pod-stablu ▷ Nastavak slijedi

---

Algoritam CART

$T \leftarrow T \cup CART(D_{lijevo}, A)$. ▷ Nastavak
$T \leftarrow T \cup CART(D_{desno}, A)$.
return T

CART algoritam koristi Gini indeks za izabir atributa za dijeljenje podataka u slučaju klasifikacijskog zadatka. $Gini(t) = 1 - \sum_{y \in Y} p(y)^2$. Dobit pri izabiru atributa za dijeljenje se mjeri kao:
$Dobit_G(P, a) = Gini(P) - q \cdot Gini(L_a) - (1 - q) \cdot Gini(R_a)$, gdje P označava skup svih entiteta čvora roditelja, L_a skup entiteta lijevog djeteta, a R_a skup svih entiteta desnog djeteta. $q = \frac{|L_a|}{|P|}$.

- Dijeljenje numeričkih i binarnih atributa se odvija kao i kod C4.5, samo koristeći Gini indeks kod klasifikacijskih zadataka.
- Kategorijski atributi kod algoritma CART uvijek proizvode podjelu na dva dijela.

Algoritam CART

- Npr. atribut s vrijednostima {crveno, plavo, žuto, zeleno, narančasto} možemo podijeliti kao: {crveno, plavo, žuto, zeleno} i {narančasto} ili {crveno, plavo, žuto}, {zeleno, narančasto} itd.
- Kod regresijskih zadataka se često koristi srednja kvadratna pogreška (eng. *Mean squared error*).
- Za evaluaciju dobrote čvora, izračuna se srednja vrijednost ciljne labele svih sadržanih entiteta $\hat{y} = \frac{\sum_{e \in N} y(e)}{N}$.
- Zatim računamo $SKP(N) = \sum_{i=1}^{N} (y_i - \hat{y})^2$.
- Dobit pri izabiru nekog atributa za dijeljenje računamo kao: $Dobit_{SKP}(P, a) = SKP(P) - q \cdot SKP(L_a) - (1 - q) \cdot SKP(R_a)$, gdje $q = \frac{|L_a|}{|P|}$.

Algoritam CART - rad s nedostajućim vrijednostima

- CART algoritam penalizira atribute s nedostajućim vrijednostima na način da reducira mjeru njihove dobrote za $x\%$, gdje $x\%$ označava postotak nedostajućih vrijednosti za taj atribut u našem skupu podataka.
- U prisutnosti nedostajućih vrijednosti, CART računa **surogat** atribute dijeljenja.
  - Umjesto da izabere samo atribut sa najvećom vrijednosti mjere kvalitete, održava listu top surogat kandidata. Surogat kandidati su alternativni atributi dijeljenja koji dobro predviđaju gdje bi primjer trebalo smjestiti ukoliko ima nedostajuću vrijednost za glavni atribut dijeljenja (u lijevo ili desno dijete). Rangiraju se prema asocijacijskoj mjeri koja određuje koliko je točka dijeljenja bolja od standardnog pravila koje predviđa da svi takvi primjeri idu u dijete s većim brojem primjera.
  - Ukoliko neki primjer ima nedostajuću vrijednost za atribut dijeljenja, smještamo ga u lijevo ili desno dijete prema najviše rangiranom surogat atributu, ukoliko i za njega ima nedostajuću vrijednost uzimamo drugi po redu rangirani surogat itd.

Algoritam CART - podrezivanje stabla

CART provodi podrezivanje tako da:

- Prvo izgradi stablo bez ograničenja (izgradnja prestaje kada se dobije podskup elemenata s istom vrijednosti ciljne varijable ili kada čvor sadrži premalo elemenata).
- Stablo se postupno reže koristeći formulu $Ra(T) = R(T) + \alpha|T|$. Gdje $R(T)$ označava točnost stabla na skupu za treniranje, $|T|$ broj listova stabla, α je numerički parametar koji se iterativno povećava i penalizira složenost stabla.
- U svakom koraku obrezivanja dobijemo novo stablo T_i, koje sadrži podskup čvorova originalnog stabla T.
- Postupkom dobijemo niz stabala: $T_0 \supseteq T_1, \ldots, \supseteq T_n$.
- Stabla u nizu se evaluiraju na skupu za testiranje (ili kros-validacijskom skupu), te se izabire pod-stablo s najmanjom pogreškom.

Algoritam QUEST

Algoritam QUEST podržava isključivo klasifikacijske zadatke. Postoji generalniji i kompliciraniji algoritam GUIDE[1] koji podržava klasifikacijske i regresijske zadatke.

Opisat ćemo glavne korake rada algoritma QUEST: a) pronalazak atributa za dijeljenje podataka, b) pronalazak točke dijeljenja, c) uvjet zaustavljanja, d) rad s nedostajućim vrijednostima.

Glavna karakteristika algoritma je da **ne radi iscrpno ispitivanje** kvalitete dijeljenja koristeći mjere teorije informacija, već koristi **statističke testove** za pronalazak atributa dijeljenja, te **statističke i numeričke** metode za pronalaženje točne točke podjele. To značajno **ubrzava** postupak stvaranja stabla.

[1] https://www3.stat.sinica.edu.tw/statistica/oldpdf/A12n21.pdf

Algoritam QUEST - traženje atributa dijeljenja

- Za svaki numerički atribut A_s, izračunaj ANOVA F test koji testira imaju li različiti podskupovi vrijednosti od A_s, za koje ciljna varijabla Y ima zadanu kategorijsku vrijednost $y_i, i = 1, \ldots, k$, istu prosječnu vrijednost i izračunaj odgovarajuću p-vrijednost po F statistici.
- Za svaki kategorijski atribut B_k izračunaj Pearsonov χ^2 test nezavisnosti B_k i Y, te izračunaj p-vrijednost s obzirom na χ^2 statistiku.
- Izaberi atribut s najmanjom p-vrijednosti, označimo ga At_{cand}.
- Ukoliko je najmanja p-vrijednost, koja odgovara At_{cand}, manja od α/M, gdje $\alpha \in [0, 1]$ označava definiranu razinu značajnosti (definira korisnik), a $M \in \mathbb{N}$ ukupan broj atributa u skupu podataka, izabiremo At_{cand} kao atribut dijeljenja.

Algoritam QUEST - traženje atributa dijeljenja

- Ukoliko je najmanja p-vrijednost, koja odgovara At_{cand}, jednaka ili veća od α/M
  - Za svaki numerički atribut A_s, izračunaj Levenovu F statistiku baziranu na apsolutnoj devijaciji vrijednosti podskupova A_s, koji sadrže vrijednosti ciljne labele $y_i, i = 1, \ldots, k$, od srednje vrijednosti podskupa zadane ciljne kategorije y_i. Time testiramo razlikuju li se varijance podskupova od A_s koji odgovaraju različitim vrijednostima klase Y. Izračunamo p-vrijednost za taj test.
  - Pronađi atribut s najmanjom p-vrijednosti, označimo ga At_{cand_1}.
  - Ukoliko je najmanja p-vrijednost manja od $\alpha/(M + M_1)$, gdje je M_1 broj numeričkih varijabli, biramo At_{cand_1} kao atribut dijeljenja. Inače, ne dijelimo dani čvor stabla.

Algoritam QUEST - traženje atributa dijeljenja, detalji

- Pretpostavimo da su u čvoru stabla t sadržani entiteti koji imaju $t_k \leq k$ različitih vrijednosti ciljne klase Y. F statistika za neprekidni atribut A_s je: $F_A = \frac{\sum_{j=1}^{t_k} N_{f,j}(t) \cdot (\bar{x}_j(t) - \bar{x}(t))^2/(t_k - 1)}{\sum_{n \in E_t} f_n (x_n - \bar{x}_{ind(y_n)}(t))^2/(N_f(t) - t_k)}$.

  $\bar{x}_j(t) = \frac{\sum_{n \in E_t |\ y(x_n)=y_j} f_n x_n}{N_{f,j}(t)}$, $\bar{x}(t) = \frac{\sum_{n \in E_t} f_n x_n}{N_f(t)}$.

  f_n je težina frekvencije asocirana s primjerom skupa za treniranje (algoritam dopušta da neki redak predstavlja više entiteta, tada se frekvencijska težina postavlja na vrijednost > 1). E_t je skup indeksa entiteta koji pripadaju čvoru stabla t. x_n je vrijednost atributa A_s primjera za treniranje s indeksom n. $ind(y_n)$ vraća indeks kategorijske vrijednosti koja odgovara y_n. $N_{f,j}(t) = \sum_{n \in E_t |\ y(x_n)=y_j} f_n$,

  $N_f(t) = \sum_{n \in E_t} f_n$.

Algoritam QUEST - traženje atributa dijeljenja, detalji

- Odgovarajuću p-vrijednost računamo kao: $p = P(F(t_k - 1, N_f(t) - t_k) > F_A)$, gdje $F(t_k - 1, N_f(t) - t_k)$ slijedi F distribuciju s $t_k - 1$ i $N_f(t) - t_k$ stupnjeva slobode.

Pearsonov χ^2 test računamo:

- Pretpostavimo da su u čvoru stabla t sadržani entiteti koji imaju $t_k \leq k$ različitih vrijednosti ciljne klase Y. Pearsonova χ^2 statistika za kategorijsku varijablu A_s s t_c različitih vrijednosti kategorijske varijable definira se kao: $X^2 = \sum_{j=1}^{t_k} \sum_{i=1}^{t_c} \frac{(n_{i,j} - \hat{m_{i,j}})^2)}{\hat{m_{i,j}}}$.

$$n_{i,j} = \sum_{n \in E_t |\ y(x_n) = y_j \ \wedge \ x_n = i} f_n,\ \hat{m}_{i,j} = \frac{n_{i,*} \cdot n_{*,j}}{n_{*,*}}.$$

$$n_{i,*} = \sum_{j=1}^{t_k} n_{i,j},\ n_{*,j} = \sum_{i=1}^{t_c} n_{i,j},\ n_{*,*} = \sum_{j=1}^{t_k} \sum_{i=1}^{t_c} n_{i,j}.$$

Algoritam QUEST - traženje atributa dijeljenja, detalji

- Odgovarajuća p-vrijednost je dana s $p = P(\chi^2_d > X^2)$, gdje χ^2_d slijedi χ^2 distribuciju sa stupnjem slobode $d = (t_k - 1) \cdot (t_c - 1)$.

Levenov F-test računamo:

- Za numerički atribut A_s računamo $z_n = |x_n - \bar{x}_{ind(y_n)}(t)|$.
- Levenov F-test za numerički atribut A_s je ANOVA F-statistika za z_n.

Algoritam QUEST - traženje točke dijeljenja

Algoritam QUEST uvijek radi binarno dijeljenje (čvor se dijeli na dvoje djece).

- Pretpostavimo da je atribut A_s izabrana za dijeljenje. Zanima nas kako odabrati točku dijeljenja. Ukoliko je A_s numerički atribut, određujemo točku d oblika $A_s \leq d$, a ako je kategorijski atribut, podskup K kategorijskih vrijednosti skupa svih kategorijskih vrijednosti od A_s.
- Ukoliko je izabrani atribut numerički:
  - Grupiramo kategorijske vrijednosti varijable Y u dvije super klase. Ukoliko Y ima samo dvije kategorije, u ovom koraku ne trebamo ništa raditi. Inače, računamo prosjek vrijednost atributa A_s za podskupove entiteta koji poprimaju različite vrijednosti varijable Y.
    - Ukoliko su prosjeci podskupova koji odgovaraju svakoj kategorijskoj vrijednosti varijable Y jednaki, kategorija varijable Y koja odgovara najvećem podskupu entiteta postaje super-klasa AK, a unija podskupova entiteta koji odgovaraju preostalim klasama pridijeljuju se super-klasi BK.

Algoritam QUEST - traženje točke dijeljenja

- Ukoliko je izabrani atribut numerički (nastavak):
  - Računamo prosjek vrijednost atributa A_s za podskupove entiteta koji poprimaju različite vrijednosti varijable Y (nastavak).
    - Ukoliko postoje dvije ili više kategorije koje odgovaraju jednako velikim podskupovima entiteta maksimalne veličine, skup entiteta koji odgovara kategoriji s najmanjim indeksom j pridijeljuje se super-klasi AK, a unija preostalih podskupova super-klasi BK.
    - Ukoliko prosjeci podskupova entiteta koji odgovaraju različitim kategorijama od Y nisu identični, primijeni k-means algoritam klasteriranja tako da su inicijalni centri klasteriranja postavljeni na prosječnu vrijednost podskupova entiteta koji odgovaraju kategorijama s najvećim prosjecima. Tim algoritmom podijeli elemente na dvije super-klase AK i BK.
    - Označimo srednju vrijednost uzorka (podskupa vrijednosti atributa) koji odgovara super-klasi AK s $\bar{x_{AK}}$, a varijancu sa s^2_{AK}. Analogno, $\bar{x_{BK}}$ i s^2_{BK} označavaju srednju vrijednost i varijancu uzorka koji odgovara super-klasi BK.

Algoritam QUEST - traženje točke dijeljenja

- Ukoliko je izabrani atribut numerički (nastavak):
  - Ako je $\min(s_{AK}^2, s_{BK}^2) = 0$ (jedan podskup sadrži identične vrijednosti), sortiramo varijance po veličini, tako da $s_1^2 \leq s_2^2$, a $\bar{x_1}, \bar{x_2}$ označavaju odgovarajuće prosjeke. Neka je $\varepsilon = 10^{-12}$ neka mala positivna konstanta. Ako $\bar{x_1} < \bar{x_2}$, $d = \bar{x_1}(1+\varepsilon)$, inače $d = \bar{x_1}(1-\varepsilon)$.
  - Ako je $\min(s_{AK}^2, s_{BK}^2) \neq 0$, primijenjujemo kvadratnu diskriminatnu analizu za određivanje točke dijeljenja d. Pretpostavljamo da je A_s normalno distribuiran u svakoj super-klasi sa izračunatom prosječnom vrijednošću i varijancom. Točka dijeljenja je među korjenima takvima da $P(x, AK|\ t) = P(x, BK|\ t)$ za čvor t. Imamo:

$$P(x, AK|\ t) = P(x|\ AK, t)P(AK|\ t) = P(AK|\ t)\frac{1}{\sqrt{2\pi s_{AK}^2}} e^{-\frac{(x-\bar{x_{AK}})^2)}{2s_{AK}^2}}$$

$$P(AK|\ t) = \sum_{j \in AK} P(j|t) = \sum_{j \in AK} \frac{P(j,t)}{\sum_{i \in E} P(i,t)},\ P(j,t) = \frac{\pi(j) N_{f,j}(t)}{N_{f,j}},$$

gdje $N_{f,j} = \sum_{n \in E|\ y(x_n) = y_j} f_n$, a $\pi(j)$ je a-priori vjerojatnost kategorije s indeksom j ciljne klase Y.

Algoritam QUEST - traženje točke dijeljenja

- Ukoliko je izabrani atribut numerički (nastavak):
  - Slučaj $\min(s_{AK}^2, s_{BK}^2) \neq 0$ (nastavak):
    Rješavanje jednadžbe $P(X, AK|t) = P(X, BK|t)$ je ekvivalentno rješavanju kvadratne jednadžbe oblika $ax^2 + bx + c = 0$ za:
    $a = s_{AK}^2 - s_{BK}^2$, $b = 2(\bar{x}_{AK} s_{BK}^2 - \bar{x}_B s_{AK}^2)$,
    $c = \bar{x}_{BK}^2 s_{AK}^2 - \bar{x}_{AK}^2 s_{BK}^2 + 2 s_{AK}^2 s_{BK}^2 \log \frac{p(AK|t) s_{BK}}{p(BK|t) s_{AK}}$.
  - Ukoliko postoji samo jedan relani korjen, biramo ga kao točku dijeljenja, uz uvjet da dijeljenje rezultira s dvoje neprazne djece. Ukoliko postoje dva realna korjena, biramo onaj koji je bliže $\bar{x_{AK}}$, pod uvjetom da dijeljenje rezultira s dvoje neprazne djece. Inače, koristimo $\frac{\bar{x_{AK}} + \bar{x_{BK}}}{2}$ kao točku dijeljenja.
- Ukoliko je atribut A_s kategorijski atribut s dvije kategorijske vrijednosti dobijemo prirodno binarnu podjelu. Ukoliko je A_s kategorijski atribut s više od dvije kategorijske vrijednosti, pretvaramo atribut A_s u numeričku varijablu ξ_s tako da pridijelimo najveću odvajajuću koordinatu kategorijama atributa.

Algoritam QUEST - pretvaranje kategorijskog atributa u numerički

- Neka je A_s kategorijski atribut s vrijednostima u skupu $a_1, \ldots, a_I$. Transformiramo A_s u numeričku varijablu ξ tako da je omjer sume kvadrata vrijednosti entiteta izvan-kategorija u odnosu na unutar-kategorija maksimizirana (kao kod klasteriranja).
- Pretvaramo svaku vrijednost x atributa A_s iz E u I-dimenzionalan vektor $\mathrm{v} = (v_1, \ldots, v_I)$, gdje $v_i = \begin{cases} 1, & x = b_i \\ 0, & \text{inače} \end{cases}$.
- Izračunamo srednju vrijednost vrijednosti atributa A_s svih entiteta u skupu za treniranje, te svih entiteta u skupu za treniranje koji imaju vrijednost varijable Y jednak kategoriji s indeksom j: $\bar{v} = \frac{\sum_{n \in E} f_n \mathrm{v}_n}{N_f}$, $\bar{v}_j = \frac{\sum_{n \in E |\ y(x_n) = y_j} f_n v_n}{N_{f,j}}$

Algoritam QUEST - pretvaranje kategorijskog atributa u numerički

- Računamo $I \times I$ matrice: $\mathrm{B} = \sum_{j=1}^{J} N_{f,j}(\bar{v}_j - \bar{v})(\bar{v}_j - \bar{v})^t$

  $T = \sum_{n \in E} f_n(v_n - \bar{v})(v_n - \bar{v})^t$
- Provedemo SVD dekompoziciju na matrici T (pozitivno semi-definitna) tako da dobijemo QDQ^t, gdje je Q ortogonalna $I \times I$ matrica, $D = diag(d_1, \ldots, d_I)$, takva da $d_1 \geq d_2, \cdots \geq d_I \geq 0$.

  Definirajmo $D^{-\frac{1}{2}} = diag(d_1^*, d_2^*, \ldots, d_I^*)$, gdje $d_i^* = d_i^{-\frac{1}{2}}$ ako $d_i > 0$, 0 inače.
- Provedimo SVD dekompoziciju na matrici $D^{-\frac{1}{2}} Q^t B Q D^{-\frac{1}{2}}$, i označimo s $\vec{a_m}$ svojstveni vektor koji odgovara najvećoj svojstvenoj vrijednosti matrice.
- Najveća odvajajuća koordinata vektora $\vec{v}$ se dobije projekcijom: $\xi = \vec{a_m}^t D^{-\frac{1}{2}} Q^t \vec{v}$.

Algoritam QUEST - kriterij zaustavljanja

- Ukoliko svi entiteti čvora imaju istu vrijednost ciljne klase, čvor ne dijelimo.
- Ukoliko svi entiteti čvora imaju identičnu vrijednost za sve atribute, čvor ne dijelimo.
- Izgradnja stabla staje kada se dosegne maksimalna dubina stabla definirana od strane korisnika.
- Ukoliko je veličina čvora manja od minimalne veličine čvora definirane od strane korisnika, čvor nećemo dijeliti.
- Ukoliko podjela rezultira čvorom djetetom čije je veličina manja od minimalne definirane veličine čvora djeteta, čvor nećemo dijeliti.

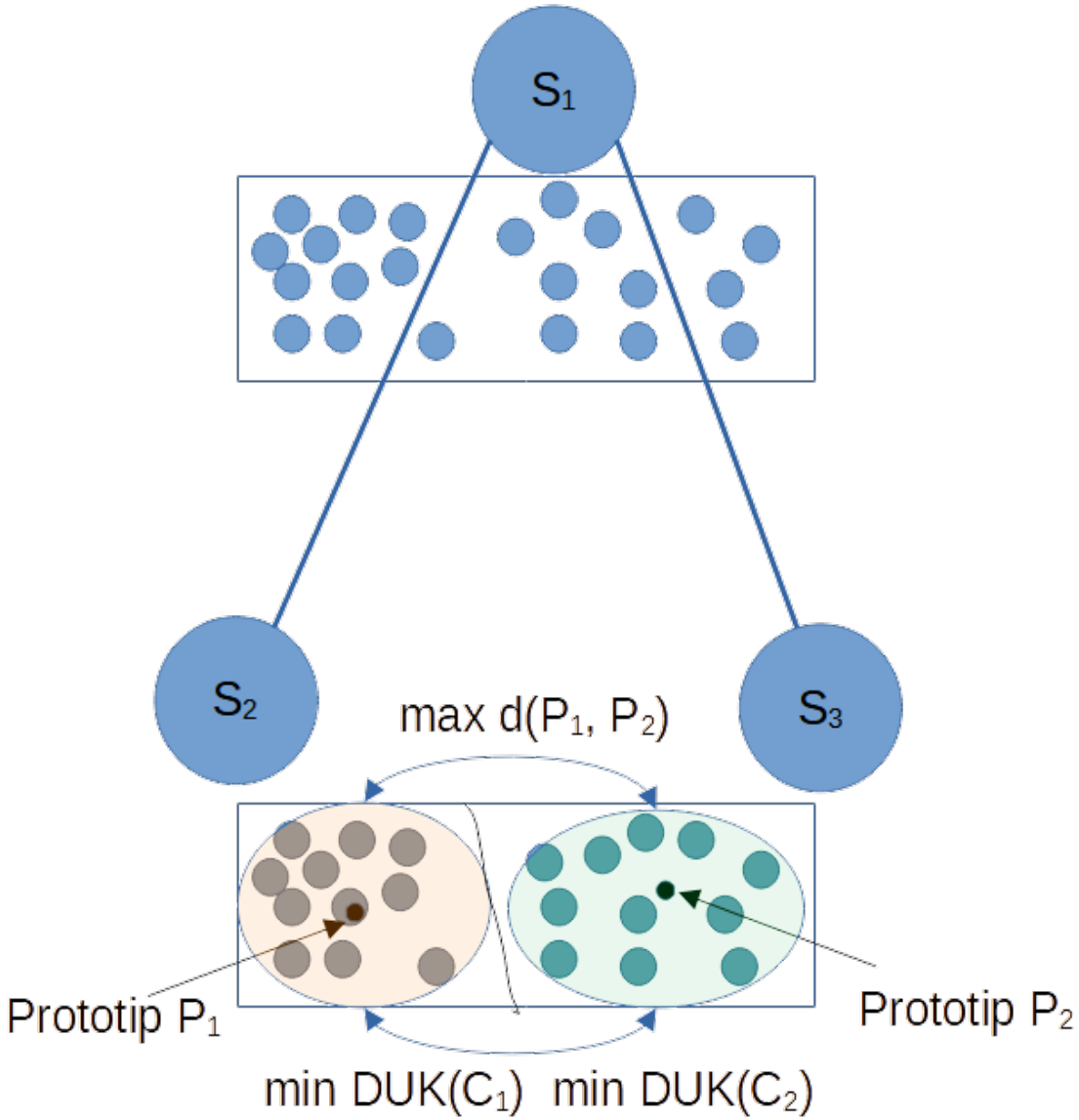
Algoritam QUEST - rad s nedostajućim vrijednostima

- Ukoliko entitet ima nedostajuću vrijednost za ciljnu klasu (prilikom treniranja), taj entitet se ignorira.
- Ukoliko entitet ima nedostajuće vrijednosti za sve aribute, taj entitet se ignorira.
- Ukoliko je težina frekvencije entiteta nedostajuća, nula ili negativna, taj entitet se ignorira.
- Inače, koristimo metodu surogat atributa i točaka dijeljenja na isti način kao i kod algoritma CART.

Algoritam prediktivnih stabala klasteriranja - PCT

- Usvajaju generalni princip indukcije **maksimalno separiranih klastera** na svakoj razini izgradnje stabla.
- Koristi **prototip** kao generalnu oznaku za centar klastera.
  - **Centroid** je specifična vrsta prototipa. Računa se kao prosječna vrijednost svih komponenata vektora entiteta koji se nalaze u klasteru. Centroid može i ne mora odgovarati nekoj točki iz klastera.
  - **Medoid** je također specifična vrsta prototipa koja **uvijek** odgovara jednoj (centralnoj) točki klastera.
  - **Mod** može biti specifična vrsta prototipa kada su entiteti opisani kategorijskim atributima.
- Prednost prediktivnih stabala klasteriranja je što mogu mijenjati način računanja prototipova i udaljenosti između klastera ovisno o danom problemu. To **značajno** povećava generalnost pristupa. Uz klasifikaciju i regresiju, podržava raspon zadataka, npr. višeciljna klasifikacija/regresija, hijerarhijska klasifikacija.
- Ovisno o izabranoj mjeri udaljenosti klastera, pristup može prilikom treniranja koristiti i entitete koji imaju nedostajuću vrijednost za **ciljnu klasu**.

Algoritam prediktivnih stabala klasteriranja - PCT



Algoritam prediktivnih stabala klasteriranja - PCT

- Porast informacija i Gini indeks se mogu smatrati posebnim vrstama udaljenosti između klastera koje koriste informacije samo o ciljnoj varijabli (minimiziraju razlike po pitanju vrijednosti ciljne varijable unutar klastera).
- Udaljenost između klastera se može računati: a) korištenjem informacija atributa, b) korištenjem informacije ciljne varijable, c) korištenjem kombinacije informacija o ciljnoj varijabli i atributima.
- Kod regresijskih zadataka kao mjera dobrote dijeljenja se koristi **redukcija varijance**, $\mathrm{Var}(E) - \sum_{E_i \in \mathcal{P}} \frac{|E_i|}{|E|} \mathrm{Var}(E_i)$.
- Kod klasifikacijskih zadataka se može koristiti Gini index ili porast informacija.

Algoritam prediktivnih stabala klasteriranja - PCT

Algoritam PCT
Ulaz: Skup za treniranje E
Izlaz: PCT

$(t^*, h^*, \mathcal{P}^*) = \text{NajboljiTest}(E)$
if $t^* \neq none$ **then**
  for each $E_i \in \mathcal{P}^*$ **do**
    $tree_i = \text{PCT}(E_i)$
  end for
  return $\text{node}(t^*, \bigcup_i \{tree_i\})$
else
  return $\text{list}(\text{Prototip}(E))$
end if

Algoritam NajboljiTest
Input: Skup za treniranje E
Output: najbolji test (t^*), njegova dobrota h^* i particija $(\mathcal{P}^*)$

$(t^*, h^*, \mathcal{P}^*) = (none, 0, \emptyset)$
for each test t **do**
  $\mathcal{P} =$ part. od E ind. od t
  $h = \text{Var}(E) - \sum_{E_i \in \mathcal{P}} \frac{|E_i|}{|E|} \text{Var}(E_i)$
  if $(h > h^*) \wedge \text{Prihvatljivo}(t, \mathcal{P})$ **then**
    $(t^*, h^*, \mathcal{P}^*) = (t, h, \mathcal{P})$
  end if
end for
return $(t^*, h^*, \mathcal{P}^*)$

Algoritam prediktivnih stabala klasteriranja - PCT

- Kriterij zaustavljanja se bazira na F-testu. Podjela mora **značajno** reducirati varijancu. Npr. podjela na dva djeteta:
- $F = \dfrac{SS/(n-1)}{(SS_L + SS_r)/(n-1)}$.
- F-test ima dva stupnja slobode, $d_1 = K - 1$ (broj grupa -1), ovdje $K = 2$ jer dijeljenje radimo na dva djeteta, te $d_2 = N_t - K$, gdje je N_t broj entiteta u čvoru koji dijelimo.
- SS je suma kvadratne pogreške vrijednosti entiteta u čvoru od prosjeka entiteta u čvoru, a SS_L i SS_R su sume kvadratne pogreške vrijendosti entiteta u lijevom/desnom djetetu od prosjeka vrijednosti entiteta u lijevom i desnom djetetu.
- Prediktivna stabla klasteriranja podržavaju grananje na dva djeteta ili više djece u slučaju kategorijskih atributa, dok je grananje isključivo na dva djeteta u slučaju numeričkih atributa.
- Entiteti s nedostajućom vrijednosti za atribut podjele se kao kod `C4.5` težinski propagiraju u sve čvorove djece.

Algoritam prediktivnih stabala klasteriranja - primjene

- Višeciljna klasifikacija:

Entitet	A_1	...	A_k	C_1	...	C_s
E_1	$a_{1,1}$	...	$a_{1,k}$	$C_{1,1}$	...	$C_{1,s}$
E_2	$a_{2,1}$	...	$a_{2,k}$	$C_{2,1}$	...	$C_{2,s}$
⋮	⋮	...	...		⋮	
E_{n-1}	$a_{n-1,1}$	...	$a_{n-1,k}$	$C_{n-1,1}$	...	$C_{n-1,s}$
E_n	$a_{n,1}$	...	$a_{n,k}$	$C_{n,1}$	...	$C_{n,s}$

- $C_{i,j} \in S$, gdje je S neki konačni skup kategorija.
- Točka dijeljenja se računa kao: $\sum_{k=1}^{s} Gini(t, Y_k) = \sum_{k=1}^{s}(1 - \sum_{y \in Y_k} p(y)^2)$.
- Možemo računati i entropiju: $\sum_{k=1}^{s} H(Y_k)$.

Algoritam prediktivnih stabala klasteriranja - primjene

- Višeciljna regresija, $C_{i,j} \in \mathbb{R}$:

Entitet	A_1	...	A_k	C_1	...	C_s
E_1	$a_{1,1}$	...	$a_{1,k}$	$C_{1,1}$	...	$C_{1,s}$
E_2	$a_{2,1}$	...	$a_{2,k}$	$C_{2,1}$	...	$C_{2,s}$
⋮	⋮	...	...		⋮	
E_{n-1}	$a_{n-1,1}$	...	$a_{n-1,k}$	$C_{n-1,1}$	...	$C_{n-1,s}$
E_n	$a_{n,1}$	...	$a_{n,k}$	$C_{n,1}$	...	$C_{n,s}$

- Izračunamo varijancu po svakoj ciljnoj varijabli $Y_k, k = 1, \ldots, s$.
- Normaliziramo varijance tako da varijanca svakog cilja podjednako pridonosi ukupnoj varijanci. Npr. dijeljenje varijance od Y_k u čvoru s varijancom tog cilja na cijelom skupu za treniranje.
- Ukupna varijanca se dobije kao $\sum_{k=1}^{s} Var_{norm}(Y_k)$.

Algoritam prediktivnih stabala klasteriranja - primjene

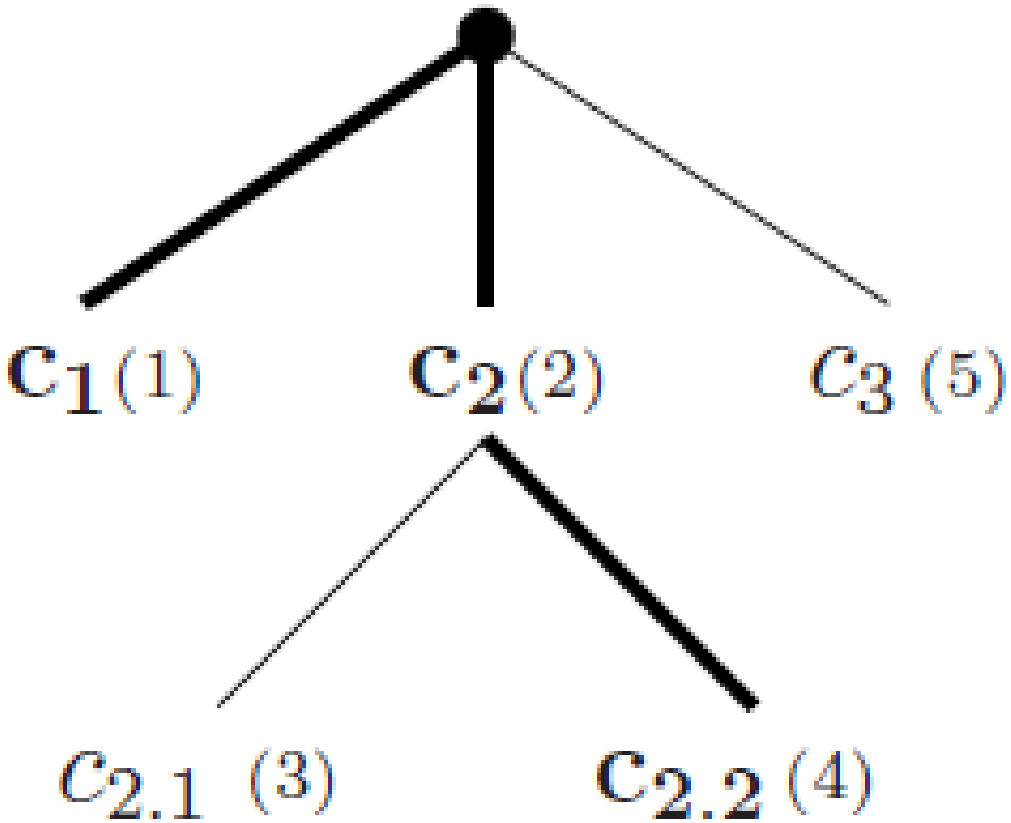
- Hijerarhijska klasifikacija:

Entitet	A_1	. . .	A_k	
E_1	$a_{1,1}$	. . .	$a_{1,k}$	
E_2	$a_{2,1}$	. . .	$a_{2,k}$	
. . .	. . .	. . .	. . .	. . .
E_{n-1}	$a_{n-1,1}$	. . .	$a_{n-1,k}$	
E_n	$a_{n,1}$	. . .	$a_{n,k}$	

Algoritam prediktivnih stabala klasteriranja - primjene

- Varijanca se računa kao $\frac{1}{|E|} \cdot \sum_{e_i \in E} d(L_i, \bar{L})^2$.
- $d(L_1, d_2) = \sqrt{\sum_{l=1}^{|L|} w(c_l) \cdot (L_{1,l} - L_{2,l})^2}$.
- L_i označava vektor reprezentacije podskupa klasa hijerarhije koje su dodijeljene elementu e_i. $\bar{L}$ je prosječna vrijednost vektora L_i za podskup entiteta koji pripadaju nekom čvoru stabla. $w(c_l)$ je težina važnosti razlike između klasa. Razlike u vektorima na višim razinama hijerarhije se penaliziraju više nego razlike na nižim razinama.

Algoritam prediktivnih stabala klasteriranja - primjene



$$L_k = \begin{matrix}(1)(2)(3)(4)(5)\\ [1,\ 1,\ 0,\ 1,\ 0]\end{matrix}$$