Umjetna inteligencija

Uvod u strojno učenje

Marko Čupić

IZDAVAČ

JAVNO DOSTUPNO NA WEB STRANICI JAVA.ZEMRIS.FER.HR/NASTAVA/UI

Ovo je popratni materijal za kolegij Umjetna inteligencija na Fakultetu elektrotehnike i računarstva Sveučilišta u Zagrebu. Tekst je usklađen s prezentacijama koje se koriste na tom kolegiju i okvirno ih prati. Uz tekst je pripremljena dodatna biblioteka napisana u programskom jeziku Java koja ilustrira algoritme koji se obrađuju u tekstu. Čitatelj se upućuje da istu skine i prati te pokreće primjere o koji se obrađuju u tekstu.

Prvo izdanje, travanj 2020.

Sadržaj

1. Uvod

Kroz tekst koji slijedi (a i neke kasnije cjeline), upoznat ćemo se s pojmom *strojno učenje* te nekim od pristupa i algoritama koje ono obuhvaća. Postoji mnoštvo načina na koje možemo definirati što bi bilo strojno učenje. Tako primjerice Kevin P. Murphy u svojoj knjizi *Machine Learning - A Probabilistic Perspective* strojno učenje definira kao "skup metoda koje u podatcima mogu automatski otkrivati obrasce, i potom te otkrivene obrasce iskorištavati pri budućem predviđanju podataka, ili obavljati druge zadatke odlučivanja u prisustvu nesigurnosti".

Ono što je zajedničko mnogim definicijama jest spoznaja da danas živimo u svijetu u kojem smo okruženi obiljem podataka, te da nam je interesantno razvijati programske sustave koji su sposobni iskorištavati te podatke, učiti iz njih i na temelju toga nuditi korisna ponašanja.

Podatci s kojima raspolažemo mogu biti *numerički* ili *kategorički*. Numerički podatci su podatci poput visine čovjeka, težine čovjeka, vremena koje je čovjeku potrebno da pretrči 100 metara i slično. Nad numeričkim podatcima možemo raditi računske operacije i imamo intepretaciju dobivenog rezultata. Kategorički podatci su skup informacija koji je podijeljen u određene grupe. Kategorički podatci dijele se u nominalne i ordinalne. Nominalni podatci su imenovani podatci; primjerice, zamislimo situaciju gdje studente anketiramo te ih kao jedno pitanja pitamo kako se osjećaju, i nudimo opcije "veselo", "tužno", "nervozno". Sličan primjer: pitamo za spol i nudimo "ženski", "muški". Nominalne podatke karakterizira činjenica da nemaju numeričke vrijednosti (ne možemo raditi aritmetičke operacije; što bi bio rezultat oduzimanja "nervozno" od "veselo"?) kao niti poretka (ne možemo reći je li "nervozno" veće/manje/jednako "tužno"). Ordinalni podatci su podatci s kojima i dalje ne možemo raditi aritmetiku, ali imaju definiran poredak. Primjerice, pitamo li studenta u anketi kako je zadovoljan kolegijem Umjetna inteligencija, te ponudimo opcije "jako zadovoljan", "umjereno zadovoljan", "ni zadovoljan, ni nezadovoljan", "umjereno nezadovoljan", "jako nezadovoljan" - dobili smo ordinalni podatak. Te podatke možemo uspoređivati, no s njima ne možemo raditi aritmetiku (koje bi bilo značenje "umjereno zadovoljan" minus "jako nezadovoljan"?).

Strojno učenje dijelimo na tri glavna područja:

- nadzirano učenje,
- nenadzirano učenje te

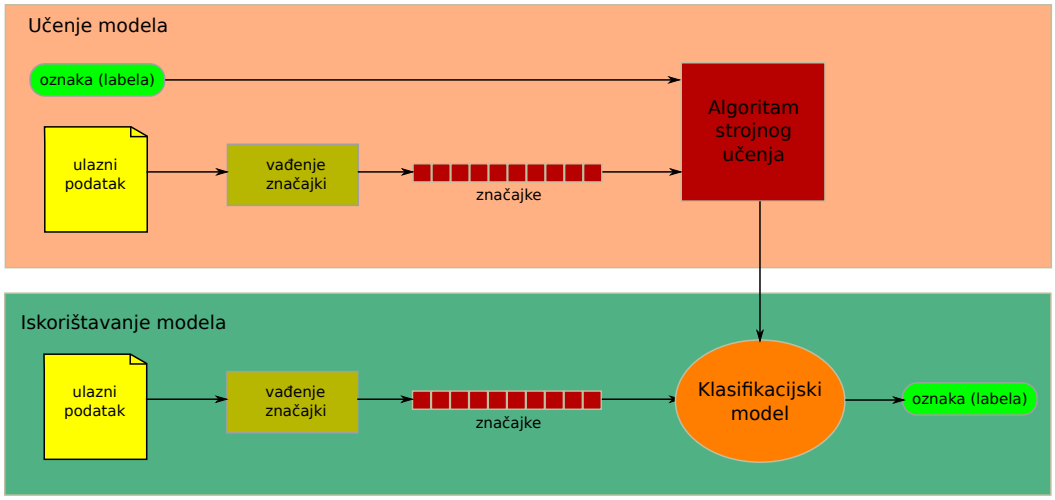
- podržano učenje.

1.1 Nadzirano učenje

Nadzirano učenje (engl. *supervised learning*) ima za cilj naučiti odnosno omogućiti obavljanje preslikavanja ulaza $\mathbf{x}$ na izlaz y, u skladu sa skupom primjeraka za učenje (engl. *training set*) koji je oblika $\mathscr{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$. Ovdje $\mathbf{x}_i$ predstavlja informacije o primjerku (to može biti vektor brojeva, slika, zvučni zapis, itd.) a y_i predstavlja podatak koji treba pridružiti tom primjerku. $\mathbf{x}_i$ ćemo zvati vektorom značajki primjerka, gdje svaka značajka govori o nekom aspektu primjerka. Primjerice, radimo li klasifikator koji će na temelju lista odrediti o kojem se drvu radi, značajke bi mogle biti: visina lista, širina lista, oblik lista, dominantna boja lista itd.

Ako je y_i kategorička varijabla, tada govorimo o klasifikacijskom problemu: primjerak je potrebno smjestiti u jedan od razreda; to bi bio slučaj s našim klasifikatorom, gdje bi y_i poprimao vrijednosti iz skupa {bor, hrast, jablan, breza, ...}. Ako je y_i numerička varijabla, tada govorimo o funkcijskoj aproksimaciji.

Kod modela strojnog učenja razlikujemo dvije faze: *faza učenja modela*, te *faza iskorištavanja modela*. Aktivnosti u svakoj od njih ilustrira slika u nastavku.



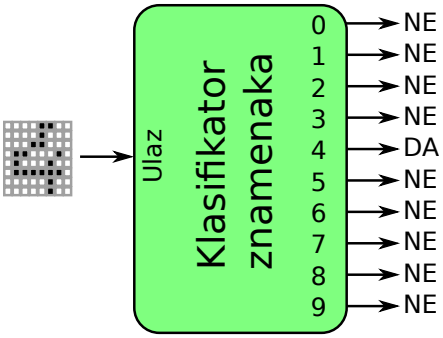
U fazi učenja, nad svakim se ulaznim podatkom obavlja vađenje značajki (engl. *feature extraction*). Na primjeru klasifikacije drveća an temelju lista, za svaki list koji imamo odredili bi prethodno spomenute značajke. Taj se vektor značajki zajedno s uparenom oznakom razreda šalje u algoritam strojnog učenja. Algoritam strojnog učenja na temelju ovih primjeraka uči optimalne parametre odabranog modela strojnog učenja - u ovom slučaju, odabranog modela koji obavlja zadaću klasifikacije (primjerice, uči težine umjetne neuronske mreže koju koristimo kao klasifikator).

U fazi iskorištavanja, korisnik postavlja ulazni podatak koji se ponovno obrađuje na jednak način: računaju se vrijednosti svih značajki i formira vektor značajki. Taj se vektor potom postavlja na ulaz naučenog modela strojnog učenja, koji na izlazu generira predviđanje (u našem slučaju, oznaku razreda u koji primjerak pripada). Jednom kad smo došli do faze iskorištavanja, gornji dio prethodne slike postaje višak i može se odbaciti.

Spomenimo da danas u određenim situacijama prethodno opisani "tijek" može biti izmijenjen tako da se izbaci središnji dio koji se bavi vađenjem ručno specificiranih značajki. Naime, ako je ulaz slika, jedan od tipičnih modela strojnog učenja koji se koristi za klasifikaciju su duboke neuronske mreže, a posebice konvolucijske neuronske mreže. Takvi modeli na ulaz mogu dobiti izvorni podatak ("čistu sliku") i tijekom učenja sami mogu u ranijim slojevima mreže otkriti koje su

značajke bitne i kako se računaju, te u kasnijim slojevima mreže razviti klasifikacijsko ponašanje. Za takve modele kažemo da rade posao od-kraja-do-kraja (engl. *end-to-end*): na jednom kraju dobiju "sirovi" ulazni podatak, a na drugom generiraju oznaku pripadnog razreda. Više o ovakvim modelima i njihovoj primjeni na obradu slika moći ćete naučiti na kolegiju Duboko učenje koji se nudi na diplomskom studiju.

Primjeri klasifikacijskih zadataka su razvrstavanje slika prema životinji koja je na slici, prepoznavanje rukom pisane znamenke sa slike, detekcija neželjene elektroničke pošte (engl. *spam*) i slično. Slika 1.1 ilustrira klasifikacijski problem gdje je ulaz skenirana slika znamenke, veličine 8×8 slikovnih elemenata; svaki slikovni element je crni ili bijeli. Klasifikatorski sustav treba na izlazu odrediti koju znamenku predstavlja ta slika, odnosno dani ulazni primjerak smjestiti u jedan od 10 razreda.



Slika 1.1: Klasifikacija znamenaka

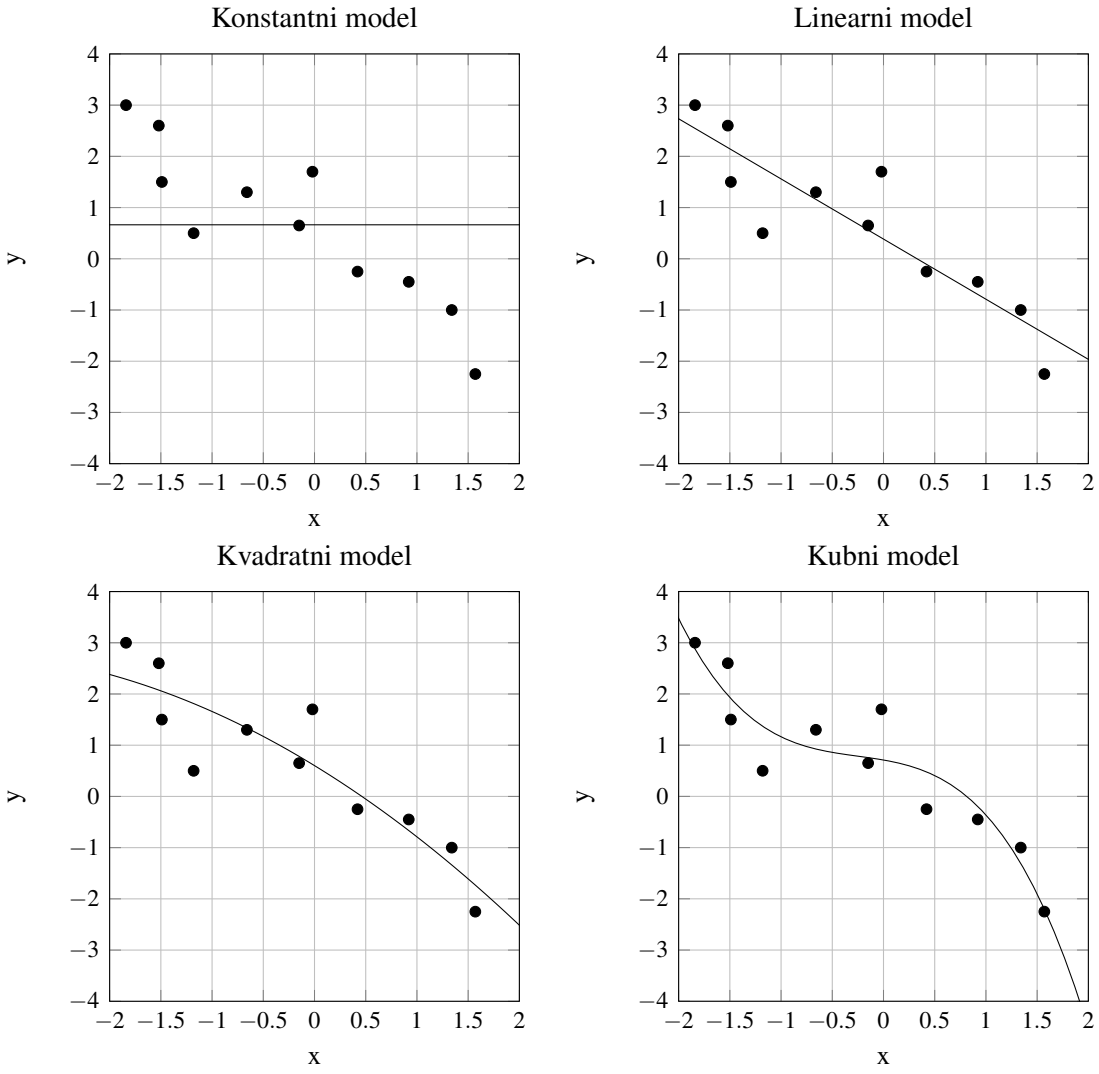
Primjeri regresije su predviđanje ukupnog broja bodova koje će student ostvariti na kolegiju na temelju bodova komponenata prve polovice semestra, predviđanje potrošnje plina na temelju povijesnih podataka, predviđanje potrošnje električne energije kućanstva na temelju vremenskih parametara (temperatura, tlak) i drugih podataka, itd.

Kao primjer regresije razmotrit ćemo skup podataka prikazan tablicom u nastavku.

x	y
-1.84	3.00
-1.52	2.60
-1.49	1.50
-1.18	0.50
-0.66	1.30
-0.15	0.65
-0.02	1.70
0.42	-0.25
0.92	-0.45
1.34	-1.00
1.57	-2.25

Potrebno je pronaći prikladan model $y = f(x)$ kako bismo iz modela mogli dobiti informaciju koliki će biti y za x-eve koji nisu u skupu primjeraka za učenje. Modela koji se mogu koristiti za regresiju ima mnoštvo i o nekima od njih (poput umjetnih neuronskih mreža) pričat ćemo kroz kasnije teme. Ovdje smo u nastavku prikazali četiri parametarska modela: konstantni ($y = a$), linearni ($y = ax + b$), kvadratni ($y = ax^2 + bx + c$) i kubni ($y = ax^3 + bx^2 + cx + d$). Svaki od ovih modela ima fiksan broj parametara koje je potrebno prilagoditi tako da model najbolje odgovara podatcima iz skupa za učenje.

Pri tome, pojam "najbolje odgovara" općenito može značiti različite stvari. Primjerice, može podrazumijevati da imamo definiranu funkciju pogreške (još je nazivamo i funkcijom gubitka) i da tijekom učenja želimo pronaći one parametre modela uz koje je ta funkcija pogreške nad podatcima koje imamo minimalna. Ako je to slučaj, tada govorimo o klasičnom optimizacijskom problemu. Kod regresije, uobičajena funkcija pogreške je polovično kvadratno odstupanje vrijednosti koju daje model i vrijednosti koja je zapisana uz primjerak, pa sumirano po svim primjercima.



Uz tako definiranu funkciju pogreške, prethodno spomenuti modeli uz koju je pogreška mala bili bi:

$$y = 0.6636 \quad \text{(konstantni model)}$$

$$y = 0.3850 - 1.1743 \cdot x \quad \text{(linearni model)}$$

$$y = 0.6019 - 1.2239 \cdot x - 0.1671 \cdot x^2 \quad \text{(kvadratni model)}$$

$$y = 0.7103 - 0.3496 \cdot x - 0.3102 \cdot x^2 - 0.4129 \cdot x^3 \quad \text{(kubni model)}$$

Što je model složeniji (ima više parametara), to će se moći bolje prilagoditi podatcima, što može biti nepoželjno.

Pogledajte sada još jednom prethodne četiri slike i četiri modela koja smo naučili. Ako smo svjesni da su podatci koje imamo samo jedanaest mjerenja ulaza i izlaza nekog procesa (a teoretski

bismo mogli napraviti još tisuće i tisuće mjerenja), odnosno predstavljaju samo mali slučajni uzorak iz inače potencijalno beskonačno velikog skupa podataka, te ako smo svjesni da je prilikom mjerenja nužno djelovao i određen šum zbog kojeg nam podatci nisu savršeni, s kojim od prethodna četiri modela bismo bili zadovoljni?

Primijetite da je prvi model vrlo krut - ima samo jedan parametar i vrlo smo ga slabo mogli prilagoditi skupu primjeraka koji imamo na raspolaganju. S druge strane, četvrti model je već dovoljno ekspresivan da je uspio modelirati i određeno "vijuganje" u skupu za učenje. Da smo uzeli još bogatiji model (kažemo model višeg kapaciteta), isti smo mogli još bolje prilagoditi podatcima, a uz dovoljno kapacitivan model mogli smo dobiti funkciju koja doslovno vijuga od mjerenja do mjerenja te izmjerenih 11 primjeraka modelira savršeno (naša definirana greška tada bi pala na nulu).

Pitanje koje si ovdje trebamo postaviti – i to je pravi problem strojnog učenja – koji od ovih modela je "najbolji" model? Da bismo odgovorili na to pitanje, uzet ćemo u obzir da ako u podatcima postoji šum ili ako postoje stršeće vrijednosti, presložen model moći će jako dobro modelirati i te pojave na uštrb dobre generalizacije. Prisjetite se: naš skup primjeraka za učenje nije kompletan opis modeliranog procesa: to je samo mali slučajan uzorak. Ono što želimo jest odabrati i naučiti onaj model koji bi savršeno odgovarao našem procesu, odnosno koji bi na beskonačno velikom skupu podataka koji bi opisivali naš proces imao minimalnu pogrešku. Problem je što nam taj skup nije dostupan (a i da jest, ako je beskonačno velik, nikada ne bismo završili postupak učenja pa opet nemamo koristi). Stoga bismo htjeli učiti na skupu primjeraka koji imamo, ali tako da minimiziramo grešku na "stvarnom" skupu. Htjeli bismo, dakle, da naš naučeni model i na podatcima koje nije vidio tijekom učenja radi dobro. Za model koji ima to svojstvo kažemo da dobro **generalizira**. Za model koji smo naučili do te mjere da ima jako malu pogrešku na skupu za učenje, ali radi veliku pogrešku na podatcima koje nije vidio, kažemo da je **prenaučen** (ili pretreniran). Takav model nam je u praksi beskoristan.

Postoji više načina kako se može utjecati na generalizacijske sposobnosti modela. Mi ćemo ovdje spomenuti pristup koji se naziva *unakrsna provjera*. Kako nemamo "pravi" beskonačno veliki skupo podataka, a htjeli bismo naučiti model koji dobro radi i na neviđenim primjerima, skup primjeraka s kojim raspolažemo, razdijelit ćemo u dva podskupa. Veći dio primjeraka smjestit ćemo u *skup za učenje* (engl. *training set*). Preostali manji dio primjeraka činit će skup za provjeru (engl. *validation set*). Koliko ćemo točno primjeraka uzeti u skup za učenje, a koliko u skup za provjeru, nije presudno. Neki autori (primjerice, Kevin P. Murphy u knjizi *Machine Learning - A Probabilistic Perspective*) uzimaju 80% raspoloživih primjeraka u skup za učenje, a preostalih 20% u skup za provjeru. Drugi autori navode drugačije podatke (primjerice, 70% u skup za učenje, 30% u skup za provjeru). Jednom kad smo primjerke razdvojili u dva skupa, provodimo učenje modela, ali samo na skupu za učenje (drugim riječima, kod naših modela iz prethodnog primjera, na temelju skupa za učenje korigirali bismo parametre modela). Povremeno, kako se model ponaša provjeravamo predočavanjem primjeraka iz skupa za provjeru; ta predočavanja koristimo isključivo da bismo odredili koliko model griješi nad primjercima tog skupa, ali pri tim predočavanjima ne korigiramo parametre (drugim riječima, model nad tim primjercima ne uči). Tijekom učenja, dogodit će se nešto interesantno: u ranim fazama učenja, kako prilagođavamo parametre modela, greška na skupu na učenje, kao i greška na skupu za provjeru, polagano će padati. Model će se prilagođavati generalnim trendovima u podatcima (učit će generalizirati). Međutim, u jednom trenutku, kako nastavljamo s učenjem nad primjercima iz skupa za učenje, model koji je dovoljno ekspresivan počet će se prilagođavati specifičnom šumu i stršećim vrijednostima iz tog skupa, a nad neviđenim podatcima počet će sve više griješiti. To ćemo identificirati time što ćemo uočiti da od tog trenutka na dalje pogreška koju računamo na skupu za provjeru počinje rasti! Trenutak u kojem to identificiramo je trenutak u kojem treba prekinuti učenje: nastavkom učenja model se počinje prenaučavati, a to ne želimo.

Ako naš model ima i parametre koji utječu na njegovu složenost, tada ćemo trebati na neki način odrediti i te parametre. Konkretno, četiri modela koja smo prikazali u prethodnom primjeru zapravo bismo mogli tretirati kao jedan model: polinomijalni, uz parametar r koji predstavlja red polinoma. Tako je naš konstantni model zapravo polinomijalni model reda 0, linearni model je zapravo polinomijalni model reda 1, i tako dalje. Koji bismo od modela u konačnici htjeli odabrati kao "najbolji" model? Da bismo odgovorili na to pitanje, skup podataka s kojim raspolažemo morat ćemo još razdijeliti:

- jedan dio primjeraka smjestit ćemo u *skup za učenje* (engl. *training set*; primjerice 40%),
- jedan dio primjeraka smjestit ćemo u *skup za provjeru* (engl. *validation set*; primjerice 30%),
- jedan dio primjeraka smjestit ćemo u *skup za ispitivanje* (engl. *test set*; primjerice 30%).

Nad skupom za učenje učit ćemo polinomijalni model reda 0, a s učenjem ćemo stati kad greška na skupu za provjeru počne rasti. Potom ćemo nad skupom za učenje učit ćemo polinomijalni model reda 1 i s učenjem stati kad greška na skupu za provjeru počne rasti. Potom ćemo nad skupom za učenje učit ćemo polinomijalni model reda 2 i s učenjem stati kad greška na skupu za provjeru počne rasti. Potom ćemo nad skupom za učenje učit ćemo polinomijalni model reda 3 i s učenjem stati kad greška na skupu za provjeru počne rasti. Jednom kad smo naučili modele uz svaki od parametara koji određuju kompleksnost modela, svaki od naučenih modela ispitat ćemo nad skupom za konačno ispitivanje. Model koji na tom skupu ima minimalnu pogrešku uzet ćemo kao najbolji model.

Primijetite da kod ovog pristupa i primjerci iz skupa za provjeru i primjerci iz skupa za konačno ispitivanje glume primjerke koje model prethodno nije vidio (u kontekstu učenja modela). Svaki od ta dva skupa, međutim, ima različitu zadaću.

Ako je skup primjeraka s kojim raspolažemo malen, tada podjela na opisani može biti problematična. Stoga su razvijene modifikacije opisanog postupka o kojima ćete imati prilike više naučiti na diplomskom studiju.

1.2 Nenadzirano učenje

Nenadzirano učenje (engl. *unspervised learning*) čine pristupi kod kojih je skup podataka oblika $\mathscr{D} = \{(\mathbf{x}_i)\}_{i=1}^N$. Drugim riječima, ništa osim samih primjeraka nije dano: nemamo numeričke ili kategoričke vrijednosti y_i koja je povezana s primjerkom. U postupke nenadziranog učenja spadaju postupci grupiranja (engl. *clustering*), postupci otkrivanja stršećih ili novih vrijednosti (engl. *outlier detection*, *novelty detection*), postupci smanjenja dimenzionalnosti (engl. *dimensionality reduction*), postupci otkrivanja veza između primjeraka (engl. *discovering graph structure*) te drugi.

Zadaća grupiranja jest na temelju sličnosti između primjeraka iste razdijeliti u određen broj razreda. Primjerice, slike životinja želimo grupirati u grupe, pri čemu se u jednoj grupi nalaze slike jedne životinjske vrste. Ovisno što znamo o podatcima, zadaća grupiranja može biti razdijeliti primjerke u unaprijed definirane razrede, ili pak može biti odrediti i potreban broj razreda i koji primjerak pripada u koji od razreda. Da bismo primjerke mogli grupirati, potrebno je biti u stanju računati njihovu udaljenost: trebamo neku metriku.

Nove ili stršeće vrijednosti su vrijednosti atributa koje imamo u skupu primjeraka, no koje su po nekom kriteriju dovoljno drugačije da ih želimo detektirati. Primjerice, zamislite neki proces kod kojega je izlaz y funkcija ulaza x u skladu s modelom $y = 2x$. Radite niz mjerenja, i dobijete skup primjeraka {(0,0),(1,2),(2,4),(3,60),(4,8),(5,10),(6,12)}. Podatak (3,60) očito je stršeći primjerak i ovdje predstavlja grešku (prilikom zapisivanja, osoba koja je bilježila podatke greškom je dopisala 0). S druge strane, podatci koji su dovoljno drugačiji ponekad mogu predstavljati i nešto novo odnosno nešto specifično za proces što u ostatku podataka nismo vidjeli. U oba slučaja, željeli bismo biti u stanju detektirati takve podatke.

Zada postupaka smanjenja dimenzionalnosti jest smanjiti broj atributa kojima opisujemo primjerke. Evo jednostavnog primjera: imamo skup primjeraka koji su točke u trodimenzijskom

prostoru, i svi leže u istoj ravnini. Zamislimo sada da smo u toj ravnini nacrtali novi koordinatni sustav: to bi bio dvodimenzijski koordinatni sustav, i svaki bismo primjerak mogli zapisati uporabom dva broja, umjesto tri. Ovo je trivijalan primjer, ali bitan je zbog vrlo značajnog problema: ako naši primjerci imaju mnoštvo atributa (a danas nije rijetkost da imamo primjerke kod kojih se broj atributa mjeri u tisućama), tada je algoritmima strojnog učenja tipično vrlo teško kvalitetno raditi s takvim primjercima (primjerice, ako razmatramo klasifikator primjeraka, postići dobru generalizaciju). Postupci smanjenja dimenzionalnosti mogu iz mnoštva atributa kojima su opisani primjerci identificirati (ili ponekad osmisliti nove) manji broj onih koji su relevantni za značenje primjerka.

1.3 Podržano učenje

Podržano učenje predstavlja dio strojnog učenja koji se bavi optimizacijom ponašanja. Za razliku od nadziranog i nenadziranog učenja, kod podržanog učenja razmatramo interakciju agenta i okoline u kojoj se agent nalazi. Agent na temelju informacija iz okoline obavlja akcije, i kao odgovor za svaku akciju od okoline dobiva nagradu ili kaznu. Zadaća podržanog učenja jest razviti "upravljački sustav" agenta, odnosno otkriti optimalnu strategiju njegovog ponašanja, tako da agent maksimizira nagrade koje dobiva "na duge staze".

Više o podržanom učenju čeka nas u zasebnoj, za to predviđenoj, temi.

2. Naivni Bayesov klasifikator

Naivni Bayesov klasifikator model je strojnog učenja koji omogućava klasifikaciju primjeraka. Kako se radi o modelu koji je utemeljen na teoriji vjerojatnosti, najprije ćemo se podsjetiti na Bayesov teorem s kojim smo se upoznali u prethodnoj cjelini, a potom pogledati primjenu iste na razvoj naivnog Bayesovog klasifikatora.

2.1 Bayesov teorem

U najjednostavnijem obliku, Bayesov teorem nam govori kako odrediti uvjetnu vjerojatnost realizacije događaja, i dana je u nastavku.

Theorem 2.1 — Bayesov teorem. Neka su A i B dva događaja. Vrijedi:

$$P(B|A) = \frac{P(A|B) \cdot P(B)}{P(A)}.$$

U slučaju da razmatramo jedan dokaz i niz mogućih hipoteza, uvjetna vjerojatnost svake od hipoteza pod utjecajem dokaza E može se izračunati kao:

$$P(H_i|E) = \frac{P(E|H_i) \cdot P(H_i)}{P(E)} = \frac{P(E|H_i) \cdot P(H_i)}{\sum_{j=1}^{m} P(H_j) \cdot P(E|H_j)}. \tag{2.1}$$

pri čemu desni izraz dobivamo ako pretpostavimo da hipoteze predstavljaju particiju (međusobno su disjunktne i njihova unija predstavlja čitav prostor događaja). U prethodnoj cjelini izveli smo i izraz za najopćenitiji slučaj: kada imamo m hipoteza i n dokaza. Tada koristimo izraz:

$$P(H_i|E_1,\ldots,E_n) = \frac{P(H_i)\prod_{k=1}^{n} P(E_k|H_i)}{P(E_1,\ldots,E_n)} = \frac{P(H_i)\prod_{k=1}^{n} P(E_k|H_i)}{\sum_{j=1}^{m} P(H_j)\prod_{k=1}^{n} P(E_k|H_j)}. \tag{2.2}$$

Za potrebe razmatranja koje slijedi, uvest ćemo još malo dodatne terminologije.

- $P(H_i)$ ćemo zvati apriorna vjerojatnost hipoteze H_i; to je naprosto vjerojatnost da hipoteza vrijedi prije no što dobijemo bilo kakve dokaze.
- $P(H_i|E)$ je aposteriorna vjerojatnost hipoteze H_i nakon što smo dobili dokaz E; drugim riječima, ako imamo dokaz E, kolika je vjerojatnost na vrijedi hipoteza H_i.
- $P(E|H_i)$ nazivamo izglednost dokaza E uz danu hipotezu H_i.

Da bismo došli do klasifikatora, preostalo je još samo postaviti posljednje pitanje: s obzirom na dokaze koje smo opazili, od koje bolesti boluje promatrani pacijent? Ovo je klasifikacijski problem u kojem pacijenta treba smjestiti u jedan od razreda.

Prvi način kako to možemo učiniti jest da prema izrazu (2.2) za sve hipoteze odredimo aposteriorne vjerojatnosti, te za pacijenta odaberemo onu hipotezu koja ima maksimalnu aposteriornu vjerojatnost; tu hipotezu nazivamo H_{MAP} i računamo prema izrazu:

$$H_{\text{MAP}} = \arg\max_{H_i} P(H_i|E_1,\ldots,E_n) = \arg\max_{H_i} \frac{P(H_i)\prod_{k=1}^{n} P(E_k|H_i)}{\sum_{j=1}^{m} P(H_j)\prod_{k=1}^{n} P(E_k|H_j)}.$$

Uočimo li da je nazivnik razlomka zapravo vjerojatnost $P(E_1,\ldots,E_n)$, i da je stoga isti za sve hipoteze čije vjerojatnosti računamo, H_{MAP} možemo odrediti i malo učinkovitije ne računajući nepotrebno nazivnik:

$$H_{\text{MAP}} = \arg\max_{H_i} P(H_i|E_1,\ldots,E_n) = \arg\max_{H_i} P(H_i)\prod_{k=1}^{n} P(E_k|H_i). \tag{2.3}$$

U posebnom slučaju da su sve hipoteze jednako vjerojatne, tj. $P(H_1) = P(H_2) = \cdots = P(H_m)$, iz izraza (2.3) možemo izbaciti i množenje s apriornim vjerojatnostima. U tom slučaju, dobivenu hipotezu zovemo hipotezom maksimalne izglednosti i računamo prema izrazu:

$$H_{\text{ML}} = \arg\max_{H_i} P(H_i|E_1,\ldots,E_n) = \arg\max_{H_i} \prod_{k=1}^{n} P(E_k|H_i). \tag{2.4}$$

2.1.1 Primjer klasifikatora na skupu *Dan za sport*

Jedan od čestih primjera na kojem se analizira izgradnja naivnog Bayesovog klasifikatora jest skup primjeraka *Dan za sport*. Ovaj skup primjeraka prikazan je tablicom u nastavku.

Redni broj	Vrijeme	Temperatura	Vlažnost	Vjetar	Igra
1.	sunčano	vruće	velika	slab	NE
2.	sunčano	vruće	velika	jak	NE
3.	oblačno	vruće	velika	slab	DA
4.	kišno	ugodno	velika	slab	DA
5.	kišno	hladno	normalna	slab	DA
6.	kišno	hladno	normalna	jak	NE
7.	oblačno	hladno	normalna	jak	DA
8.	sunčano	ugodno	velika	slab	NE
9.	sunčano	hladno	normalna	slab	DA
10.	kišno	ugodno	normalna	slab	DA
11.	sunčano	ugodno	normalna	jak	DA
12.	oblačno	ugodno	velika	jak	DA
13.	oblačno	vruće	normalna	slab	DA
14.	kišno	ugodno	velika	jak	NE

Primjerci su definirani kao uređene četvorke (vrijeme,temperatura,vlažnost,vjetar) koje klasificiramo u dva razreda: DA i NE. Pretpostavimo sada da imamo primjerak (kišno,vruće,velika,jak); bi li to bio dobar dan za baviti se sportom ili ne? Uvidom u prethodnu tablicu vidimo da taj primjerak nije sadržan u tablici, pa ne možemo pročitati u koji bismo razred primjerak trebali smjestiti. Stoga ćemo primijeniti izraz (2.3) kako bismo odredili koja je od hipoteza najizglednija: hipoteza H_{DA} ili hipoteza H_{NE}. Što su naši opaženi dokazi? Atribut *vrijeme* imao je vrijednost "kišno", atribut *temperatura* imao je vrijednost "vruće", atribut *vlažnost* imao je vrijednost "velika" i atribut *vjetar* imao je vrijednost "jak". Prilagodimo li izraz (2.3) ovom konkretnom slučaju, imamo:

$$H_{\text{MAP}} = \arg\max_{H_i} P(H_i|\text{vrijeme=kišno}, \text{temperatura=vruće}, \text{vlažnost=velika}, \text{vjetar=jak})$$

Stoga trebamo odrediti apriorne vjerojatnosti obje hipoteze. To radimo uvidom u skup podataka kojim raspolažemo. Skup ima 14 primjeraka pri čemu ih je 9 u razredu DA, a 5 u razredu NE. Ovo ćemo iskoristiti da bismo procijenili apriorne vjerojatnosti hipoteza:

$$P(H_{DA}) = \frac{9}{14},$$

$$P(H_{NE}) = \frac{5}{14}.$$

Trebat ćemo i 8 aposteriornih vjerojatnosti koje također možemo odrediti brojanjem po primjercima:

$$P(\text{vrijeme=kišno}|H_{DA}) = \frac{3}{9},$$

$$P(\text{temperatura=vruće}|H_{DA}) = \frac{2}{9},$$

$$P(\text{vlažnost=velika}|H_{DA}) = \frac{3}{9},$$

$$P(\text{vjetar=jak}|H_{DA}) = \frac{3}{9},$$

$$P(\text{vrijeme=kišno}|H_{NE}) = \frac{2}{5},$$

$$P(\text{temperatura=vruće}|H_{NE}) = \frac{2}{5},$$

$$P(\text{vlažnost=velika}|H_{NE}) = \frac{4}{5},$$

$$P(\text{vjetar=jak}|H_{NE}) = \frac{3}{5},$$

Primjerice, aposteriornu vjerojatnost $P(\text{vrijeme=kišno}|H_{DA})$ odredili smo kao omjer broja primjeraka koji su u razredu DA i imaju vrijeme=kišno i broja primjeraka koji su u razredu DA.

Za razred DA aposteriorna vjerojatnost proporcionalna je:

$$P(H_{DA})\cdot P(\text{vrijeme=kišno}|H_{DA})\cdot P(\text{temperatura=vruće}|H_{DA})\cdot P(\text{vlažnost=velika}|H_{DA})\cdot P(\text{vjetar=jak}|H_{DA})$$

što je:

$$\frac{9}{14}\cdot\frac{3}{9}\cdot\frac{2}{9}\cdot\frac{3}{9}\cdot\frac{3}{9} = \frac{1}{189} \approx 0.00529$$

Za razred NE aposteriorna vjerojatnost proporcionalna je:

$$P(H_{NE})\cdot P(\text{vrijeme=kišno}|H_{NE})\cdot P(\text{temperatura=vruće}|H_{NE})\cdot P(\text{vlažnost=velika}|H_{NEA})\cdot P(\text{vjetar=jak}|H_{NE})$$

što je:

$$\frac{5}{14}\cdot\frac{2}{5}\cdot\frac{2}{5}\cdot\frac{4}{5}\cdot\frac{3}{5}=\frac{24}{875}\approx 0.02743$$

Kako je $\max(0.00353, 0.02743) = 0.02743$, MAP-hipoteza je H_{NE}, čime primjerak razvrstavamo u razred NE.

Programska implementacija u memoriji ne bi pamtila čitav skup primjeraka jer isti može biti ogroman. Umjesto toga, u fazi *učenja* klasifikatora na temelju primjeraka izračunala bi i u prikladnim podatkovnim strukturama (primjerice, mapama) zapamtila apriorne vjerojatnosti svakog od razreda, te aposteriorne vjerojatnosti vrijednosti svakog od atributa po razredima. U fazi uporabe (iskorištavanja), na temelju predanog primjerka iz mapa bi samo dohvatila potrebne podatke, izmnožila ih i odabrala MAP-hipotezu.

Isprobajte

Postupak klasifikacije Naivnim Bayesovim klasifikatorom možete isprobati i sami. U narednom retku zadajte naredbu:

```
java -cp book-ml.jar ml.bayes.Classifier sport.txt
```

Kroz argument zadajete definiciju datoteke sa skupom primjeraka za učenje. `sport.txt` je ugrađena datoteka; dostupna je i `sport-en.txt` koja predstavlja isti skup samo na engleskom jeziku. Pokrenut će se program i ispisati nekoliko informacija, kao i napomenu da za izlazak zadate naredbu "exit", a za ispis učitanog skupa naredbu "dataset". Program će čekati u interaktivnoj ljusci da upisujete primjerke koje želite klasificirati. Primjerice, ako upišete:

```
(kišno,vruće,velika,jak)
```

dobit ćete ispis i klasifikaciju za primjer koji smo analizirali i u tekstu.

Želite li klasifikator isprobati na vlastitom primjeru, možete i sami pripremiti svoju datoteku: u prvi redak upišite nazive atributa odvojene tabom (posljednji će se smatrati ciljnim), a u preostale retke upišite specifikacije primjeraka (navodite samo vrijednosti atributa i također ih razdvajate tabom). Potom prilikom pokretanja programa zadajte stazu do Vaše datoteke. Ako program pokrenete primjerice s ugrađenom datotekom pa zadate naredbu `dataset-file`, dobit ćete sintaksno ispravan prikaz datoteke za učitani skup.

2.1.2 Zaglađivanje

Kako su skupovi primjeraka za učenje tipično ograničeni, događat će se da prilikom izračuna aposteriornih vjerojatnosti za neku vrijednost ciljnog atributa i zadanu vrijednost traženog atributa nemamo niti primjerak u skupu primjeraka za učenje, čime ta vjerojatnost postaje nula. Kako su vjerojatnosti naših hipoteza proporcionalne umnošku aposteriornih vjerojatnosti, ako je jedna aposteriorna vjerojatnost jednaka nula, čitava vjerojatnost hipoteze postaje nula. Ponekad bismo ovakvo ponašanje htjeli izbjeći, uz argumentaciju da to što u skupu primjeraka za učenje nismo imali niti jedan ovakav primjerak ne znači da je to u stvarnosti nemoguće.

Ilustrirajmo opisano na primjeru, kako bi bilo jasnije o čemu točno govorimo. I dalje smo na skupu *Dan za sport*. Pogledajmo aposteriorne vjerojatnosti za svaku moguću vrijednost atributa *Vrijeme* uz vrijednost ciljnog atributa *Igra*=NE, odnosno hipoteze H_{NE}.

$$P(\text{vrijeme=sunčano}|H_{NE})=\frac{3}{5}=0.6$$

$$P(\text{vrijeme=oblačno}|H_{NE})=\frac{0}{5}=0.0$$

$$P(\text{vrijeme=kišno}|H_{NE}) = \frac{2}{5} = 0.4$$

Iz ovoga jasno vidimo da će vjerojatnost bilo koje hipoteze koja zahtijeva da je vrijeme oblačno te dan nije prikladan za sport biti 0, neovisno o vrijednostima svih ostalih atributa (primjerice, temperature, vlažnosti odnosno vjetra), čime je hipoteza nemoguća.

Da bismo ovo izbjegli odnosno da bismo izgradili u određenom smislu "robusniji" klasifikator, aposteriorne vjerojatnosti ćemo redistribuirati postupkom *zaglađivanja* na način da malo smanjimo aposteriorne vjerojatnosti za koje imamo potporu u skupu primjeraka za učenje, te da na njihov račun podignemo aposteriorne vjerojatnosti koje su prethodno bile nula. Postupak zaglađivanja koji ćemo koristiti naziva se *Laplaceovo zaglađivanje* (koristi se i naziv *aditivno zaglađivanje*; engl. *Laplace smoothing, Additive smoothing*).

Opišimo najprije ideju, a potom ćemo dati i izraz prema kojem ćemo računati zaglađene vjerojatnosti. Pretpostavimo da razmatramo aposteriornu vjerojatnost $P(\text{vrijeme=oblačno}|H_{NE})$: prethodno smo već pregledali skup primjeraka za učenje i utvrdili da u njemu nema niti jedan primjerak koji je imao oblačno vrijeme, a da nije bio pogodan za sport, i to je rezultiralo aposteriornom vjerojatnosću iznosa nula. Stoga ćemo zamisliti da zapravo radimo s novim skupom primjeraka za učenje u koji smo ubacili jedan izmišljeni primjerak koji je imao baš tu kombinaciju vrijednosti. Možemo li to nekako opravdati? Na smislen način teško. Kako bismo minimizirali "štetu" koju ovime radimo, još ćemo malo modificirati skup. Trenutno razmatramo aposteriorne vjerojatnosti različitih vrijednosti atributa *Vrijeme* uz H_{NE}. Stoga ćemo skup modificirati tako da u njega ne ubacimo samo jedan izmišljeni primjerak: baš onaj koji je imao oblačno vrijeme, već po jedan za svaku moguću vrijednost atributa *Vrijeme*: jedan koji je imao oblačno vrijeme uz H_{NE}, jedan koji je imao sunčano vrijeme uz H_{NE} te jedan koji je imao kišno vrijeme uz H_{NE}. Pri ovome treba pripaziti da sve aposteriorne vjerojatnosti onda računamo na jednak način. U našem konkretnom slučaju, atribut vrijeme poprima vrijednosti iz skupa kardinaliteta 3 (skup je {sunčano, oblačno, kišno}), pa ćemo sve aposteriorne vjerojatnosti uz atribut *Vrijeme* računati tako da u brojnik dodamo +1 (za taj jedan izmišljeni primjerak koji je bio baš zadane vrijednosti), a u nazivnik dodamo +3 (jer smo efektivno ubacili 3 izmišljena primjerka - po jedan za svaku vrijednost atributa *Vrijeme*). Ovakava izvedba zaglađivanja vrsta je Laplaceovog zaglađivanja i još se naziva zaglađivanje dodavanjem jednog primjerka (engl. *add-one smoothing*). Izračunamo li aposteriorne vjerojatnosti na ovaj način, dobit ćemo:

$$P(\text{vrijeme=sunčano}|H_{NE}) = \frac{3+1}{5+3} = \frac{4}{8} = 0.5$$

$$P(\text{vrijeme=oblačno}|H_{NE}) = \frac{0+1}{5+3} = \frac{1}{8} = 0.125$$

$$P(\text{vrijeme=kišno}|H_{NE}) = \frac{2+1}{5+3} = \frac{3}{8} = 0.375$$

Općenito, Laplaceovo zaglađivanje je zaglađivanje uz parametar α čije je značenje upravo broj izmišljenih primjeraka kod kojih je zadani atribut postavljen baš na vrijednost koju gledamo u aposteriornoj vjerojatnosti. U tom slučaju, ukupan broj izmišljenih primjeraka koje ubacujemo u skup jednak je α puta kardinalitet skupa iz kojeg atribut poprima vrijednosti, čime je konačan izraz po kojem se računa Laplaceovo zaglađivanje:

$$P(x_j = w|H_v) = \frac{|D_{x_j=w \wedge y=v}| + \alpha}{|D_{y=v}| + \alpha \cdot |V(x_j)|}$$

pri čemu je značenje oznaka sljedeće:

- $P(x_j = w|H_v)$ je aposteriorna vjerojatnost da vrijedi hipoteza H_v (odnosno da ciljni atribut y ima vrijednost v) ako je vrijednost atributa x_j jednaka w,
- $D_{x_j=w \wedge y=v}$ je podskup primjeraka za učenje u koji su uključeni samo primjerci kod kojih je vrijednost atributa x_j jednaka w i istovremeno je vrijednost ciljnog atributa y jednaka v, a $|\cdot|$ je kardinalitet tog skupa,
- $D_{y=v}$ je podskup primjeraka za učenje u koji su uključeni samo primjerci kod kojih je vrijednost ciljnog atributa y jednaka v, a $|\cdot|$ je kardinalitet tog skupa,
- $V(x_j)$ je skup vrijednosti koje može poprimiti atribut x_j, a $|\cdot|$ je kardinalitet tog skupa.

U primjeru koji smo razmatrali, x_j je bio *Vrijeme*, $V(x_j)$={sunčano, oblačno, kišno}, v je bio *NE*, a w je išao po elementima iz {sunčano, oblačno, kišno}.

Ako je $\alpha = 0$, nema zaglađivanja. Ako je $\alpha = 1$ imamo *zaglađivanje dodavanjem jednog primjerka* (pazi: u brojniku, uz modifikaciju koja time slijedi za nazivnik; engl. *add-one smoothing*).

Primijetimo također da će efekt zaglađivanja biti to manji što je veličina skupa primjeraka za učenje (a time i raznih podskupova koje dobijemo filtriranjem) veća.

Isprobajte

Postupak klasifikacije Naivnim Bayesovim klasifikatorom uz mogućnost Laplaceovog zaglađivanja možete isprobati i sami. U naredbenom retku zadajte naredbu:

```
java -cp book-ml.jar ml.bayes.Classifier sport.txt
```

Kroz argument zadajete definiciju datoteke sa skupom primjeraka za učenje. `sport.txt` je ugrađena datoteka. Pokrenut će se program i ispisati nekoliko informacija, kao i napomena da za izlazak zadate "exit". Program će čekati u interaktivnoj ljusci da upisujete primjerke koje želite klasificirati. Primjerice, ako upišete:

```
(kišno,vruće,velika,jak)
```

dobit ćete ispis i klasifikaciju za primjer koji smo analizirali i u tekstu i taj je bez zaglađivanja. Sada u programu možete zadati naredbu:

`laplace-smoothing=`α (na mjestu α upišite željeni broj, primjerice, 0, 1, 2, ...)

i nakon njezinog izvođenja u novom retku ponovno zadajte primjerak koji želite klasificirati.

Želite li klasifikator isprobati na vlastitom primjeru, možete i sami pripremiti svoju datoteku: u prvi redak upišite nazive atributa odvojene tabom (posljednji će se smatrati ciljnim), a u preostale retke upišite specifikacije primjeraka (navodite samo vrijednosti atributa i također ih razdvajate tabom). Potom prilikom pokretanja programa zadajte stazu do Vaše datoteke.

3. Stabla odluke

Stabla odluke su formalizam koji omogućava rješavanje klasifikacijskih te aproksimacijskih zadataka temeljeći se na slijedu ispitivanja vrijednosti atributa primjerka. U okviru ovog poglavlja razmotrit ćemo uporabu stabla odluke za klasifikacijske svrhe te njihovu izgradnju uporabom algoritma ID3.

Pogledajmo najprije skup primjeraka koji će nam biti na raspolaganju. Radi se o primjercima koji opisuju oštećenje rotacijskog elementa određenog stroja te na temelju karakteristika oštećenja definiraju je li popravak elementa hitan ili ne. Skup primjeraka koji nam je na raspolaganju prikazan je u tablici 3.1.

Tablica 3.1: Skup primjeraka: *Hitnost popravka rotacijskog elementa*.

Redni broj	Oštećenje	Položaj	Boja	Hitno
1.	malo	dalji	tamno	NE
2.	srednje	bliži	svijetlo	DA
3.	malo	dalji	svijetlo	NE
4.	veliko	bliži	svijetlo	DA
5.	veliko	dalji	tamno	DA
6.	veliko	dalji	svijetlo	DA
7.	srednje	dalji	svijetlo	NE
8.	veliko	bliži	tamno	DA
9.	srednje	dalji	tamno	NE
10.	srednje	bliži	tamno	DA
11.	malo	bliži	svijetlo	NE
12.	malo	bliži	tamno	DA

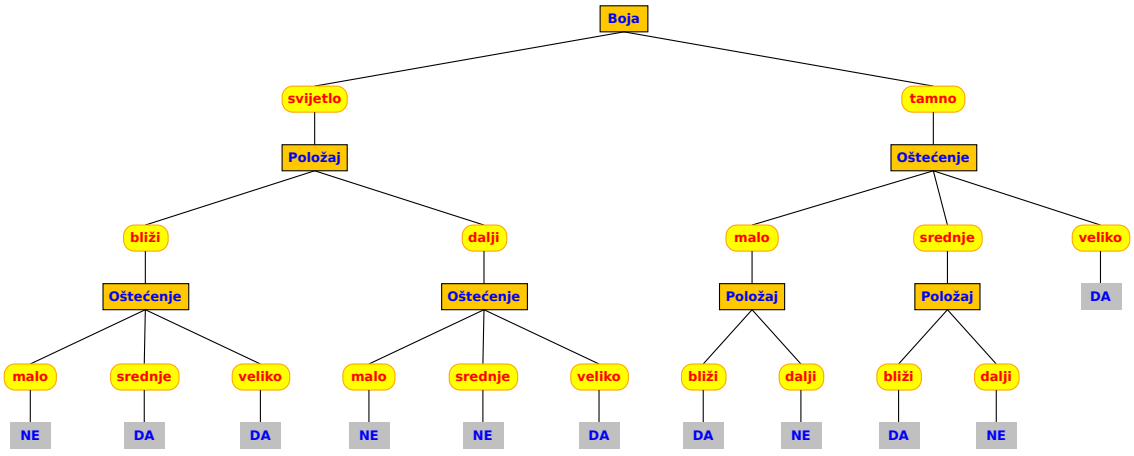
Svaki primjerak opisan je s tri atributa: *Oštećenje*, *Položaj* i *Boja*. Ove atribute zvat ćemo još i značajkama (engl. *features*) primjerka. Primjerke razvrstavamo u dva razreda: one koje je

potrebno hitno popraviti te one kod kojih popravak nije hitan. Posljednji stupac tablice 3.1 sadrži ovu informaciju: ako je vrijednost DA, rotacijski element potrebno je hitno popraviti. Ako je vrijednost NE, popravak nije hitan. Evo i detaljnijeg opisa svakog od atributa.

- *Oštećenje* - specificira kolika je veličina uočenog oštećenja; vrijednosti koje atribut poprima su *malo*, *srednje* te *veliko*.
- *Položaj* - specificira gdje se nalazi oštećenje s obzirom na centar rotacije elementa; vrijednosti koje atribut poprima su *bliži* i *dalji*.
- *Boja* - specificira nijansu boje uočenog oštećenja; vrijednosti koje atribut poprima su *svijetlo* i *tamno*.

Atribut *Hitno* koji je prikazan u posljednjem stupcu tablice još nazivamo i ciljnim atributom. Njegove su vrijednosti u ovom primjeru *DA* i *NE*, a razmatrat ćemo klasifikatorske sustave koji će promatranom primjerku na temelju vrijednosti atributa *Oštećenje*, *Položaj* i *Boja* pokušati dodijeliti ispravnu vrijednost ciljnog atributa.

Pogledajmo sada jedno moguće stablo odluke. Slika 3.1 prikazuje stablo odluke koje sve primjerke iz skupa primjeraka iz tablice 3.1 klasificira korektno.



Slika 3.1: Primjer stabla odluke za skup *Hitnost popravka rotacijskog elementa*.

Korijen stabla je čvor u kojem se ispituje vrijednost atributa *Boja*. Čvor ima dvije grane jer atribut boja može poprimiti dvije vrijednosti. Lijevu granu potrebno je slijediti ako primjerak koji ispitujemo ima *svijetlo* kao vrijednost atributa *Boja*, a desnu ako primjerak koji ispitujemo ima *tamno* kao vrijednost atributa *Boja*. U oba slučaja dolazimo do podstabla koje obrađujemo na analogan način. Iznimka su listovi stabla u kojima se ne ispituje vrijednost nekog od atributa već donosi konačna odluka o razredu kojem primjerak pripada. Ti su čvorovi na slici prikazani kao sivi pravokutnici u koje je upisana vrijednost ciljnog atributa.

Kako bismo uporabom prikazanog stabla odluke zaključili koja je vrijednost ciljnog atributa za primjerak (veliko,bliži,tamno)? Korijen stabla sadrži čvor koji ispituje atribut *Boja*; stoga gledamo koja je boja u našem primjerku koji klasificiramo (*Boja*=tamno), pa u stablu slijedimo desnu granu. Dolazimo do čvora koji ispituje vrijednost atributa *Oštećenje*; stoga gledamo koju vrijednost ima taj atribut u našem primjerku koji klasificiramo (*Oštećenje*=veliko), pa u stablu ponovno slijedimo desnu granu. Time smo došli do lista koji primjerku pridjeljuje vrijednost DA kao vrijednost ciljnog atributa, odnosno primjerak razvrstava u razred primjeraka koje je potrebno hitno popraviti.

Primijetite da je utvrđeni razred u skladu s onime što je za primjerak (veliko,bliži,tamno) definirano u tablici 3.1. Primijetite također da za donošenje ispravne odluke nismo morali razmotriti vrijednosti svih atributa primjerka koji klasificiramo.

Prilikom izgradnje stabala odluke htjet ćemo da izgrađeno stablo ima određene karakteristike.

Neiznenađujuće, za početak ćemo htjeti da isto u većini slučajeva ispravno radi svoj posao na skupu primjeraka na temelju kojeg je izgrađeno, odnosno da primjercima pridjeljuje vrijednosti ciljnog atributa kako je specificirano u primjercima za učenje. Ako se pitate zašto ne u svim slučajevima, odgovor leži u činjenici da želimo stabla koja dobro generaliziraju, pa ako za neke primjerke utvrdimo da su *stršeće vrijednosti* (engl. *outliers*), neće nas smetati što će konstruirano stablo njima pridijeljivati drugačiju vrijednost ciljnog atributa od one koja je zapisana u skupu primjeraka za učenje.

Od svih stabala koja su usklađena s prethodnim zahtjevom, preferirat ćemo ona koja su manja, odnosno koja imaju manji broj čvorova.

Klasifikatorska stabla možemo graditi na različite načine. U ovom poglavlju, upoznat ćemo se algoritmom ID3 (engl. *Iterative Dichotomiser 3*) kojeg je osmislio Ross Quinlan, a kako se ovaj postupak temelji na pojmovima entropije i informacijske dobiti, najprije ćemo se upoznati s tim pojmovima.

Razmotrimo četiri skupa primjeraka za učenje, čije su karakteristike specificirane u sljedećoj tablici.

Ime skupa	Ukupan broj primjeraka	Broj primjeraka u razredu DA	Broj primjeraka u razredu NE
$\mathscr{S}_1$	10	10	0
$\mathscr{S}_2$	10	3	7
$\mathscr{S}_3$	10	5	5
$\mathscr{S}_4$	10	0	10

Entropija je mjera neuređenosti skupa. Intuitivno, mogli bismo razmišljati ovako: kada bismo iz skupa nasumice izvlačili primjerke i pokušavali pogoditi (bez uvida u atribute izvučenog primjerka) kojem će razredu pripadati atribut, što je mjera neuređenosti skupa veća, više bismo griješili pri pogađanju.

Pogledajmo skup $\mathscr{S}_1$: on sadrži 10 elemenata, i svi pripadaju razredu DA. Uz pretpostavku da tu informaciju znamo, svaki slučajno izvučeni primjerak savršeno bismo klasificirali uvijek pridjeljujući primjerke u razred DA. Ovaj skup ima minimalnu neuređenost. Analogno vrijedi i za skup $\mathscr{S}_4$ kod kojeg su svi primjerci smješteni u razred NE; i ovaj skup ima minimalnu neuređenost.

Kod razreda $\mathscr{S}_2$ i $\mathscr{S}_3$ nećemo više moći uvijek korektno pogađati ispravan razred izvučenog primjerka. Pri tome bismo s razredom $\mathscr{S}_2$ mogli proći bolje no s razredom $\mathscr{S}_3$. U razredu $\mathscr{S}_2$ 70% primjeraka pripada razredu NE; stoga, ako svakom izvučenom primjerku iz razreda $\mathscr{S}_2$ uvijek pridijelimo oznaku NE, u prosjeku ćemo griješiti u 30% slučajeva. Za razliku od toga, u razredu $\mathscr{S}_3$ po 50% primjeraka pripada razredima DA odnosno NE; stoga ako ćemo uvijek primjerke svrstavati u razred DA (ili NE), griješit ćemo čak u 50% slučajeva.

Uzevši ova zapažanja u obzir, mogli bismo reći da je mjera neuređenosti skupa $\mathscr{S}_3$ veća od mjere neuređenosti skupa $\mathscr{S}_2$ koja je pak veća od mjera neuređenosti skupova $\mathscr{S}_1$ i $\mathscr{S}_4$, koje su minimalne moguće.

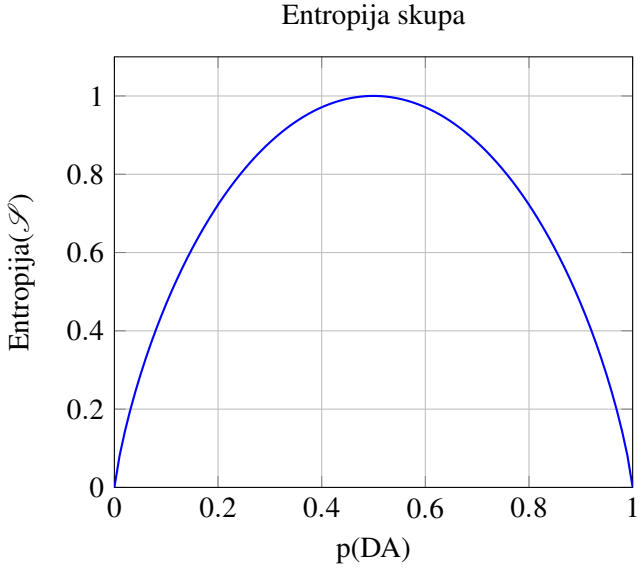
Entropiju skupa $\mathscr{S}$ definiramo s obzirom na vjerojatnosti da slučajno izvučeni primjerak pripada pojedinom razredu. Mjera je definirana kao negativna suma vjerojatnosti da primjerak pripada određenom razredu pomnoženoj dualnim logaritmom iste. U našem konkretnom slučaju imamo dva razreda: DA i NE. Stoga će entropija biti određena izrazom:

$$\text{Entropija}(\mathscr{S}) = -p(\text{DA}) \cdot \log_2 (p(\text{DA})) - p(\text{NE}) \cdot \log_2 (p(\text{NE}))$$

Ako je vjerojatnost nekog razreda 0, odnosno trebamo izračunati $0 \cdot log_2 0$, uzimat ćemo da je to jednako 0. Za skupove $\mathscr{S}_1$ do $\mathscr{S}_4$ entropije izračunate prema prethodnom izrazu prikazane su u nastavku.

Ime skupa	$p(DA)$	$p(NE)$	Entropija($\mathscr{S}_i$)
$\mathscr{S}_1$	$\frac{10}{10} = 1$	$\frac{0}{10} = 0$	0
$\mathscr{S}_2$	$\frac{3}{10} = 0.3$	$\frac{7}{10} = 0.7$	0.8813
$\mathscr{S}_3$	$\frac{5}{10} = 0.5$	$\frac{5}{10} = 0.5$	1
$\mathscr{S}_4$	$\frac{0}{10} = 0$	$\frac{10}{10} = 1$	0

U skladu s našim očekivanjem, vidimo da je entropija minimalna za skupove $\mathscr{S}_1$ i $\mathscr{S}_4$ i taj minimum je 0, entropija je veća za skup $\mathscr{S}_2$, a najveća za skup $\mathscr{S}_3$. Kako se iznos entropije mijenja ovisno o vjerojatnosti da primjerak pripada razredu DA prikazano je na slici 3.2, gdje vidimo da je maksimalni iznos koji entropija poprima jednak 1 (ovo vrijedi za slučaj binarne klasifikacije).



Slika 3.2: Entropija skupa u kojem primjerci pripadaju jednom od dva razreda: DA i NE, s obzirom na vjerojatnost da primjerak pripada razredu DA. Uočimo da je u ovom slučaju $p(NE) = 1 - p(DA)$.

Općenito, ako primjerci mogu pripadati jednom od K razreda, entropija skupa definira se izrazom u nastavku koji je analogan prethodnoj definiciji samo poopćen na više razreda:

$$\text{Entropija}(\mathscr{S}) = -\sum_{c \in \mathscr{C}} p(\text{c}) \cdot \log_2 (p(\text{c})) \tag{3.1}$$

gdje smo s $\mathscr{C}$ označili skup svih razreda, pa prikazana suma ide po razredima. $p(c)$ je vjerojatnost da primjerak pripada razredu c.

Ideja algoritma ID3 jest u svakom koraku razmotriti kolika je korist ako se razmatrani skup podataka podijeli po svakom od atributa. Zatim pohlepno napravi podjelu skupa primjeraka po tom atributu, stvori čvor s oznakom odabranog atributa te rekurzivno gradi podstabla nad napravljenim podskupovima. Mjera kvalitete podjele koju koristi ID3 je *informacijska dobit*: informacijska dobit jednaka je entropiji početnog skupa umanjenoj za entropije napravljenih podskupova skalirane omjerom veličine podskupa i početnog skupa. Intuitivno, ona nam govori koliko smo uspjeli smanjiti neuređenost podjelom skupa u manje podskupove. Možda zvuči komplicirano, no idemo pogledati na konkretnom primjeru skupa prikazanog u tablici 3.1.

U početnom skupu imamo 12 primjeraka. 7 od njih pripada razredu DA pa je $p(DA) = 7/12 = 0.5833$, a 5 ih pripada razredu NE pa je $p(NE) = 5/12 = 0.4167$. Entropija ovog skupa je dakle $-0.5833 * \log_2(0.5833) - 0.4167 * log_2(0.4167) = 0.9799$.

Primjerke ovog skupa možemo podijeliti prema vrijednostima triju atributa. Razmotrimo najprije podjelu po atributu *Oštećenje*. Ovaj atribut može poprimiti 3 različite vrijednosti, što znači da ćemo napraviti 3 podskupa.

Pogledajmo podskup koji čine svi primjerci koji imaju *Oštećenje*=malo. Taj podskup čine primjerci 1, 3, 11 i 12:

Redni broj	Oštećenje	Položaj	Boja	Hitno
1.	malo	dalji	tamno	NE
3.	malo	dalji	svijetlo	NE
11.	malo	bliži	svijetlo	NE
12.	malo	bliži	tamno	DA

Skup ima 4 primjerka. 1 pripada razredu DA pa je $p(DA) = 1/4 = 0.25$, a 3 pripadaju razredu NE pa je $p(NE) = 3/4 = 0.75$. Stoga ovaj skup ima entropiju:

$$Entropija(\text{Oštećenje=malo}) = -0.25 * log_2(0.25) - 0.75 * log_2(0.75) = 0.8113.$$

Prilikom izračuna informacijske dobiti entropiji početnog skupa oduzet ćemo ovu vrijednost skaliranu omjerom veličine ovog skupa (4) i početnog skupa (12), odnosno vrijednost 0.8113*4/12=0.2704.

Pogledajmo podskup koji čine svi primjerci koji imaju *Oštećenje*=srednje. Taj podskup čine primjerci 2, 7, 9 i 10:

Redni broj	Oštećenje	Položaj	Boja	Hitno
2.	srednje	bliži	svijetlo	DA
7.	srednje	dalji	svijetlo	NE
9.	srednje	dalji	tamno	NE
10.	srednje	bliži	tamno	DA

Skup ima 4 primjerka. 2 pripadaju razredu DA pa je $p(DA) = 2/4 = 0.5$, a 2 razredu NE pa je $p(NE) = 2/4 = 0.5$; primijetite da je skup maksimalno neuređen. Ovaj skup ima entropiju:

$$Entropija(\text{Oštećenje=srednje}) = -0.5 * log_2(0.5) - 0.5 * log_2(0.5) = 1.$$

Prilikom izračuna informacijske dobiti entropiji početnog skupa oduzet ćemo ovu vrijednost skaliranu omjerom veličine ovog skupa (4) i početnog skupa (12), odnosno vrijednost $1 * 4/12 = 0.3333$.

Konačno, pogledajmo podskup koji čine svi primjerci koji imaju *Oštećenje*=veliko. Taj podskup čine primjerci 4, 5, 6 i 8:

Redni broj	Oštećenje	Položaj	Boja	Hitno
4.	veliko	bliži	svijetlo	DA
5.	veliko	dalji	tamno	DA
6.	veliko	dalji	svijetlo	DA
8.	veliko	bliži	tamno	DA

Skup ima 4 primjerka i sva četiri pripadaju razredu DA. Stoga je $p(DA) = 4/4 = 1$, a $p(NE) = 0/4 = 0$; primijetite da je skup maksimalno uređen; ima minimalnu neuređenost. Skup ima entropiju:

$$Entropija(\text{Oštećenje=veliko}) = -1 * log_2(1) - 0 * log_2(0) = 0.$$

Prilikom izračuna informacijske dobiti entropiji početnog skupa oduzet ćemo ovu vrijednost skaliranu omjerom veličine ovog skupa (4) i početnog skupa (12), odnosno vrijednost 0*4/12=0.

Sada smo spremni izračunati informacijsku dobit podjele početnog skupa u tri podskupa prema vrijednostima atributa *Oštećenje*. Prisjetimo se, entropija početnog skupa veličine 12 bila je 0.9799. Vrijednosti koje smo izračunali za svaki od podskupova ponovljene su u tablici u nastavku.

Vrijednost atributa	Veličina podskupa	Entropija podskupa
malo	4	0.8113
srednje	4	1
veliko	4	0

Informacijska dobit ove podjele stoga je:

$$\begin{aligned} \textit{Informacijska Dobit}(\text{Oštećenje}) = \textit{Entropija}(\text{Početni skup}) \\ - \frac{4}{12} \cdot \textit{Entropija}(\text{Oštećenje=malo}) \\ - \frac{4}{12} \cdot \textit{Entropija}(\text{Oštećenje=srednje}) \\ - \frac{4}{12} \cdot \textit{Entropija}(\text{Oštećenje=veliko}) \end{aligned}$$

čime nakon uvrštavanja dobivamo:

$$\begin{aligned} \textit{Informacijska Dobit}(\text{Oštećenje}) &= 0.9799 - \frac{4}{12} \cdot 0.8113 - \frac{4}{12} \cdot 1 - \frac{4}{12} \cdot 0 \\ &= 0.9799 - 0.2704 - 0.3333 - 0 \\ &= 0.3761. \end{aligned}$$

Da bismo odlučili hoćemo li početni skup primjeraka doista razdijeliti prema vrijednostima atributa *Oštećenje*, trebamo pogledati i kolike su informacijske dobiti ako skup razdijelimo prema atributu *Položaj* odnosno prema atributu Boja. Proanalizirajmo najprije podjelu prema atributu *Položaj*.

Pogledajmo podskup koji čine svi primjerci koji imaju *Položaj*=bliži. Taj podskup čine primjerci 2, 4, 8, 10, 11 i 12:

Redni broj	Oštećenje	Položaj	Boja	Hitno
2.	srednje	bliži	svijetlo	DA
4.	veliko	bliži	svijetlo	DA
8.	veliko	bliži	tamno	DA
10.	srednje	bliži	tamno	DA
11.	malo	bliži	svijetlo	NE
12.	malo	bliži	tamno	DA

Skup ima 6 primjerka od kojih pet pripada razredu DA, a jedan razredu NE. Stoga je $p(DA) = 5/6 = 0.8333$, a $p(NE) = 1/6 = 0.1667$. Skup ima entropiju:

$$\textit{Entropija}(\text{Položaj=bliži}) = -0.8333 * log_2(0.8333) - 0.1667 * log_2(0.1667) = 0.65.$$

Prilikom izračuna informacijske dobiti entropiji početnog skupa oduzet ćemo ovu vrijednost skaliranu omjerom veličine ovog skupa (6) i početnog skupa (12), odnosno vrijednost 0.65*6/12=0.325.

Pogledajmo podskup koji čine svi primjerci koji imaju *Položaj*=dalji. Taj podskup čine primjerci 1, 3, 5, 6, 7 i 9:

Redni broj	Oštećenje	Položaj	Boja	Hitno
1.	malo	dalji	tamno	NE
3.	malo	dalji	svijetlo	NE
5.	veliko	dalji	tamno	DA
6.	veliko	dalji	svijetlo	DA
7.	srednje	dalji	svijetlo	NE
9.	srednje	dalji	tamno	NE

Skup ima 6 primjerka od kojih dva pripada razredu DA, a četiri razredu NE. Stoga je $p(DA) = 2/6 = 0.3333$, a $p(NE) = 4/6 = 0.6667$. Skup ima entropiju:

$$Entropija(\text{Položaj=dalji}) = -0.3333 * log_2(0.3333) - 0.6667 * log_2(0.6667) = 0.9183.$$

Prilikom izračuna informacijske dobiti entropiji početnog skupa oduzet ćemo ovu vrijednost skaliranu omjerom veličine ovog skupa (6) i početnog skupa (12), odnosno vrijednost 0.9183*6/12=0.4591.

Informacijska dobit ove podjele stoga je:

$$\begin{aligned} \textit{Informacijska Dobit}(\text{Položaj}) = \textit{Entropija}(&\text{Početni skup}) \\ &- \frac{6}{12} \cdot \textit{Entropija}(\text{Položaj=bliži}) \\ &- \frac{6}{12} \cdot \textit{Entropija}(\text{Položaj=dalji}) \end{aligned}$$

čime nakon uvrštavanja dobivamo:

$$\begin{aligned} \textit{Informacijska Dobit}(\text{Položaj}) &= 0.9799 - \frac{6}{12} \cdot 0.65 - \frac{6}{12} \cdot 0.9183 \\ &= 0.9799 - 0.325 - 0.4591 \\ &= 0.1957. \end{aligned}$$

Konačno, razmotrimo i podjeli prema atributu *Boja*. Pogledajmo podskup koji čine svi primjerci koji imaju *Boja*=svijetlo. Taj podskup čine primjerci 2, 3, 4, 6, 7 i 11:

Redni broj	Oštećenje	Položaj	Boja	Hitno
2.	srednje	bliži	svijetlo	DA
3.	malo	dalji	svijetlo	NE
4.	veliko	bliži	svijetlo	DA
6.	veliko	dalji	svijetlo	DA
7.	srednje	dalji	svijetlo	NE
11.	malo	bliži	svijetlo	NE

Skup ima 6 primjerka od kojih tri pripadaju razredu DA i tri razredu NE. Stoga je $p(DA) = 3/6 = 0.5$, a $p(NE) = 3/6 = 0.5$. Skup ima entropiju:

$$Entropija(\text{Boja=svijetlo}) = -0.5 * log_2(0.5) - 0.5 * log_2(0.5) = 1.$$

Prilikom izračuna informacijske dobiti entropiji početnog skupa oduzet ćemo ovu vrijednost skaliranu omjerom veličine ovog skupa (6) i početnog skupa (12), odnosno vrijednost 1*6/12=0.5.

Pogledajmo podskup koji čine svi primjerci koji imaju *Boja*=tamno. Taj podskup čine primjerci 1, 5, 8, 9, 10, 12:

Redni broj	Oštećenje	Položaj	Boja	Hitno
1.	malo	dalji	tamno	NE
5.	veliko	dalji	tamno	DA
8.	veliko	bliži	tamno	DA
9.	srednje	dalji	tamno	NE
10.	srednje	bliži	tamno	DA
12.	malo	bliži	tamno	DA

Skup ima 6 primjerka od kojih četiri pripadaju razredu DA i dva razredu NE. Stoga je $p(DA) = 4/6 = 0.6667$, a $p(NE) = 2/6 = 0.3333$. Skup ima entropiju:

$$Entropija(\text{Boja=svijetlo}) = -0.6667 * log_2(0.6667) - 0.3333 * log_2(0.3333) = 0.9183.$$

Prilikom izračuna informacijske dobiti entropiji početnog skupa oduzet ćemo ovu vrijednost skaliranu omjerom veličine ovog skupa (6) i početnog skupa (12), odnosno vrijednost 0.9183*6/12=0.4591.

Informacijska dobit ove podjele stoga je:

$$\begin{aligned} \textit{Informacijska Dobit}(\text{Boja}) = \; & \textit{Entropija}(\text{Početni skup}) \\ & - \frac{6}{12} \cdot \textit{Entropija}(\text{Boja=svijetlo}) \\ & - \frac{6}{12} \cdot \textit{Entropija}(\text{Boja=tamno}) \end{aligned}$$

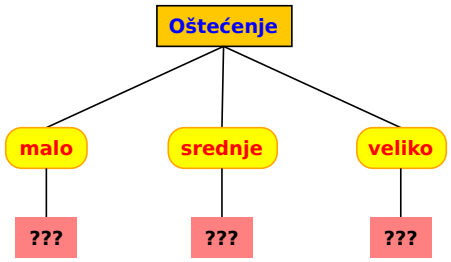
čime nakon uvrštavanja dobivamo:

$$\begin{aligned} \textit{Informacijska Dobit}(\text{Boja}) &= 0.9799 - \frac{6}{12} \cdot 1 - \frac{6}{12} \cdot 0.9183 \\ &= 0.9799 - 0.5 - 0.4591 \\ &= 0.0207. \end{aligned}$$

Koji ćemo od atributa uzeti za podjelu u korijenskom čvoru? Tablica u nastavku navodi utvrđene informacijske dobiti.

Podjela prema atributu	Informacijska dobit
Oštećenje	0.3761
Položaj	0.1957
Boja	0.0207

Najveća informacijska dobit (odnosno maksimalno smanjenje neuređenosti) postiže se za podjelu prema atributu *Oštećenje*. Stoga ćemo kao korijenski čvor stvoriti čvor koji analizira atribut *Oštećenje* i ima tri grane: malo, srednje i veliko; svaka od te tri grane analizirat će odgovarajući podskup primjeraka koji smo pripremili prilikom analize atributa *Oštećenje*. Djelomično stablo koje smo u ovom trenutku izgradili prikazano je u nastavku.



U lijevom čvoru ponavljamo postupak ID3, nad skupom primjeraka:

Redni broj	Oštećenje	Položaj	Boja	Hitno
1.	malo	dalji	tamno	NE
3.	malo	dalji	svijetlo	NE
11.	malo	bliži	svijetlo	NE
12.	malo	bliži	tamno	DA

To su svi primjerci koji imaju "malo" kao vrijednost atributa *Oštećenje*. Broj primjeraka je 4; jedan pripada razredu DA, a tri pripadaju razredu NE. Stoga je $p(DA) = 1/4 = 0.25$, $p(NE) = 3/4 = 0.75$. Entropija ovog podskupa (koji za ovu granu predstavlja novi "početni skup") stoga je $-0.25 \cdot \log_2(0.25) - 0.75 \cdot log_2(0.75) = 0.8113$.

U ovoj grani trebamo razmotriti podjele ovog skupa samo prema atributima *Položaj* i *Boja*; vrijednost atributa *Oštećenje* svim je primjercima fiksirana u roditeljskom čvoru pa se ovi primjerci po tom atributu ne razlikuju.

Razmotrimo podjelu prema atributu *Položaj*. Dobivamo dva podskupa; za *Položaj*=bliži imamo:

Redni broj	Oštećenje	Položaj	Boja	Hitno
11.	malo	bliži	svijetlo	NE
12.	malo	bliži	tamno	DA

Veličina ovog podskupa je 2, a entropija 1. Za *Položaj*=dalji imamo:

Redni broj	Oštećenje	Položaj	Boja	Hitno
1.	malo	dalji	tamno	NE
3.	malo	dalji	svijetlo	NE

Veličina ovog podskupa je 2, a entropija 0. Informacijska dobit podjele prema atributu *Položaj* stoga je:

$$\begin{aligned} \textit{Informacijska Dobit}(\text{Položaj}) &= \textit{Entropija}(\text{Početni skup}) \\ &\quad - \frac{2}{4} \cdot \textit{Entropija}(\text{Položaj=bliži}) \\ &\quad - \frac{2}{4} \cdot \textit{Entropija}(\text{Položaj=dalji}) \\ &= 0.8113 - \frac{2}{4} \cdot 1 - \frac{2}{4} \cdot 0 \\ &= 0.8113 - 0.5 - 0 \\ &= 0.3113. \end{aligned}$$

Razmotrimo podjelu prema atributu *Boja*. Dobivamo dva podskupa; za *Boja*=svijetlo imamo:

Redni broj	Oštećenje	Položaj	Boja	Hitno
3.	malo	dalji	svijetlo	NE
11.	malo	bliži	svijetlo	NE

Veličina ovog podskupa je 2, a entropija 0. Za *Boja*=tamno imamo:

Redni broj	Oštećenje	Položaj	Boja	Hitno
1.	malo	dalji	tamno	NE
12.	malo	bliži	tamno	DA

Veličina ovog podskupa je 2, a entropija 1. Informacijska dobit podjele prema atributu *Položaj*

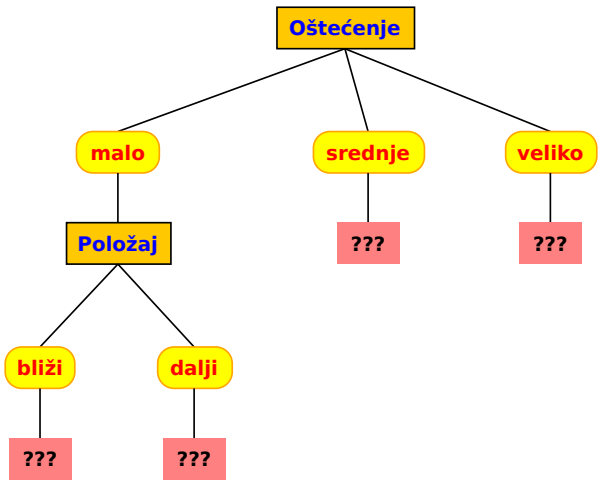
stoga je:

$$\begin{aligned} \textit{Informacijska Dobit}(\text{Boja}) &= \textit{Entropija}(\text{Početni skup}) \\ &\quad - \frac{2}{4} \cdot \textit{Entropija}(\text{Boja=svijetlo}) \\ &\quad - \frac{2}{4} \cdot \textit{Entropija}(\text{Boja=tamno}) \\ &= 0.8113 - \frac{2}{4} \cdot 0 - \frac{2}{4} \cdot 1 \\ &= 0.8113 - 0 - 0.5 \\ &= 0.3113. \end{aligned}$$

Koji ćemo sada atribut uzeti za podjelu u analiziranom čvoru? Tablica u nastavku navodi utvrđene informacijske dobiti.

Podjela prema atributu	Informacijska dobit
Položaj	0.3113
Boja	0.3113

Kako za oba atributa imamo jednaku informacijsku dobit, možemo uzeti bilo koji od njih. Mi ćemo uzeti atribut *Položaj*. Stoga u stablo dodajemo novi čvor koji primjerke razvrstava prema atributu *Položaj*:



U najnižem lijevom čvoru stabla sada analiziramo primjerke 11 i 12; primijetite da oni imaju fiksirano *Oštećenje*=malo i *Položaj*=bliži:

Redni broj	Oštećenje	Položaj	Boja	Hitno
11.	malo	bliži	svijetlo	NE
12.	malo	bliži	tamno	DA

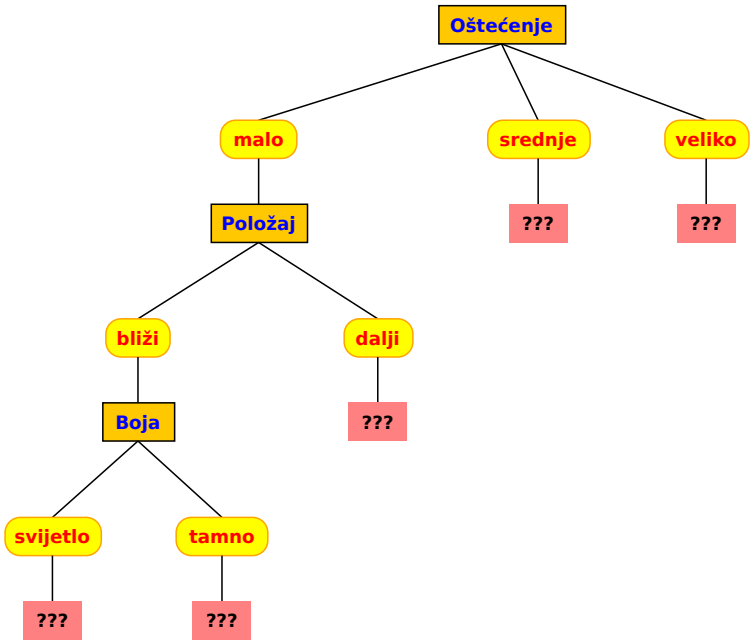
Veličina ovog podskupa je 2, a entropija 1. Preostalo nam je primjerke razvrstati po atributu *Boja*. Za *Boja*=svijetlo imamo:

Redni broj	Oštećenje	Položaj	Boja	Hitno
11.	malo	bliži	svijetlo	NE

a za *Boja*=tamno imamo:

Redni broj	Oštećenje	Položaj	Boja	Hitno
12.	malo	bliži	tamno	DA

Entropija oba skupa je 0, a informacijska dobit razvrstavanja po ovom atributu je 1. Stoga na ovom mjestu dodajemo novi čvor koji primjerke razvrstava po atributu *Boja* te u njegovu lijevu granu šaljemo prvi podskup, a u desnu granu drugi. Slika u nastavku prikazuje trenutno stablo:



U najdonjem lijevom čvoru na analizu dolazi samo primjerak:

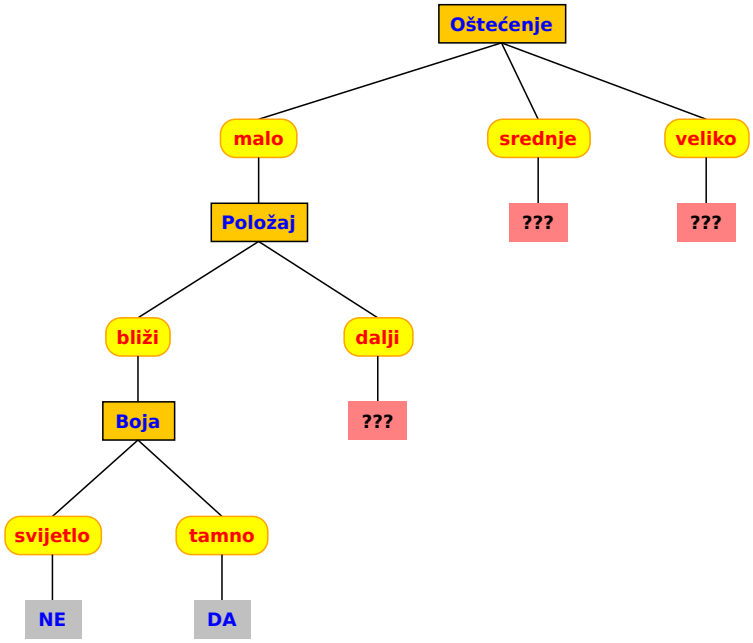
Redni broj	Oštećenje	Položaj	Boja	Hitno
11.	malo	bliži	svijetlo	NE

Kako ovdje imamo situaciju da svi primjerci pripadaju istom razredu, čvor pretvaramo u klasifikacijski čvor i zapisujemo da se primjercima pridjeljuje razred NE.

U susjedni čvor na analizu dolazi samo primjerak:

Redni broj	Oštećenje	Položaj	Boja	Hitno
12.	malo	bliži	tamno	DA

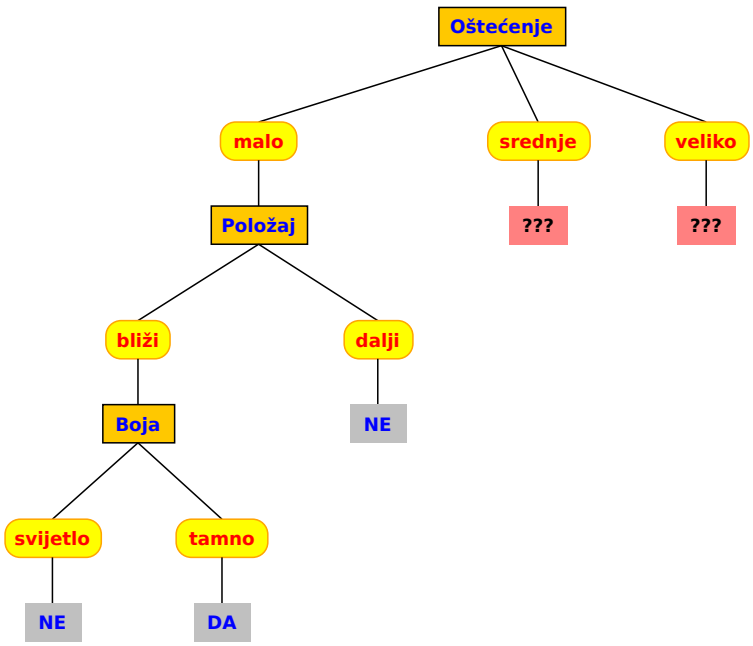
I ovdje imamo situaciju da svi primjerci pripadaju istom razredu. Čvor pretvaramo u klasifikacijski čvor i zapisujemo da se primjercima pridjeljuje razred DA. Trenutno izgrađeno stablo prikazano je na slici u nastavku.



U desno dijete čvora *Položaj* na analizu dolazi skup:

Redni broj	Oštećenje	Položaj	Boja	Hitno
3.	malo	dalji	svijetlo	NE
11.	malo	bliži	svijetlo	NE

Kako svi primjerci pripadaju istom razredu, čvor pretvaramo u klasifikacijski čvor i zapisujemo da se primjercima pridjeljuje razred NE. Trenutno izgrađeno stablo prikazano je na slici u nastavku.



Analizu nastavljamo sa srednjim djetetom korijenskog čvora. U njega smo poslali na analizu primjerke 2, 7, 9 i 10:

Redni broj	Oštećenje	Položaj	Boja	Hitno
2.	srednje	bliži	svijetlo	DA
7.	srednje	dalji	svijetlo	NE
9.	srednje	dalji	tamno	NE
10.	srednje	bliži	tamno	DA

Veličina ovog skupa je 4, a entropija 1. Skup možemo razdijeliti prema vrijednostima atributa *Položaj* odnosno prema vrijednostima atributa *Boja*.

Podjela prema vrijednostima atributa *Položaj* daje za *Položaj*=bliži:

Redni broj	Oštećenje	Položaj	Boja	Hitno
2.	srednje	bliži	svijetlo	DA
10.	srednje	bliži	tamno	DA

odnosno za *Položaj*=dalji:

Redni broj	Oštećenje	Položaj	Boja	Hitno
7.	srednje	dalji	svijetlo	NE
9.	srednje	dalji	tamno	NE

Entropija oba podskupa je 0 pa je informacijska dobit podjele prema atributu *Položaj* jednaka 1.

Podjela prema vrijednostima atributa *Boja* daje za *Boja*=svijetlo:

Redni broj	Oštećenje	Položaj	Boja	Hitno
2.	srednje	bliži	svijetlo	DA
7.	srednje	dalji	svijetlo	NE

odnosno za *Boja*=tamno:

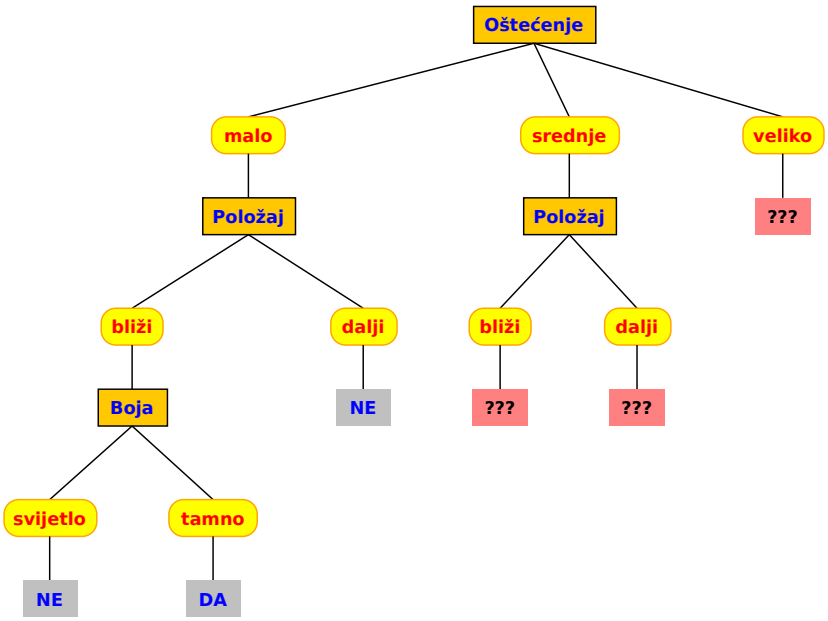
Redni broj	Oštećenje	Položaj	Boja	Hitno
9.	srednje	dalji	tamno	NE
10.	srednje	bliži	tamno	DA

Entropija oba podskupa je 1 pa je informacijska dobit podjele prema atributu *Boja* jednaka 0.

Koji ćemo sada atribut uzeti za podjelu u analiziranom čvoru? Tablica u nastavku navodi utvrđene informacijske dobiti.

Podjela prema atributu	Informacijska dobit
Položaj	1
Boja	0

Biramo atribut *Položaj*, te stvaramo čvor koji obavlja razvrstavanje prema njemu. Trenutno stanje stabla prikazano je na slici u nastavku.



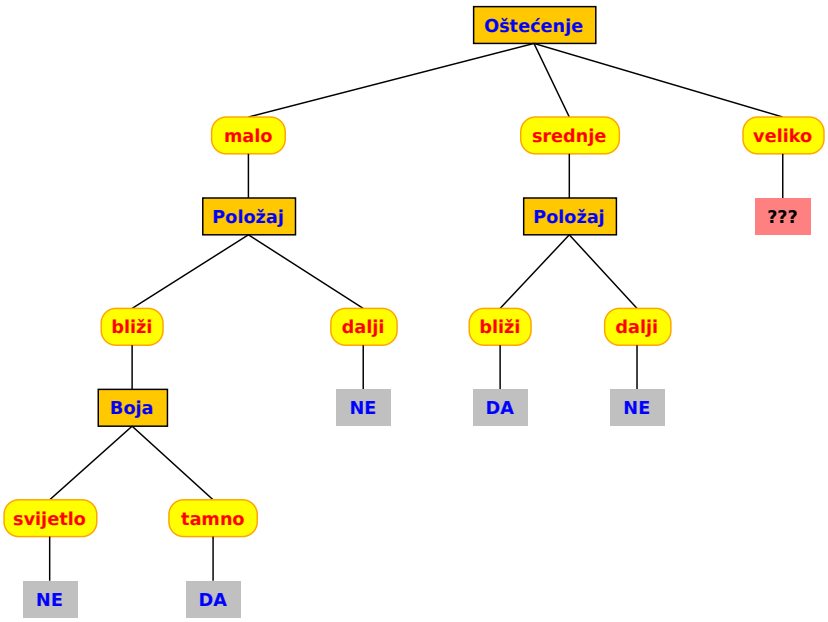
U lijevo dijete novostvorenog čvora šaljemo podskup:

Redni broj	Oštećenje	Položaj	Boja	Hitno
2.	srednje	bliži	svijetlo	DA
10.	srednje	bliži	tamno	DA

a u desno *Položaj*=dalji:

Redni broj	Oštećenje	Položaj	Boja	Hitno
7.	srednje	dalji	svijetlo	NE
9.	srednje	dalji	tamno	NE

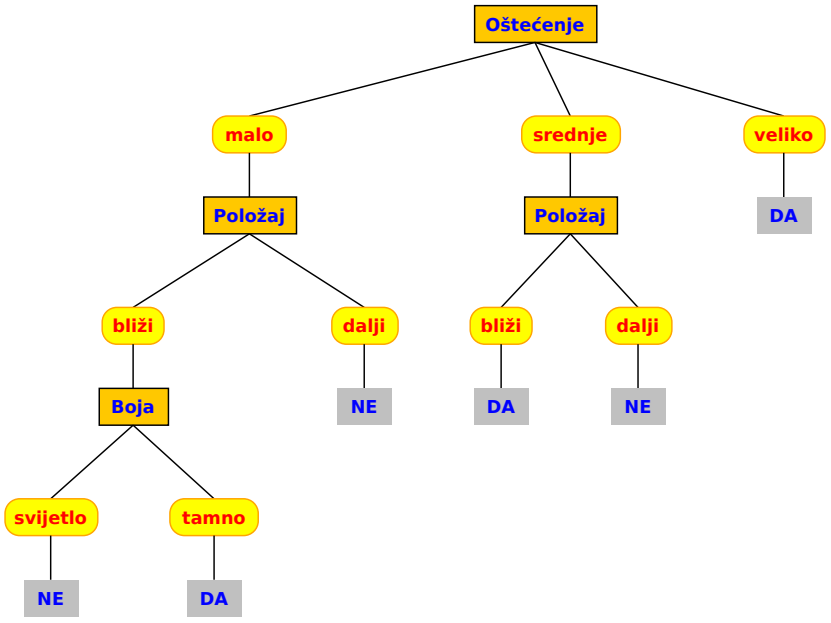
i rekurzivno provodimo algoritam. U oba slučaja utvrdit ćemo da svi primjerci imaju konzistentnu klasificikaciju (svi unutar podskupa pripadaju istom razredu), pa ćemo dodati čvorove koji obavljaju klasifikaciju; u prvom slučaju DA, a u drugom NE.



Konačno, u najdesnije dijete korijenskog čvora na analizu smo poslali podskup:

Redni broj	Oštećenje	Položaj	Boja	Hitno
4.	veliko	bliži	svijetlo	DA
5.	veliko	dalji	tamno	DA
6.	veliko	dalji	svijetlo	DA
8.	veliko	bliži	tamno	DA

Vidimo da svi primjerci pripadaju istom razredu; stoga ćemo na ovom mjestu napraviti čvor koji obavlja klasifikaciju u razred DA, što je prikazano na konačnoj slici u nastavku.



Isprobajte

Algoritam izgradnje stabla možete isprobati i samostalno. U naredbenom retku zadajte:
`java -cp book-ml.jar gui.id3.MainBuilder hitno.txt`
čime ćete pokrenuti program koji će Vam omogućiti da interaktivno gradite stablo odluke. Klikom na neizgrađeni čvor u donjem dijelu prozora prikazat će se kompletna analiza s objašnjenjima kao i linkovi kojima ćete moći odabrati kako želite dalje nastaviti. U slučajevima da u podskupu svi primjerci pripadaju istom razredu, samo ćete moći dodati klasifikacijski čvor. U suprotnom, moći ćete odabrati po kojem od atributa maksimalne informacijske dobiti (ako ih je više) želite napraviti podjelu.

Datoteka `hitno.txt` kao i datoteke `sport.txt` i `sport2.txt` su ugrađene datoteke. Možete i sami pripremiti svoje primjere: napravite tekstovnu datoteku na disku u kojoj prvi redak sadrži nazive atributa (posljednji atribut se smatra ciljnim) a svaki preostali redak po jedan primjerak. Kao separator koristite tabulator.

Želite li sami birati prema kojem atributu želite raditi podjelu (bez ograničenja da to mora biti jedan od onih koji daju maksimalnu informacijsku dobit), možete pokrenuti:
`java -cp book-ml.jar gui.id3.MainFreeBuilder hitno.txt`

Ako samo želite pogledati gotovo stablo, bez potrebe da ga gradite čvor po čvor, zadajte:
`java -cp book-ml.jar gui.id3.Main hitno.txt`

3.1 Formalna definicija algoritma ID3

Algoritam ID3 je rekurzivan pohlepni algoritam za izgradnju stabla odluke. Algoritam u rekurzivnom pozivu analizira trenutni skup primjeraka - označimo ga sa *S*.

1. Ako je skup primjeraka *S* prazan, tada se stvara klasifikacijski čvor koji primjerke klasificira u razred koji je najčešću u skupu primjeraka koji je razmatrao neposredni roditelj.
2. Ako svi primjerci u promatranom skupu *S* pripadaju istom razredu (označimo taj razred kao *r*), algoritam stvara klasifikacijski čvor koji primjercima dodjeljuje razred *r* i tu rekurzija staje.
3. Ako više nema atributa koji nisu razmatrani u nekom od roditelja (sve do korijena), stvara se klasifikacijski čvor koji primjercima pridjeljuje razred koji je najčešći u analiziranom skupu *S*.
4. U suprotnom, algoritam razmatra podjele skupova u podskupove prema svim atributima koji nisu u roditeljskim čvorovima. Algoritam odabire atribut koji daje maksimalnu informacijsku dobit, stvara čvor koji razmatra vrijednost tog atributa te za svaku različitu vrijednost atributa dodaje po jedno dijete koje gradi rekurzivnim pozivom uz podskup skupa *S* u kojem se nalaze samo oni primjerci kojima je odabrani atribut postavljen na razmatranu vrijednost.

Slučaj pod brojem 1 može se dogoditi u dubljim koracima rekurzije, s obzirom da malo po malo smanjujemo skup primjeraka koji analiziramo, da dođemo u situaciju da kada razmatrani skup razvrstavamo u podskupove prema vrijednostima nekog atributa, da je neki od tih podskupova prazan (primjerice, analizom smo došli u situaciju da razvrstavamo prema atributu *Oštećenje*, a svi primjerci u podskupu imaju *Oštećenje*=veliko ili *Oštećenje*=malo; tada bismo u granu koja odgovara *Oštećenje*=srednje poslali prazan skup koji bi potom bio razriješen kao slučaj 1. Evo i konkretnog primjera. Razmotrite skup podataka prikazan tablicom u nastavku.

Redni broj	Atribut1	Atribut2	Atribut3	Cilj
1.	a1	b1	c1	DA
2.	a1	b2	c1	DA
3.	a2	b1	c1	DA
4.	a2	b2	c1	DA
5.	a3	b1	c1	DA
6.	a3	b2	c1	DA
7.	a1	b1	c2	DA
8.	a1	b2	c2	DA
9.	a1	b3	c2	NE
10.	a3	b1	c2	NE
11.	a3	b2	c2	NE
12.	a3	b3	c2	NE

U korijenskom čvoru otkrit ćemo da se najveća informacijska dobit postiže podjelom po atributu *Atribut3*. Stoga ćemo napraviti korijenski čvor koji će primjerke razvrstavati prema atributu *Atribut3* i koji će imati dva djeteta: lijevo za *Atribut3*=c1 u koje ćemo dalje poslati primjerke:

Redni broj	Atribut1	Atribut2	Atribut3	Cilj
1.	a1	b1	c1	DA
2.	a1	b2	c1	DA
3.	a2	b1	c1	DA
4.	a2	b2	c1	DA
5.	a3	b1	c1	DA
6.	a3	b2	c1	DA

i desno za *Atribut3*=c2 u koje ćemo dalje poslati primjerke:

Redni broj	Atribut1	Atribut2	Atribut3	Cilj
7.	a1	b1	c2	DA
8.	a1	b2	c2	DA
9.	a1	b3	c2	NE
10.	a3	b1	c2	NE
11.	a3	b2	c2	NE
12.	a3	b3	c2	NE

Analizom ovog drugog slučaja utvrdit ćemo da se najveća informacijska dobit postiže podjelom po atributu *Atribut1*. Stoga ćemo stvoriti čvor koji obavlja razvrstavanje prema tom atributu i ima tri djeteta. U prvo ćemo poslati podskup primjeraka kod kojeg primjerci imaju *Atribut1*=a1:

Redni broj	Atribut1	Atribut2	Atribut3	Cilj
7.	a1	b1	c2	DA
8.	a1	b2	c2	DA
9.	a1	b3	c2	NE

srednjem djetetu ćemo poslati podskup primjeraka kod kojeg primjerci imaju *Atribut1*=a2:

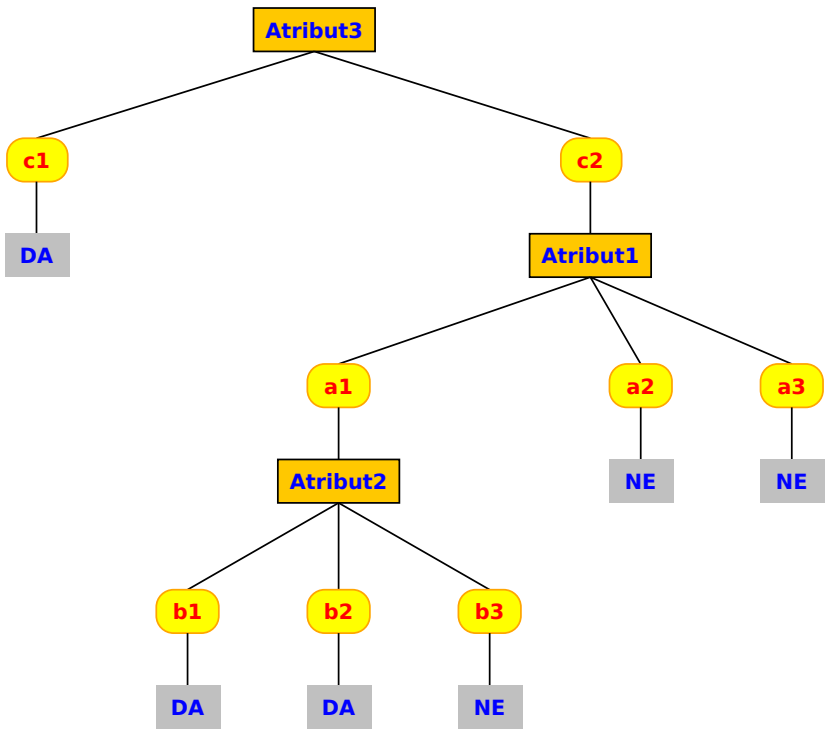
Redni broj	Atribut1	Atribut2	Atribut3	Cilj
-	-	-	-	-

i posljednjem djetetu ćemo poslati podskup primjeraka kod kojeg primjerci imaju *Atribut1*=a3:

Redni broj	Atribut1	Atribut2	Atribut3	Cilj
10.	a3	b1	c2	NE
11.	a3	b2	c2	NE
12.	a3	b3	c2	NE

Primijetite što se dogodilo sa srednjim djetetom: na analizu je dobilo prazan skup primjeraka. Obrada tog slučaja pogledat će primjerke koje je analizirao roditelj i ustanoviti isti imao 2 primjerka razreda DA i 4 primjerka razreda NE; posljedično, napravit će se terminalni klasifikacijski čvor koji će primjercima dodijeljivati razred NE.

Konačan izgled stabla odluke nakon što su razriješeni svi čvorovi prikazan je na slici u nastavku.



Isprobajte

Ovo možete isprobati i samostalno. U naredbenom retku zadajte:
`java -cp book-ml.jar gui.id3.MainBuilder slucaj1.txt`

Važno. Umjesto opisanog načina konstrukcije stabla koja u svakom čvoru radi grananje po svim poznatim vrijednostima atributa (neovisno o tome postoji li u analiziranom podskupu primjeraka ijedan primjerak s takvom vrijednosti), moguće je napraviti inačicu koja svakom čvoru koji radi grananje prema nekom atributu pridijeli pretpostavljenu vrijednost ciljnog atributa (kao onu koja je najzastupljenija u promatranom podskupu), i potom radi grananje samo prema vrijednostima atributa koje su doista prisutne u promatranom podskupu.

Opisana modifikacija u određenom je smislu čak i robusnija, jer omogućava uporabu stabla za klasifikaciju čak i onih primjeraka koji sadrže vrijednost atributa koja nije prethodno bila viđena.

3.1.1 Svojstva i nadogradnje algoritma

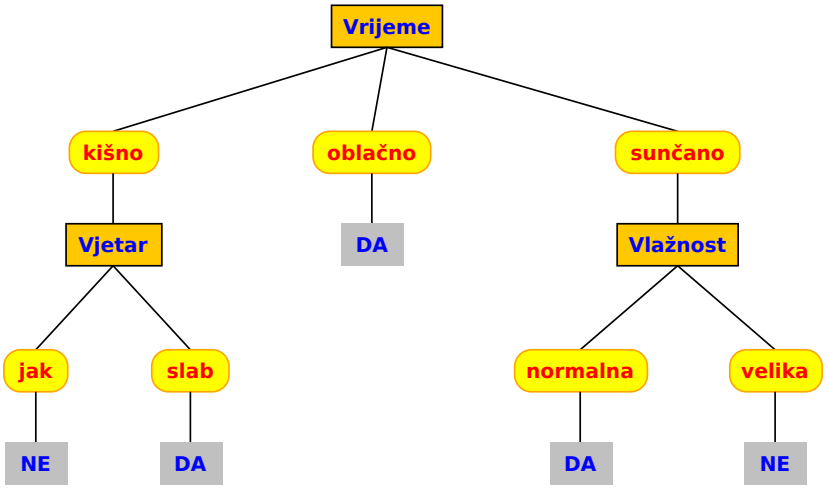
Algoritam ID3 je pohlepan algoritam jer u svakom koraku donosi lokalno optimalnu odluku. Time algoritam može upasti u lokalni minimum i izgraditi stablo koje nije doista minimalno moguće.

Daljnja nadogradnja ovog algoritma su algoritmi C4.5 te C5 koje ovdje nećemo razmatrati. Spomenimo da postoje i daljnja proširenja ideje stabala odluke, a jedan od primjera su takozvane *Slučajne šume* (engl. *Random forest*): kod tog pristupa na temelju različitih podskupova atributa primjerka gradi se niz klasifikacijskih stabala; prilikom uporabe, svako od stabala obavlja klasifikaciju te se na temelju njihovih rezultata mogu odrediti vjerojatnosti da primjerak pripada svakom od razreda (primjerice, razmotrite slučaj gdje šuma ima 100 stabala, i 90 od njih primjerak klasificira kao DA, a 10 kao NE). Također, spomenimo da se prilikom odabira atributa ne mora uvijek gledati informacijska dobit već postoje i druge mjere kvalitete podjele (primjer je *gini index* – ostavljamo zainteresiranom čitatelju da istraži o čemu se radi).

U prisustvu stršećih vrijednosti, algoritam se može prenaučiti. Posljedica toga jest staranje stabla koje ima puno više čvorova no što je to potrebno, te koje loše generalizira. Jedna od tehnika popravljanja generalizacijskih sposobnosti stabla jest postupak podrezivanja koji se najčešće provodi nakon što je (potencijalno prenaučeno) stablo izgrađeno. Kreće se od najdubljih čvorova stabla i putuje prema korijenu. Razmatra se cijena uklanjanja podstabla kojem je razmatrani čvor korijen te zamjena tog čitavog podstabla jednim klasifikacijskim čvorom koji bi primjercima pridruživao razred koji je bio najčešći među primjercima koje je čvor analizirao prilikom izgradnje stabla. Ako je ta cijena prihvatljiva, provodi se uklanjanje podstabla i zamjena klasifikacijskim čvorom.

3.2 Skup primjeraka *Dan za sport*

Jedan od čestih primjera na kojem se analizira izgradnja stabla odluke jest skup primjeraka *Dan za sport*. Klasifikacijsko stablo koje obavlja klasifikaciju primjeraka prikazano je na slici u nastavku.



Skup primjeraka *Dan za sport* prikazan je tablicom u nastavku.

Redni broj	Vrijeme	Temperatura	Vlažnost	Vjetar	Igra
1.	sunčano	vruće	velika	slab	NE
2.	sunčano	vruće	velika	jak	NE
3.	oblačno	vruće	velika	slab	DA
4.	kišno	ugodno	velika	slab	DA
5.	kišno	hladno	normalna	slab	DA
6.	kišno	hladno	normalna	jak	NE
7.	oblačno	hladno	normalna	jak	DA
8.	sunčano	ugodno	velika	slab	NE
9.	sunčano	hladno	normalna	slab	DA
10.	kišno	ugodno	normalna	slab	DA
11.	sunčano	ugodno	normalna	jak	DA
12.	oblačno	ugodno	velika	jak	DA
13.	oblačno	vruće	normalna	slab	DA
14.	kišno	ugodno	velika	jak	NE

Isprobajte

Čitav postupak izgradnje ovog stabla odluke možete isprobati i samostalno. U naredbenom retku zadajte:
`java -cp book-ml.jar gui.id3.MainBuilder sport.txt`
pa kliknite na korijenski čvor. U donjem dijelu prozora prikazat će se sva objašnjenja i izračuni i tu ćete moći odabrati željenu akciju. Ponavljanjem postupka malo po malo moći ćete izgraditi čitavo stablo.

Bibliografija

Knjige

Članci

Konferencijski radovi i ostalo