Učenje stabala odluke

Mauro Raguzin

27. veljače 2023.

- Stabla odluke: jedan od najstarijih i povijesno najuspješnijih pristupa strojnog učenja
- Učimo funkciju hipoteze iz vektora vrijednosti na ulazu
- Vektori poprimaju vrijednosti nekog konačnog broja *atributa*
- Na izlazu dajemo jednu vrijednost — *odluku*
- Odluka je za nas booleovska — da ili ne ($+$ ili $-$)
- Iz skupa atributa ovog problema učenja izdvajamo *ciljni* atribut koji odgovara klasifikaciji primjera

Stablo odluke je ekvivalentno formuli logike sudova

$$Cilj \Leftrightarrow (Put_1 \vee Put_2 \vee \ldots),$$

gdje je svaki *Put* konjunkcija odgovarajućih testova atributa na vektoru. Iz poznatih rezultata iz logike sudova možemo zaključiti da je ova formula ekvivalentna nekoj disjunktivnoj normalnoj formi, što znači da se svaka funkcija logike sudova može zapisati kao stablo odluke.
Lako možemo zaključiti da je ovakvih stabla jednostavno previše da bi bilo moguće izgraditi garantirano optimalno stablo tj. ono koje reprezentira ulaze s najmanjim brojem čvorova ili najmanjom visinom.

Ipak, primjenom dobrih pohlepnih heurstika, moguće je dobiti sasvim razumna stabla odluke.

- Glavni algoritam učenja je rekurzivan: podijeli-pa-vladaj+pohlepna heuristika
- Možemo ga promatrati kao dubinsko pretraživanje (DFS), samo što sada *dodajemo* podstabla u svakoj razini rekurzije
- Ta podstabla odgovaraju svim mogućim odabirima grane s trenutnog internog čvora (atributa)
- Grane do čvorova-djece predstavljaju jednu od odabranih vrijednosti tog (roditeljskog) atributa
- Pri svakom silasku u novo podstablo, moramo kao skup relevantnih primjera u novom potproblemu uzeti *samo* one primjere koji imaju odgovarajuću komponentu vektora jednaku vrijednosti odabrane grane; također, zanemarujemo odabran atribut jer on više ne može biti relevantan u daljnjim odlukama

Rekurzija jasno u nekom trenutku mora završiti, što se ovdje može dogoditi na četiri posebna načina:

1. Ako su *svi* preostali (pot)primjeri pozitivni ili negativni (k), tada smo gotovi s ovom granom: trenutni čvor je list s klasifikacijom k.
2. Ako nam je preostalo nekoliko pozitivnih i nekoliko negativnih primjera, odabiremo *najboljeg* koji će predstavljati ovaj interni čvor; korištenja mjera dobrote je objašnjena ispod.
3. Ako više uopće nema preostalih primjera, znači da smo iscrpili naučenu bazu i nije moguće klasificirati samo na temelju trenutnih primjera, pa kao jedinu mogućnost koristimo već iskorištene primjere za učenje roditeljskog čvora: vrijednost klasifikacije ovog lista se dobiva kao većinska funkcija nad (ulaznim) klasifikacijama tih primjera.
4. Inače, u situaciji smo gdje imamo neke preostale primjere, ali nemamo više atributa koji bi se testirali. Tada vraćamo većinsku vrijednost klasifikacije *preostalih* primjera.

Najbolji atribut za podjelu tj. za testiranje na danom internom čvoru odabiremo pohlepno, koristeći mjeru dobitka informacije

$$Gain(A) = B\left(\frac{p}{p+n}\right) - \sum_{k=1}^{d} \frac{p_k + n_k}{p+n} B\left(\frac{p_k}{p_k + n_k}\right), \quad (1)$$

gdje je

$$H(Cilj) = B(q) = -(q \log_2 q + (1-q)\log_2(1-q)),$$

a p i n su respektivno brojevi pozitivnih i negativnih primjera na ovoj razini stabla; p_k i n_k odgovaraju tom broju u podstablu u koje siđemo odabirom jedne od d mogućih vrijednosti trenutnog atributa.
Ovaj pristup često daje jako dobre rezultate, ali postoje određene poteškoće koje ćemo sada spomenuti.

- Jedan od glavnih problema kod učenja stabala je *redundantna izražajnost* (engl.*overfitting*)
- To je fenomen gdje stabla odluke mogu pokupiti razne, u stvarnosti kompletno irelevantne veze između atributa i ishoda
- To je tipično za skupove podataka s velikom količinom šuma te velikim brojem atributa, a može se donekle poboljšati povećanjem broja podataka za učenje
- Postoje razne tehnike *podrezivanja* koje rješavaju ovaj problem: eliminiranju one unutarnje čvorove koji ne doprinose značajno odluci, počevši od listova izgrađenog stabla i penjajući se uz stablo. Pritom se ograničavamo na promatranje samo onih unutarnjih čvorova čija su sva djeca listovi.
- Ovdje se određivanje irelevantnosti radi korištenjem testova značajnosti: želimo provjeriti postoji li dovoljno velika vjerojatnost da nul-hipoteza vrijedi za dani atribut, koristeći poznate statističke distribucije i prag dozvoljene vjerojatnosti

- Još jedan problem je i baratanje s *numeričkim* atributima
- Do sada smo pretpostavljali da će stablo obrađivati kategoričke atribute, kod kojih ne očekujemo velik raspon vrijednosti; s druge strane, ako nam se na ulazu pojave atributi s domenom u proizvoljnom (recimo realnom) skupu brojeva, onda više ne možemo očekivati da će postupak učenja ikada uočiti sve vrijednosti takve domene
- Također, naivni pristup učenju ovakvih atributa može dovesti do jako visokih i nerazumljivih stabala: rješenje je u particioniranju domene svih numeričkih atributa na način da odabrane točke podjele čine diskretiziranu domenu, uz maksimizaciju dobitka informacije pri takvom odabiru. Tu su opet korisne pohlepne heuristike.

Dio ovog rada je i Java program koji implementira osnovni algoritam učenja stabala odluke kao i sva iznad spomenuta poboljšanja. Osnovni pristup učenju je temeljen na algoritmu iz [1], s dodatkom koji omogućuje istovremeni rad i s kategoričkim i s numeričkim atributima.
Ulaz programa:
`datoteka.csv` $\texttt{vrijednost}_1$ $\texttt{vrijednost}_2$ ... $\texttt{vrijednost}_n$

Pri particioniranju numeričkih atributa se koristi *segmentiranje pod nadzorom* (engl. *supervised binning*), pristup ukratko opisan u [1] i prikazan u [3]

- Ideja je da možemo u samo jednom prolazu kroz sve uočene numeričke vrijednosti danog atributa napraviti kompletnu "optimalnu" (u pohlepnom smislu) particiju ukoliko prije početka particioniranja sortiramo sve te vrijednosti
- Tada algoritam postaje jednostavan rekurzivan, podijeli-pa-vladaj zadatak pri čemu samo treba pripaziti na baratanje s duplikatima, što je ipak olakšano zbog sortiranosti
- Pohlepnost dolazi do izražaja na praktički jednak način kao kod računa (1), samo što se sada maksimizira dobitak pri odabiru točke podjele raspona vrijednosti, a ne odabira cijelog atributa
- Točka podjele mora ležati između dviju susjednih ali različitih klasifikacija, dakle pri prijelazu s $+$ na $-$ ili obratno

Objasnimo još implementaciju podrezivanja izgrađenog stabla, koja koristi χ^2 distribuciju

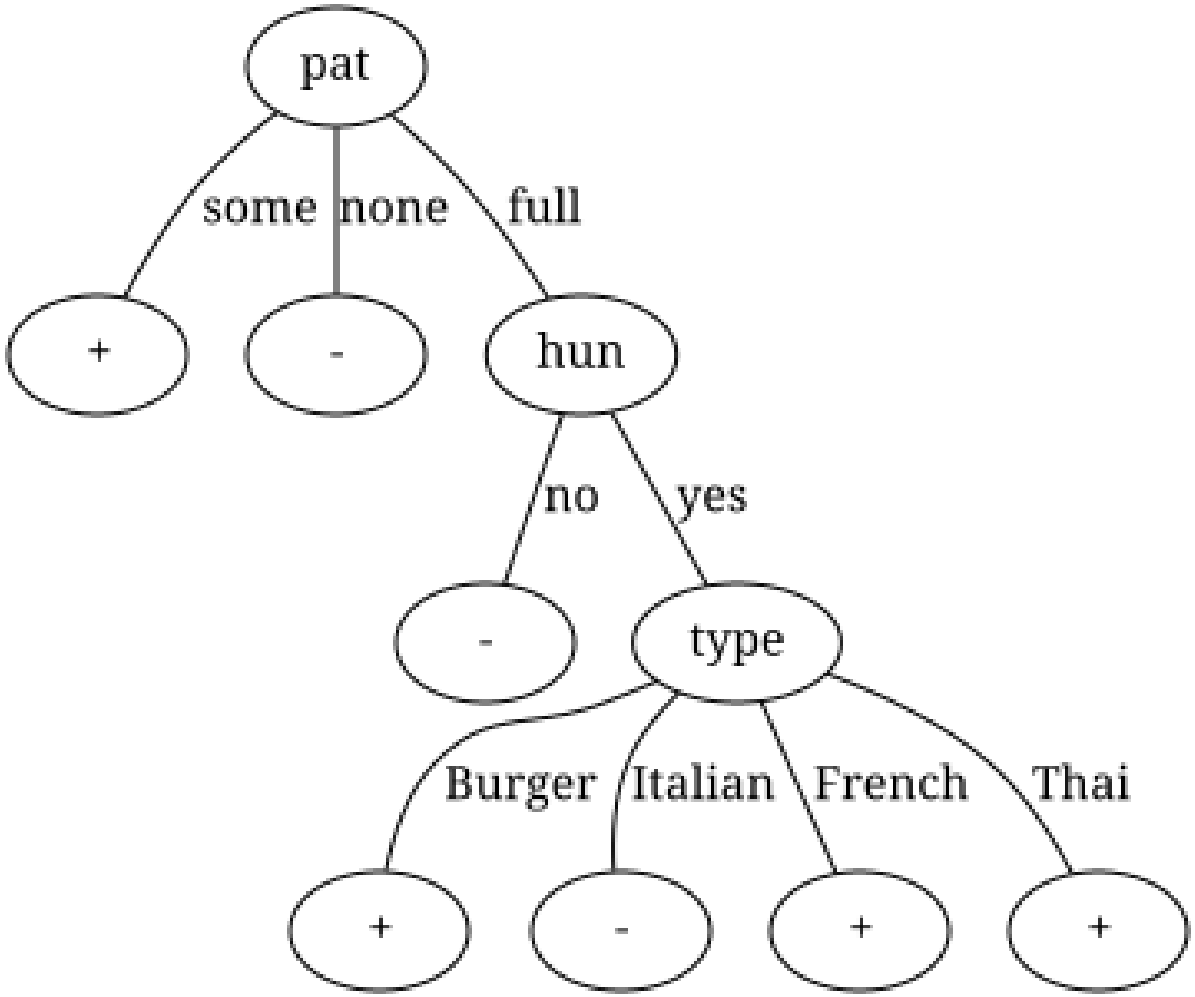
- Pretpostavimo da je na trenutnom čvoru stabla dostupan uzorak od $v = n + p$ primjera
- Mi želimo izračunati očekivano odstupanje (pod nul-hipotezom) od stvarno izmjerenog. Dakle, za atribut ovog čvora moramo izračunati očekivani broj pozitivnih primjera $\hat{p}_k$ i očekivani broj negativnih primjera $\hat{n}_k$ za svih d različitih vrijednosti ovog atributa $(k = 1, 2, \ldots, d)$.
- Zatim ukupno odstupanje možemo dobiti kao

$$\Delta = \sum_{k=1}^{d} \frac{(p_k - \hat{p}_k)^2}{\hat{p}_k} + \frac{(n_k - \hat{n}_k)^2}{\hat{n}_k}$$

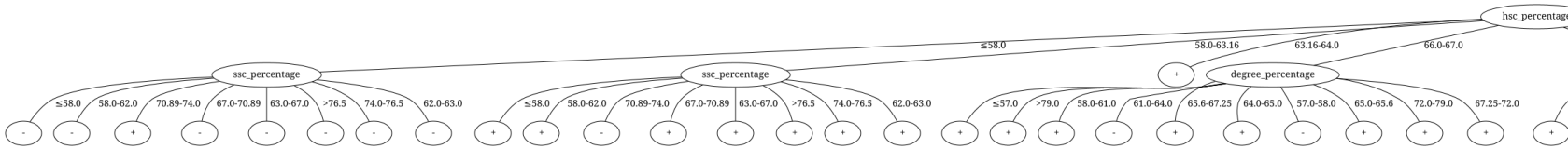
jer je pod nul-hipotezom očekivana vrijednost te veličine distribuirana upravo po χ^2 distribuciji s $v - 1$ stupnjeva slobode

- Moramo dobiti donju među odgovarajućih vrijednosti χ^2 distribucije kako bismo vidjeli je li naša Δ prihvatljiva; ako je premala, p-vrijednost je prevelika da bismo bili sigurni da nul-hipoteza ne vrijedi pa odbacujemo atribut. Trenutni prag p-vrijednosti je u programu postavljen na 0.05, što je uobičajena vrijednost [1].
- Spomenutu donju među možemo dobiti računanjem inverzne CDF odnosno kvantila χ^2 distribucije za kumulativnu vjerojatnost $1 - p$, što radimo pomoću Apache Common Math biblioteke

U [1] je dana mala tablica (12 primjera) s podacima koji odgovaraju nečijem odabiru restorana ovisno o raznim okolnostima, poput razine gladi, tipu hrane koja se nudi, postoji li prethodna rezervacija itd. Primjenom izrađenog programa (koji na kraju izvodi i podrezivanje stabla) na taj skup podataka dobivamo stablo sa slike 13.



Sljedeći primjer koristi bazu podataka za učenje sa stranice Kaggle [2] kako bi predvidio hoće li se diplomant na ulazu (opisan u terminima predmeta studija na različitim razinama obrazovanja, ocjena tijekom cijelog školovanja, rezultata standardiziranih testova i sl.) moći odmah zaposliti tj. biti primljen od strane neke firme ili ne. Kako je originalan skup podataka poprilično velik i raznolik, rezultirajuće stablo je preveliko za prikazati ovdje, pa na slici 14 prikazujemo jedno manje stablo dobiveno na podskupu ovih primjera, nakon obrezivanja.



Slika: Stablo iz primjera zapošljavanja

Korištena literatura

[1] Stuart Russell i Peter Norvig. *Artificial intelligence: A modern approach, global edition*. 3. izdanje. London, England: Pearson Education, travanj 2016.

[2] Ahsan Raza. *Job Placement Dataset — Kaggle*. URL: https://www.kaggle.com/datasets/ahsan81/job-placement-dataset.

[3] Saed Sayad. *Supervised Binning*. Pristupljeno = 2023-02-26. URL: https://www.saedsayad.com/supervised_binning.htm.