Evolucija drveta klasifikacije upotrebom genetičkog programiranja

Petar Zečević 169/2016 i Aleksandra Ružić 47/2016

Uvod

Drvo klasifikacije predstavlja model koji podacima iz skupa podataka sa određenim atributima, treba da odredi klase tj. vrednosti ciljne promenljive. Model (drvo) se pravi nad trening skupom, a testira nad test skupom (u oba su poznate vrednosti ciljne promenljive). Drvo klasifikacije je stablo u čijim su unutrašnjim čvorovima uslovi, a u listovima klase ciljne promenljive. Uslovi su oblika 'atribut' 'relacija' 'vrednost', gde je 'atribut' ime kolone iz skupa podataka, 'relacija' je operacija iz dozvoljenog skupa operacija (što je {<, <=, >, >=, ==, !=} ako je atribut numerički, u suprotnom {==, !=}), a 'vrednost' je moguća vrednost iz dozvoljenog skupa vrednosti tog atributa. Kada se primer iz skupa podataka "provlači" kroz drvo, za svaki čvor se proverava da li podatak ispunjava uslov. Ako ispunjava, ide u levi, a u suprotnom u desni čvor. List do kog dođe, određuje njegovu vrednost ciljne promenljive.

Nalaženje optimalnog drveta klasifikacije genetskim programiranjem podrazumeva da su stabla jedinke i da se operatori genetskog algoritma (selekcija, mutacija, ukrštanje) implementiraju u skladu s tim. U radu će biti razmatran blokovki pristup genetskom algoritmu koji traži drvo klasifikacije. Dodatno, zbog specifičnosti problema i kompleksnosti jedinke, uvode se dodatni operatori (potkresivanje, mutacija relacije, mutacija vrednosti, dodavanje gradivnih blokova).

Rad opisuje implementaciju varijante BGP algoritma (Building block approach to Genetic Programming). BGP je evolutivni algoritam koji služi za formiranje stabla odlučivanja. Ideja BGP algorimta je da početna populacija ima jedinke što manje dubine i da se kroz generacije ta dubina kontrolisano povećava. Time se obezbeđuje da, za rezultat, algoritam bira stablo sa što manjom dubinom, a što znači da se izbegavaju preprilagođeni modeli i modeli koji u podstablima sadrže međusobno kontradiktorna ili suvišna pravila. Rezultati i impementacija algoritma će biti poređeni sa rezultatima i idejnom implementacijom BGP-a u radu "Searching the Forest: Using Decision Trees as Building Blocks for Evolutionary Search in Classification Databases" autora SE Rouwhorst i AP Engelbrecht [1]. Algoritam je implementiran u programskom jeziku Python.

Opis BGP algoritma

Jedinke algoritma su stabla. Inicijalna populacija se formira od nasumičnih stabala dubine 1 (dakle, koren u kom je uslov i dva lista u kom su klase). Veličina populacije se bira eksperimentalno. Ocena kvaliteta jedinke je preciznost (mogu se koristiti i druge ocene kvaliteta stabla). Operatori i njihovi opisi su sledeći:

- *mutacija relacije*: bira se nasumično unutrašnji čvor stabla, i bira se nasumična relacija iz skupa dozvoljenih relacija za taj atribut i postavlja se kao relacija u uslovu u tom čvoru.
- *mutacija vrednosti*: bira se nasumično unutrašnji čvor stabla, i bira se nasumična vrednost iz skupa dozvoljenih vrednosti za taj atribut i postavlja se kao vrednost u uslovu u tom čvoru.
- *ukrštanje*: za obe jedinke bira se nasumično unutrašnji čvor, i podstabla ispod datih čvorova se zamenjuju. Izabrani čvorovi ne moraju biti na istoj dubini.

- *potkresivanje*: bira se nasumično unutrašnji čvor stabla i zamenjuje se listom. Potkresivanje se vrši da bi se izbeglo preprilagođavanje.
- *selekcija*: turnirska. Ako je veličina turnira k, bira se iz populacije nasumičnih k jedinki i one "učestvuju u turniru". Pobednik turnira učestvuje u ukrštanju. U jednoj varijanti ove selekcije, pobednik je jedinka sa najvećom prilagođenošću. U drugoj, izabrane turnirske jedinke se sortiraju po prilagođenosti, i šansa da pobedi i-ta jedinka je `p*(1-p)^(i-1)`, gde je `p` korisnički definisan parametar koji se (pogodili ste) eksperimentalno određuje. Postoje i mnoge druge varijante ove selekcije [2]. Veličina turnira se bira eksperimentalno.
- *dodavanje gradivnih blokova*: bira se nasumično list iz stabla i zamenjuje se nasumičnim drvetom dubine 1.

Za svaki od operatora, osim selekcije, postoji po parametar koji predstavlja verovatnoću njegovog dešavanja. Ti parametri se određuju eksperimentalno. Takođe, da bi se dodavao gradivni blok na jedinke, potrebno je da bude ispunjen sledeći uslov: `(avg_depth + avg_width)-(prev_avg_depth + prev_avg_width) < L`, gde su:

- `avg_depth` - prosečna dubina stabla u trenutnoj generaciji
- `avg_width` - prosečna širina stabla u trenutnoj generaciji (broj listova)
- `prev_avg_depth` - prosečna dubina stabla u prošloj generaciji
- `prev_avg_width` - prosečna širina stabla u prošloj generaciji
- `L` - korisnički zadat parametar. Određuje se eksperimentalno.

Kriterijum zaustavljanja je `min_rule_acc < e^(c * trainsize * A)`, gde su:

- `A = 1/T(0) - 1/T(t)`, gde je `T(t) = T(0) - t` "temperaturna" funkcija broja iteracija (sa povećanjem broja iteracija, funkcija "hladi" program, smanjujući gornji izraz, a time povećavajući mu šansu da se zaustavi). `T(0)` je korisnički zadat parametar. Određuje se eksperimentalno.
- `trainsize` - veličina trening skupa
- `min_rule_acc` - najmanja preciznost među pravilima
- `c` - korisnički zadat parametar. Određuje se eksperimentalno.

Algoritam počinje od formiranja inicijalne populacije. Onda u se svakoj generaciji, redom, ispituje da li mogu da se dodaju gradivni blokovi na jedinke iz populacije, biraju jedinke za ukrštanje, vrši ukrštanje, mutacije i potkresivanje. Pamti se i ažurira u svakoj generaciji najbolja jedinka. Generacije se smenjuju dok se ne ispuni kriterijum zaustavljanja.

Opis algoritma je uzet iz [1].

Implementacija i razlike

U radu, drvo je direktno implementirano kao stablo. Čvor je predstavljen klasom koja ima dve potklase: jedna za unutrašnje čvorove i jedna za listove. Klasa za unutrašnje čvorove sadrži atribute koji označavaju levog potomka, desnog potomka i uslove čvora (atribut, relacija i vrednost). Klasa za listove sadrži atribut koji označava klasu. Klasa za čvor ima i atribut indeks koji označava poziciju čvora u drvetu, a pozicije se dodeljuju kao za balansirano binarno drvo. Balansirano drvo je binarno stablo kod kog za svaki čvor važi da je aspolutna vrednost razlike dubina njegovih podstabala najviše 1. Indeksi kod takvih stabala se dodeljuju na sledeći način: koren je indeksa 1, a deca čvora sa indeksom i, imaju indekse 2i i 2i+1 . Klasa koja predstavlja drvo sadrži atribut koji predstavlja koreni čvor. Drvo se instancira tako što se napravi koren, a zatim se dodaju čvorovi na mesta njegovih potomaka. Takođe, zato što drvo nije balansirano, klasa za drvo takođe ima atribut koji označava

skup zauzetih indeksa, odnosno pozicija, da bi se omogućio pristup bilo kom čvoru iz stabla (što je neophodno da bismo mogli pristupiti nasumičnom čvoru za bilo koji od gore navedenih operatora). Klasa za drvo ima metod koji izračunava preciznost drveta.

U BGP algoritmu, u uslovima unutrašnjih čvorova, na desnoj strani relacije može da stoji i atribut. U našoj implementaciji to nije moguće. Razlog za to je, osim jednostavnosti, i činjenica da originalni Hantov algoritam tu osobinu ne podržava.

Umesto datog izraza za zaustavljanje, za kriterijum zaustavljanja smo koristili maksimalan broj iteracija. Napravljene su tri verzije funkcije koje se razlikuju u kriterijumu za dodavanje gradivnih blokova. Prva verzija nema kriterijum za dodavanje gradivnih blokova. Druga verzija koristi kriterijum iz gore navedenog rada. Treća verzija koristi sledeći uslov za dodavanje gradivnog bloka: `avg_depth >= tree_depth AND avg_width >= tree_width` gde su:

- `avg_depth`- prosečna dubina stabla u trenutnoj generaciji.
- `tree_depth`- dubina stabla koje se razmatra za nadogradnju.
- `avg_width`- prosečna širina stabla (broj listova) u trenutnoj generaciji.
- `tree_width`- širina stabla (broj listova) koje se razmatra za nadogradnju.

Prva verzija je pravila stabla prevelike dubine u odnosu na broj atributa i zato njeni rezultati neće biti prikazani u radu, a rezultati druge dve verzije će biti upoređivani, međusobno i sa rezultatom iz referisanog rada.

Rezultati

Algoritam je testiran sa sledećim parametrima:

- veličina test skupa = 30%
- broj iteracija = 30
- brojnost populacije = 1000
- veličina turnira za selekciju = 20
- verovatnoća za ukrštanje = 0.9
- verovatnoća za mutaciju relacije = 0.6
- verovatnoća za mutaciju desne strane uslova = 0.7
- verovatnoća potkresivanja = 0.2
- verovatnoća dodavanja gradivnog bloka = 0.3.

Algoritam je pokretan za sledeća tri skupa podataka:

- Iris- podaci o cveću. Postoji 150 redova i 5 kolona. Svaki red se odnosi na po jedan cvet. Atributi su dužina latice (Petal Length), širina latice (Petal Width), dužina listića čašice (Sepal Length) i širina listića čašice (Sepal Width). Svi atributi su kontinualni. Klasa je vrsta cveća. Postoji 3 različite klase: setosa, virginica i versicolor. Sve tri klase su podjednako zastupljene. Atributi koji opisuju latice su jako korelisani sa ciljnom klasom, dok su atributi koji opisuju listiće čašice slabo korelisani sa ciljnom klasom. Klase virginica i versicolor su slične međusobno, dok se setosa dosta razlikuje od njih.
- Ionosphere- podaci o signalima prikupljenih radarom. Postoji 351 red i 35 kolona. Svi atributi su kontinualni. Klasa predstavlja da li je rezultat snimanja dobar ili loš. Ima dve klase koje su nejednako zastupljene.
- Pima-indians-diabetes - podaci o dijabetesu među ženama indijanskog porekla iznad 21 godine. Postoji 768 redova i 9 kolona. Svaki red opisuje jednu osobu. Atributi su kontinualni i predstavljaju medicinske podatke koi su potencijalni indikatori dijabetesa. Atributi su sledeći: broj dosadašnjih trudnoća, glukoza, krvni

pritisak, debljina kože, nivo insulina u krvi, BMI, funkcija dijabeteskog pedigrea i starost. Klasa označava da li osoba ima dijabetes. Klase su nejednako zastupljene (1 se nalazi u 268 redova).

Skupovi su uzeti sa sajta "UCI Machine Learning archive": http://www.ics.uci.edu/ ~ mlearn.

Obe verzije algoritma su pokretane 15 puta za svaki skup. Rezultati pokretanja su upoređeni sa rezultatima iz [1] i modelom pravljenim pomoću biblioteke sklearn u sledećoj tabeli:

Podaci	Rezultati	BGP	Verzija2	Verzija3	DecisionTree (sklearn)
Iris	*Trening preciznost*	-	0,92933333	0,881333333	0,977333333
	Test preciznost	0,941	0,92666667	0,865333333	0,941333333
	Prosečan broj pravila	3.73	3,66666667	3	4,666666667
	Prosečan broj konjukcija u pravilu	2.02	2,09090909	1,666666667	2,5
Ionosphere	*Trening preciznost*	-	0,864113475	0,85787234	0,946729375
	Test preciznost	0.892	0,786206897	0,863793103	0,900858091
	Prosečan broj pravila	4.70	5,4	2,933333333	6,533333333
	Prosečan broj konjukcija u pravilu	2.39	2,975308642	1,636363636	2,806122449
Pima	*Trening preciznost*	-	0,754085603	0,763294423	0,83621781
	Test preciznost	0.725	0,74015748	0,722834646	0,660158493
	Prosečan broj pravila	3.70	5	2,866666667	25
	Prosečan broj konjukcija u pravilu	1.97	3,186666667	1,604651163	4.821333333

Razlike koje se moraju pomenuti između [1] i ovog rada su sledeće:

1. Broj pokretanja- u [1] algoritam je za svaki skup pokretan 30 puta, što nama zbog hardverskih ograničenja nije bilo moguće.
2. Parametri- gore navedeni parametri su isti za obe verzije i sva tri skupa dok su u [1] autori menjali te parametre u zavisnosti od skupa za koji su pokretali. Naglasili bismo razliku u veličinama test skupova među podacima. Dosta veći trening skupovi su verovatno doprineli boljim rezultatima u [1].

Zaključak

Prva verzija funkcije (bez kriterijuma za dodavanje gradivnih blokova) iako daje pristojne preciznosti, pravi preduboka stabla sa redudantnim ili besmislenim uslovima. Rešenje bi bilo potkresivanje loših (u smislu redudantnih ili besmislenih) uslova nakon formiranja stabla ili povećavanje verovatnoće za potkresivanje i smanjivanje verovatnoće za dodavanje gradivnog bloka tokom formiranja stabla.

Druga verzija za manje skupove (Iris i Ionosphere) pravi za nijansu veća stabla sa lošijim preciznostima (pogotovo za Ionosphere) od klasičnog BGP algoritma. Za veće skupove (Pima) se pokazala bolje. Napravila je značajno veće stablo sa malo većom preciznošću od BGP-a.

Treća verzija se lošije pokazala na Iris skupu, mada ako bi se iz računatog proseka preciznosti izbacila dva jako loša primera koja odudaraju od ostatka rezultata, preciznost bi bila mnogo bolja. Problem je verovatno bio to što je početna generacija bila jako loša i odatle nije mogla da se napravi bolja jedinka. Potencijalno rešenje bi bilo da se poveća broj jedinki u populaciji ili da se poveća verovatnoća mutacija i/ili verovatnoća dodavanja gradivnog bloka. U ostalim skupovima daje slične preciznosti kao BGP (mada za nijansu manje), ali sa manjim stablima i po širini i po dubini, što je značajno jer je jedna od osnovnih prednosti BGP algoritma pravljenje manjih stabala koja izbegavaju preprilagođavanje.

Stabla koja smo formirali pomoću biblioteke sklearn imaju slične preciznosti kao i stabla formirana BGP-om za Iris i Ionosphere. Za skup Pima, BGP je značajno bolji, što je možda posledica preprilagođavanja zbog prevelike dubine. Generalno, stablo iz sklearn-a pravi dosta veća stabla (pogotovo za skup Pima) od naših i od BGP stabala. Naša stabla daju veću preciznost na skupu Pima. Stabla iz sklearn-a daju bolju preciznost na skupu Iris od verzije 3 (razlozi za to su već objašnjeni) i na skupu Ionosphere od verzije 2.

Da bi se dao konačan zaključak ovog rada, dalji koraci bi bili eksperimentalno podešavanje parametara za svaki skup pojedinačno, povećavanje broja jedinki u populaciji, povećavanje broja iteracija, izbacivanje listova sa klasama (umesto da algoritam slučajno dodeli klasu na kraju pravila, treba sračunati najbolju klasu za svako pravilo) itd.

Literatura

1. SE Rouwhorst, AP Engelbrecht: "Searching the Forest: Using Decision Trees as Building Blocks for Evolutionary Search in Classification Databases", 2000.
2. Predrag Janičić, Mladen Nikolić: "Veštačka inteligencija", Beograd, 2019.
3. Computational Intelligence - An Introduction, Andries Engelbrecht, John Willey & Sons, 2007.