# Izboljšan operator križanja evolucijskega algoritma za grajenje odločitvenih dreves

**Sašo Karakatič[1], Vili Podgorelec[1]**

[1]*Fakulteta za elektrotehniko, računalništvo in informatiko, Univerza v Mariboru*
*E-pošta: saso.karakatic@um.si, vili.podgorelec@um.si*

## Improved Crossover Operator in Evolutionry Algorithm for Decision Trees

*We propose an innovative crossover operator for evolutionary induced classification trees. Proposed crossover improves the standard random crossover in a way it chooses the crossover location of the trees. Location is chosen based on the accuracy and usage of the tree nodes, so that worst performing subtrees are replaced with the random subtree from the second parent. This crossover process is compared with the standard random crossover on benchmark datasets to evaluate its usefulness.*

## 1 Uvod

Klasifikacija s pomočjo odločitvenih dreves je ena izmed metod strojnega učenja, za katero je značilna enostavna razumljivost tako domenskim strokovnjakom kot tudi laikom [1]. Iz korena drevesa si sledijo pogoji z dvema ali več možnimi izidi, katerim uporabnik sledi vse do lista drevesa, ki pa vsebuje odgovor na klasifikacijsko vprašanje – razred instance. Obstajajo že avtomatizirani postopki za gradnjo klasifikacijskih odločitvenih dreves: ID3 [2], ki deluje samo s številčnimi vrednostmi, in njegova izboljšava C4.5 [3]. Obstajajo pa tudi meta hevristične optimizacijske metode, kot so evolucijski algoritmi, ki se lahko uporabijo za gradnjo klasifikacijskih odločitvenih dreves. Evolucijski algoritem, katerega rezultat je drevo, imenujemo tudi genetsko programiranje. Osnovni subjekt v evolucijskih algoritmih je subjekt, ki predstavlja rešitev problema. Vsi subjekti skupaj gradijo generacijo, ki se skozi evolucijo razvija s ciljem najti čim boljšo rešitev problema. Nad subjekti se v vsaki iteraciji izvedejo določeni genetski operatorji: selekcija, križanje in mutacija [4]. S selekcijo se izberejo posamezniki, ki bodo igrali vlogo staršev v operatorju križanja. Ti posamezniki so izbrani z določeno stopnjo naključnosti, a še vedno igra njihova kvaliteta (fitnes) vlogo pri izbiri. Izbrani posamezniki se nato po parih križajo, da ustvarijo nove subjekte rešitev.

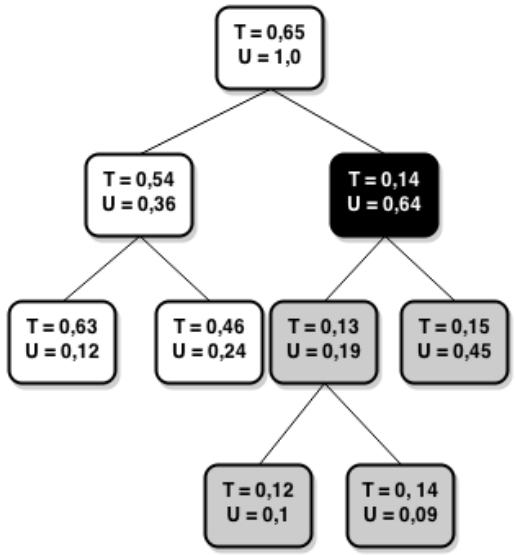
Križanje v genetskih pristopih je zelo odvisno od same reprezentacije subjektov; vedno pa ustvari otroka s kombiniranjem obeh staršev. V našem primeru, kjer je vsak subjekt sestavljen iz drevesa, se otroci kreirajo s kombinacijo dveh dreves – vzamemo drevo enega starša in en del zamenjamo za del iz drevesa drugega starša. Ta menjava poddreves je največkrat narejena naključno. V članku predlagamo nov način križanja, kjer se najprej znebimo slabih delov drevesa z uporabo metrik kot so točnost (ang. accuracy) in uporabljanost (ang. usage) poddreves.

Koza [4] je opisal naključno križanje kot slep operator, ki ne vključuje nobenega konteksta pri izbiri lokacije križanja. Temu je sledil s prispevkom, kjer je predstavil križanje z omejitvijo na samo podobna drevesa [5], s čimer je ohranil del konteksta pri križanju, ampak hkrati uvedel stranski učinek, da so drevesa hitro konvergirala k lokalnemu optimumu in tako omejila prostor iskanja rešitve. Razširitev je poizkušal narediti D'haeseleer, tako da je križal drevesa samo na enaki lokaciji [6]. Podobno metodo, kot je naša predlagana, sta razvila Iba in Garis [7], kjer se poiščejo najslabši deli drevesa in se zamenjajo za najboljši del. Ta pristop ima slabost, da se vsakič kopirajo eni in enaki najboljši deli, kar privede do prehitre konvergence k lokalnemu optimumu, kjer se napredovanje ustavi. Podoben pristop zamenjave najslabšega dela za najboljši del sta preizkušala Hengpraprohm in Chongstitvatana [8], a sta za to uporabljala primerjavo drevesa pred in po končnemu obrezovanju. Tudi tukaj pride do enakega problema prehitre konvergence k lokalnemu optimumu. Zanimiv pristop križanja sta uporabila Majeed in Ryan [9], ki sta iz dveh staršev kreirala več otrok, a sta za napredovanje v novo generacijo izbrala le najboljšega. V prvem staršu sta izbrala več lokacij za križanje, medtem ko so poddrevesa iz drugega starša bila izbrana naključno. Hevristični pristop je izbral tudi Zhang [10], ki je računal »lepljivost« vozlišč, s katero je meril kako dobro posamezna vozlišča kombinirajo med sabo.

V nadaljevanju najprej opišemo operator križanja, ga umestimo v genetski algoritem in predstavimo našo izboljšavo usmerjenega križanja. Sledi opis eksperimenta in dobljeni rezultati. Namen eksperimenta je, da preizkusimo veljavnost in uporabnost naše izboljšave na križanju. To preverimo na več standardnih podatkovnih množicah za klasifikacijo, kjer rezultate podkrepimo tudi s statističnimi metodami. Po interpretaciji rezultatov in diskusiji sledi zaključek prispevka, kjer povzamemo naše delo in naredimo končno razsodbo o uporabnosti predlagane izboljšave.

## 2 Izboljšava operatorja križanja

Križanje je operator, ki kritično vpliva na obliko in kakovost odločitvenih dreves v novo kreiranih subjektih – otrocih, saj mora tvoriti novo drevo tako, da obdrži značilnosti obeh staršev. Prvega starša z operatorjem




Slika 1. Evalvacija vozlišč v odločitvenem drevesu

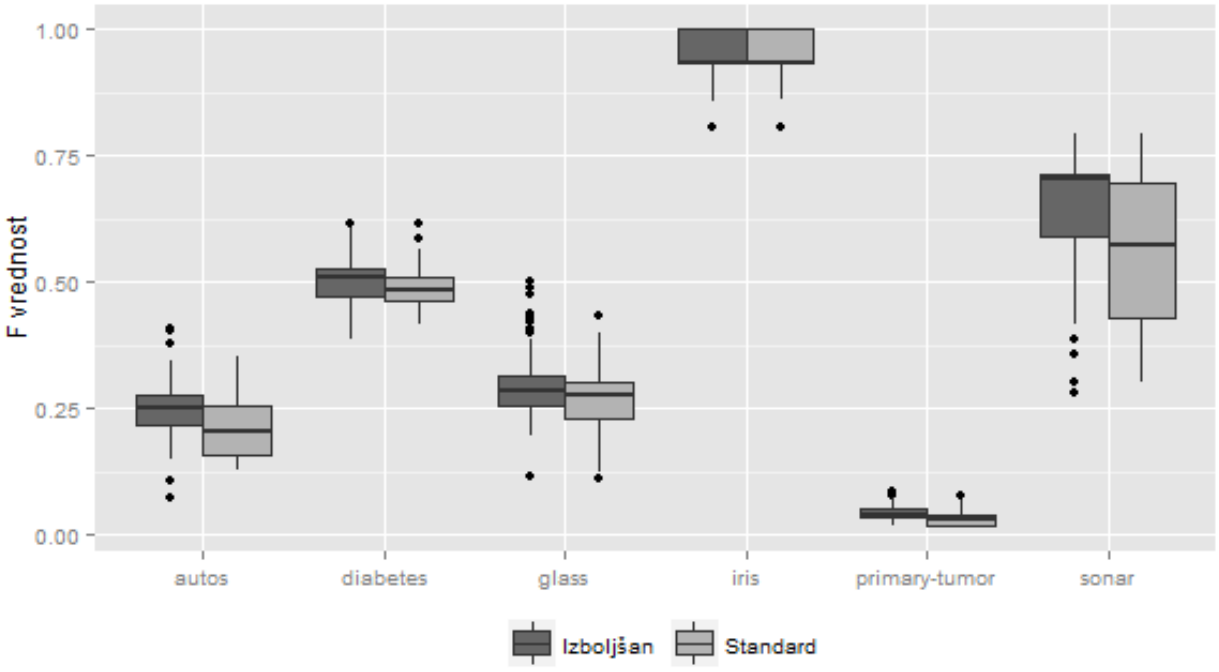
selekcije izberemo, da tvori ogrodje za novega otroka. Standardni pristop križanja narekuje, da naključno izberemo del iz ogrodja in ga odstranimo. Na njegovo mesto prilepimo naključno izbran del drevesa iz drugega starša. Rezultat je novo drevo – nov subjekt, ki bo tvoril novo generacijo v našem procesu skozi evolucijo.

Naš cilj je najti pravo lokacijo križanja in na ta način izboljšati proces tako, da bodo otroci po tem križanju boljši od otrok po standardnem naključnem križanja. Lokacijo križanja smo izbrali s pomočjo naslednje tehnike. Vsak del poddrevesa lahko obravnavamo kot neodvisno odločitveno drevo in tako lahko vsakemu delu poddrevesa posebej izračunamo osnovne metrike klasifikacije. Te metrike se izračunajo samo na podlagi tistih klasifikacijskih instanc, katere so bile v času učenja obdelane v tem poddrevesu. Naša hipoteza je, da lahko na ta način najdemo najslabši del v odločitvenem drevesu in ga zamenjamo v križanju z delom drugega drevesa. S tem procesom eliminacije najslabših delov drevesa upamo, da bodo taka drevesa vsaj tako dobra kot drevesa, kreirana z naključnim križanjem.

Najprej smo za ocenjevanje poddreves uporabili samo točnost klasifikacije na instancah, ki so stekle skozi posamezni del poddrevesa. Ta pristop se ni izkazal za pravega, saj so bili največkrat najslabše ocenjeni kar listi, ki so bili najmanj uporabljeni pri samem procesu. Napake, narejene bližje korenu drevesa, bolj vplivajo na kvaliteto samega drevesa kot pa napake vozlišč proti robu drevesa, kar je privedlo do sledečih sprememb. Poleg same točnosti poddrevesa smo uporabili tudi metriko uporabljanosti poddrevesa, ki pove kolikšen delež vseh klasifikacijskih instanc je ta del poddrevesa obdelal v postopku učenja. Uporabljena funkcija je zapisana v enačbi (1) in je seštevek uporabljanosti poddrevesa in točnosti.

$$e = uporabljanost + ( 1 - točnost ) \qquad (1)$$

Drugi člen enačbe (1-točnost) predstavlja stopnjo napake (ang. error rate). Predpostavili smo, da bomo na ta način še vedno izbrali dele drevesa, ki imajo nizko točnost, ampak bomo preferirali dele drevesa, ki se najbolj uporabljajo in s tem najbolj vplivajo na metrike celotnega drevesa. Ta proces ignorira korensko vozlišče drevesa. Primer ocenitve drevesa je prikazan na sliki 1, kjer je najslabše vozlišče označeno s črno barvo, celotno izbrano poddrevo za zamenjavo pa je obarvano sivo.




Slika 2: Primerjava F vrednosti med dvema vrstama križanja

## 3 Eksperiment

Izvedli smo eksperiment, v katerem smo primerjali dve vrsti križanja: naše izboljšano križanje, ki smo ga opisali v prejšnjem poglavju, in standardno naključno križanje. V eksperimentu je bilo uporabljenih 6 standardnih klasifikacijskih podatkovnih množic, katerih osnovne karakteristike so prikazane v tabeli 1 in tabeli 2. Vsaka množica je bila razdeljena v 10 rezin (ang. fold), da smo izvedli navzkrižno validacijo. Algoritem genetskega programiranja je bil izveden 10 krat na vsaki rezini, kar je prineslo 100 zagonov algoritma na vsaki množici za vsako vrsto križanja – skupaj 1200 zagonov.

Ostali parametri genetskega programiranja so sledeči: algoritem je tekel skozi 2000 generacij s 100% verjetnostjo križanja in 10% verjetnostjo mutacije. Vsaka generacija je vsebovala 150 subjektov (klasifikacijskih odločitvenih dreves), med katerimi je najboljši avtomatsko napredoval v novo generacijo (elitizem velikosti enega subjekta). Selekcijska metoda za izbiro staršev je bila binarna turnirska metoda, kjer izberemo naključno dva posameznika iz generacije v turnir in najboljši v turnirju je izbran za starša pri križanju.

V eksperimentu smo merili metriki točnost (ang. accuracy), specifičnost, senzitivnost, priklic, preciznost in povprečno F vrednost (ang. F-score) vseh razredov. Nad dobljenimi rezultati smo izvedli statistični test Mann-Whitney U neodvisnih vzorcev, saj rezultati niso bili normalno porazdeljeni, da bi ugotovili, če so dobljene razlike med metrikami statistično signifikantne. V tabeli 1 in 2 so dodane p vrednosti testov za vsako množico; rezultati, ki so statistično signifikantni ($p < 0,05$), so zapisani odebeljeno. Povprečna in maksimalna vrednost sta pri boljšem križanju prav tako zapisani odebeljeno.

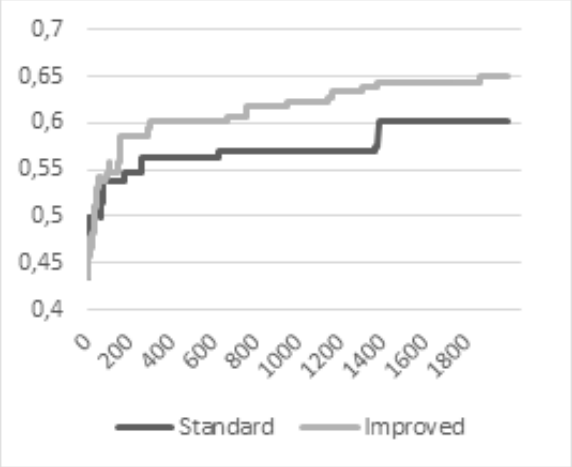
Tabela 1 in tabela 2 povzemata rezultate eksperimenta, iz katerih je razvidno, da naša izboljšava križanja ne samo izenači rezultate, temveč jih tudi preseže v vseh podatkovnih množicah po povprečni vrednosti – tako v točnosti kot v povprečni F vrednosti. V tabeli 1 je narejena primerjava na metriki točnost klasifikacije in kaže, da je izboljšano križanje v povprečju doseglo boljše rezultate v vseh uporabljenih množicah, v kar treh bazah je ta prednost statistično značilna (autos, primary-tumor in sonar). Tabela 2 prikazuje primerjavo križanj na podlagi povprečne F vrednosti vseh razredov. Tukaj je prednost našega križanja še bolj izrazita, saj je prednost statistično značilna v kar petih od šestih množic (autos, glass, diabetes, primary-tumor, sonar). V dveh primerih je naš algoritem našel nasploh boljšo točnost kot je to bilo v primeru naključnega križanja (tabela 1) in v treh primerih je bila izboljšana F vrednost napram naključnemu križanju (tabela 2). Na sliki 2 so prikazane F vrednosti na vseh podatkovnih množicah. Izstopa podatkovna zbirka iris, kjer je v obeh križanjih genetski algoritem deloval zelo dobro in je izrazita izboljšava praktično nemogoča, kar tudi razloži, da prednost našega algoritma na podatkovni zbirki iris ni statistično značilna.

Tabela 2. Primerjava križanja glede na točnost F vrednost.

| Podatkovna zbirka | Število razredov | Povprečje standard | Povprečje izboljšano | p | Maks standard | Maks izboljšano |
|---|---|---|---|---|---|---|
| **autos** | 7 | 0,2069 | **0,2460** | < **0,001** | 0,35 | **0,41** |
| **glass** | 7 | 0,2623 | **0,2913** | < **0,001** | 0,43 | **0,50** |
| **diabetes** | 2 | 0,4899 | **0,5121** | **0,015** | 0,62 | 0,62 |
| **iris** | 3 | 0,9443 | **0,9478** | 0,597 | 1,00 | 1,00 |
| **primary-tumor** | 22 | 0,0321 | **0,0419** | < **0,001** | 0,08 | **0,09** |
| **sonar** | 2 | 0,5501 | **0,6510** | < **0,001** | 0,79 | 0,79 |

Tabela 1. Primerjava križanja glede na točnost klasifikacije.

| Podatkovna zbirka | Število instanc | Povprečje standard | Povprečje izboljšano | p | Maks standard | Maks izboljšano |
|---|---|---|---|---|---|---|
| **autos** | 205 | 0,4705 | **0,5044** | < **0,001** | 0,62 | **0,67** |
| **glass** | 214 | 0,5744 | **0,5814** | 0,136 | 0,73 | **0,77** |
| **diabetes** | 768 | 0,6382 | **0,6444** | 0,789 | 0,70 | 0,70 |
| **iris** | 150 | 0,9447 | **0,9480** | 0,629 | 1,00 | 1,00 |
| **primary-tumor** | 339 | 0,2601 | **0,2741** | **0,007** | 0,38 | 0,38 |
| **sonar** | 208 | 0,6067 | **0,6740** | < **0,001** | 0,80 | 0,80 |




Slika 3. Primerjava točnosti skozi evolucijo

Podrobneje si še poglejmo algoritem skozi evolucijo, da skušamo ugotoviti zakaj doseže izboljšava boljše rezultate od standardnega križanja. Slika 3 prikazuje točnost najboljšega posameznika v generaciji skozi evolucijo. Na vertikalni osi je prikazana točnost, na horizontalni pa zaporedna številka generacije (od 0 do 2000). Iz slike je razvidno, da algoritem s standardnim križanjem doseže lokalni optimum mnogo prej, okoli 1400. generacije in pred tem se že okoli 600. generacije zatakne za približno 800 generacij. Po drugi strani pa algoritem z našim križanjem nadaljuje z izboljševanjem točnosti vse do 1860. generacije, ko doseže svoj optimum. Razlika v točnosti je približno 5% v korist našemu križanju. Tudi če bi izvajanje algoritma ustavili v katerikoli točki pred 2000. generacijo, bi algoritem z izboljšanim križanjem izdelal odločitveno drevo z boljšo točnostjo, saj je prednost prevzel že kar takoj v začetnih generacijah.

## 4 Zaključek

Predstavili smo inovativni način križanja v evolucijskem postopku gradnje odločitvenih dreves za klasifikacijo, ki upošteva točnost poddreves in njihovo uporabljanost za določitev lokacije križanja. Po podrobnejšem opisu metode je sledil eksperimentalni del, kjer smo veljavnost metode preizkusili in jo primerjali s klasičnim naključnim križanjem.

Eksperiment je potekal na več standardnih množicah in je pokazal, da je naša izboljšava doprinesla k izboljšanju končnih rezultatov. Rezultati so bili podkrepljeni s statistično analizo, ki je še dodatno potrdila prednost našega izboljšanega križanja. Na podlagi dobljenih rezultatov lahko pričakujemo, da bi podobne rezultate pridobili še na drugih podatkovnih zbirkah, kar pa seveda zahteva še nadaljnje raziskave v tej smeri.

Dodatno bi lahko raziskali tudi spremembo enačbe za računanje koristnosti vozlišč, ki bi namesto točnosti vsebovale eno izmed drugih metrik klasifikacije, kot na primer specifičnost, senzitivnost, natančnost, priklic ali kar kombinacijo teh z F vrednostjo.

## Literatura

[1] E. Cantu-Paz and C. Kamath, “Inducing oblique decision trees with evolutionary algorithms,” *Evol. Comput. IEEE Trans.*, vol. 7, no. 1, pp. 54–68, 2003.

[2] J. R. Quinlan, “Induction of decision trees,” *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986.

[3] R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann Publishers, 1993.

[4] J. R. Koza, *Genetic Programming: vol. 1, On the programming of computers by means of natural selection*, vol. 1. MIT press, 1992.

[5] J. R. Koza and J. Noyes, *Genetic Programming II Videotape: The Next Generation*, vol. 55. MIT Press Cambridge, MA, 1994.

[6] P. D’haeseleer, “Context preserving crossover in genetic programming,” in *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, 1994, pp. 256–261.

[7] H. Iba and H. de Garis, “Extending genetic programming with recombinative guidance,” *Adv. Genet. Program.*, vol. 2, pp. 69–88, 1996.

[8] S. Hengpraprohm and P. Chongstitvatana, “Selective crossover in genetic programming,” *Population (Paris).*, vol. 400, p. 500, 2001.

[9] H. Majeed and C. Ryan, “Using context-aware crossover to improve the performance of GP,” in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, 2006, pp. 847–854.

[10] M. Zhang, X. Gao, and W. Lou, “A new crossover operator in genetic programming for object classification.,” *IEEE Trans. Syst. Man. Cybern. B. Cybern.*, vol. 37, no. 5, pp. 1332–43, Oct. 2007.