SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Diplomski studij računarstva

Usporedba utrošenog vremena i energije za odabir podskupa značajki kod Firefly i Flower Pollination algoritama

Rijeka, veljača 2023. Ivana Štimac

Sadržaj

1. Uvod

Problem koji se rješava ovim projektom je usporedba utroška vremena i energije dvaju algoritma za proces odabira podskupa značajki te za proces klasifikacije. Promatrani algoritmi su Firefly algoritam i Flower Pollination algoritam. Odabir podskupa značajki je proces izbacivanja redundantnih ili nepotrebnih značajki. Navedeni proces poboljšava performanse klasifikacije, smanjuje prostor pretraživanja te smanjuje šum u podacima. [1]

Potrebno je učitati podatke koji će se koristiti u klasifikaciji, provesti odabir podskupa značajki pomoću algoritma te naposljetku provesti klasifikaciju nad učitanim skupom podataka, ali koristeći samo podskup značajki koji je odabran od strane algoritma. Mjeri se utrošeno vrijeme i energija za provođenje odabira podskupa značajki te za klasifikaciju provedenu nad skupom podataka s odabranim značajkama. Opisani postupak ponavlja se za svaki od algoritama te za tri različita skupa podataka koji imaju različite količine podataka i početni broj značajki. Cilj ovoga projekta je usporediti i zaključiti koji algoritam daje bolje rezultate.

2. Korišteni algoritmi

Algoritmi korišteni za provedbu procesa odabira podskupa značajki pripadaju prirodom inspiriranim algoritmima (eng. nature-inspired algorithms), koji su namijenjeni rješavaju optimizacijskih problema. U ovom projektu korišteni su Firefly i Flower Pollination algoritmi. Provode se kroz određeni broj iteracija, gdje se u svakoj nastoji doći bliže optimalnom rješenju.

2.1 Firefly algoritam

Firefly algoritam temelji se na ponašanju krijesnica u prirodi. Nastoji oponašati uzorak svijetljenja kojim se međusobno privlače. Algoritam se temelji na tri idealizirana pravila [2]:

- Sve krijesnice su istoga spola, što znači da svaka krijesnica privlači druge krijesnice na isti način
- Privlačnost (eng. attractiveness) među krijesnicama je proporcionalna jačini svjetla (eng. brightness) kojim one svijetle, što znači da će se krijesnica koja manje svijetli kretati prema onoj koja svijetli jače. Ako nema krijesnice koja svijetli jače od ostalih, one će se kretati nasumično
- Jačina svijetljenja je definirana funkcijom dobrote

S obzirom na to da privlačnost ovisi o jačini svijetljenja pojedine krijesnice, može se definirati navedeni faktor pomoću udaljenosti:

$$\beta = \beta_0 e^{-\gamma r^2} \tag{2.1}$$

gdje je β faktor privlačnosti, β_0 iznos privlačnosti kada je r = 0, r je udaljenost, a γ faktor apsorpcije svjetla.

Promjena položaja krijesnice i prema krijesnici j, koja više svijetli, definirano je kao:

$$x_i^{t+1} = x_i^t + \beta_0 e^{-\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha\epsilon_i^t \tag{2.2}$$

gdje t označava redni broj iteracije, a r_{ij} udaljenost između dvije promatrane krijesnice. Drugi član ovoga izraza je posljedica navedene privlačnosti, a treći član uvodi određenu količinu nasumičnosti kod promjene položaja krijesnice.

Firefly algoritam ima nekoliko prednosti u odnosu na ostale prirodom inspirirane algoritme. Privlačnost između krijesnica smanjuje se povećanjem udaljenosti što dovodi do podijele populacije u podgrupe, gdje se svaka podgrupa kreće oko svog lokalnog optimuma (eng. automatical subdivision). Ako ima dovoljno krijesnica, podijelit će se u onoliko skupina koliko ima lokalnih optimuma. Od tih lokalnih može se odrediti globalni optimum te zbog toga može riješiti multimodalne probleme (eng. multimodality). Također, privlačnost je nelinearni faktor, a moguće je i podešavanje nasumičnosti kroz iteracije što dovodi do ubrzane konvergencije. Složenost ovoga algoritma je linearna, u najgorem slučaju iznosi $O(n^2t)$, ali ako je n mali (broj jedinki u populaciji, npr. n = 40), a t velik (broj iteracija, npr. t = 5000), složenost postaje O(t). [3]

2.2 Flower Pollination algoritam

Flower Pollination algoritam temelji se na oprašivanju cvijeća u prirodi. U svakom koraku algoritma svaki cvijet je na jedan od dva načina oprašen [4]:

- Globalno oprašivanje: Postoje dvije vrste globalnog oprašivanja, biotičko oprašivanje (eng. biotic pollination) i hibridno oprašivanje (eng. cross-pollination). Biotičko oprašivanje je preko insekata i drugih životinja, a hibridno oprašivanje je između cvijeća dvije različite biljke istog tipa, koje se temelji na nasumičnosti.
- Lokalno oprašivanje: I kod lokalnog oprašivanja postoje dvije vrste, abiotičko oprašivanje (eng. abiotic pollination) i samooprašivanje (eng. self-pollination). Abiotičko oprašivanje je preko vjetra, vode ili gravitacije, a samooprašivanje je oprašivanje između cvijeća iste biljke.

Globalno oprašivanje je unutar algoritma implementirano pomoću sljedećeg izraza:

$$x_i^{t+1} = x_i^t + \gamma \times L(\lambda) \times (g_* - x_i^t) \tag{2.3}$$

gdje je γ faktor skaliranja koji kontrolira veličinu koraka, $L(\lambda)$ Lévyjev let (eng. Levy flight), a g_* je trenutno globalno najbolje rješenje. Lévyjev let se koristi kako bi se oponašao let kukaca na velikim udaljenostima.

Lokalno oprašivanje je unutar algoritma implementirano na sljedeći način:

$$x_i^{t+1} = x_i^t + \epsilon(x_j^t - x_k^t) \tag{2.4}$$

gdje su x_j i x_k cvijeće, odnosno rješenja, iz bliskog okruženja x_i.

Za svaku jedinku u populaciji (unutar svake iteracije) generira se nasumični broj koji se uspoređuje s konstantom P. Konstanta P može iznositi primjerice 0.8, a ako je nasumični broj veći od P, izvršava se globalno oprašivanje, a ako je manji od P, izvršava se lokalno oprašivanje.

Neke od prednosti ovoga algoritma su: mali broj parametara, lagana implementacija, a svejedno visoka učinkovitost [4]. Također, algoritam konvergira eksponencijalno, a zbog hibridnog oprašivanja bježi iz lokalnih optimuma. [5]

3. Metodologija

Za klasifikaciju i odabir podskupa značajki korišten je Python programski jezik [6] te Spyder razvojno okruženje [7]. U programskom kodu korištena je knjižnica otvorenog koda za učenje modela strojnog učenja, Scikit-learn [8], koja nudi različite alate za prilagodbu modela, prethodnu obradu podataka, odabir i evaluaciju modela te mnoge druge alate.

3.1 Skupovi podataka i njihovo učitavanje

Skupovi podataka korišteni za učenje modela strojnog učenja preuzeti su s UCI Machine Learning repozitorija [9]. Korišteno je tri skupa podataka, koji se razlikuju u inicijalnom broju značajki i instanci.

Prva baza podataka su podaci o vinu (*Wine dataset*) koja se sastoji od 13 značajki i 178 redaka što je relativno mali skup podataka i značajki u odnosu na druge korištene skupove podataka. Vina se klasificiraju u tri skupine s obzirom na talijanske regije iz kojih potječu, što znači da se radi o višeklasnoj klasifikaciji. Drugi skup podataka je *Breast Cancer dataset* koji ima 32 značajke te 569 redaka što ga čini srednje velikim skupom podataka. Posljednja baza podataka su podaci o reklamama (*Internet Advertisements dataset*) koji se sastoji od 1558 značajki te 3279 redaka pa je ova baza podataka najveća u odnosu na druge korištene. Za *Breast Cancer* i *Internet Advertisements* skupove podataka koristi se binarna klasifikacija.

Za svaki skup podataka podaci su podijeljeni u pet podskupova (eng. folds), gdje su četiri korištena kao trening set, a jedan kao set za testiranje. Ovaj postupak se ponavlja kroz pet iteracija, gdje se u svakoj iteraciji drugi podskup koristi kao test set. Opisani postupak proveden je pomoću *StratifiedKFold* funkcije preuzete iz Scikit-learn knjižnice, a u ovom slučaju K je jednak 5.

3.2 Odabir podskupa značajki

Za proces odabira podskupa značajki korišteni su algoritmi objašnjeni u prethodnom poglavlju. Programski kod tih algoritama preuzet je s Github repozitorija, odnosno iz knjižnice Niapy [10]. Niapy je okvir za izgradnju algoritama inspiriranih prirodom koji su dio evolucijskog računarstva.

Algoritmi pripadaju u omotač metodu (eng. wrapper method) izbora podskupa značajki pa je evaluacija rezultata prilagođena tako da funkcija dobrote koristi model kojim dobiva ocjenu F1 (eng. F1 score) za svaki podskup značajki. Ocjena F1 bit će objašnjena u sekciji 3.5. Zatim koristi sljedeću formulu za računanje funkcije dobrote: α * (1 - F1_score) + (1 - α) * (veličina podskupa / ukupan broj značajki). α definira koliku važnost dobiva ocjena F1 kod evaluacije odabranog podskupa značajki u odnosu na veličinu odabranog podskupa. Ovdje je α postavljen na 0.99.

Kod inicijalizacije algoritama ostavljene su unaprijed zadane vrijednosti potrebne za izvršavanje algoritma. Kod Firefly algoritma to su: $\alpha = 1$, $\beta_0 = 1$, $\gamma = 0.01$ i $\epsilon = 0.97$, dok vrijednost parametra P kod Flower Pollination algoritma iznosi 0.8. Kod oba algoritma je broj jedinki u populaciji 20. Povećanjem ove vrijednosti na 50 značajno se povećalo vrijeme potrebno za odabir podskupa značajki, dok se promatrani rezultati nisu značajno poboljšali.

3.3 Klasifikacija

Za klasifikaciju podataka korišten je Random Forest klasifikator preuzet također iz Scikit-learn knjižnice. Broj stabala odluke od kojih se sastoji postavljen je na 10. Nakon treniranja modela, pomoću navedenog klasifikatora može se izračunati točnost, ocjena F1 te je moguće prikazati matricu konfuzije (eng. confusion matrix). Sve navedene metrike bit će objašnjene u sekciji 3.5.

3.4 Mjerenje utrošenog vremena i energije

pyRAPL je softver korišten za mjerenje utrošene energije i vremena. [11] Može biti uključen direktno u programski kod i može mjeriti samo one dijelove koda za koje je to

potrebno. Prema tome, moguće je odvojeno izmjeriti koliko je vremena i energije utrošeno za postupak odabira podskupa značajki te za klasifikaciju.

3.5 Metrike

Metrike korištene za evaluaciju rješenja su točnost i ocjena F1 te je korišten grafički prikaz - matrica konfuzije.

3.5.1 Točnost

Točnost (eng. accuracy) je omjer točno klasificiranih instanci i ukupnog broja instanci. Kod binarne klasifikacije točnost je moguće izračunati i preko sljedeće formule:

$$Accuracy = 1 - \frac{FP + FN}{FP + FN + TP + TN} = 1 - ErrorRate \tag{3.1}$$

gdje FP (*False Positive*) označava broj predikcija gdje je klasifikator krivo predvidio negativnu klasu kao pozitivnu, FN (*False Negative*) označava broj predikcija gdje je klasifikator krivo predvidio pozitivnu klasu kao negativnu, TP (*True Positive*) označava broj predikcija gdje je klasifikator točno predvidio pozitivnu klasu kao pozitivnu te TN (*True Negative*) označava broj predikcija gdje je klasifikator točno predvidio negativnu klasu kao negativnu. [12]

Točnost modela moguće je dobiti iz klasifikatora, pozivom funkcije *score()*. Kao parametri šalju se podaci i stvarne klasne tih podataka iz test seta, a može poprimiti vrijednosti između 0 i 1, gdje vrijednost 1 poprima u slučaju kada se svi podaci iz test seta točno klasificiraju.

3.5.2 Ocjena F1

Ocjena F1 računa se pomoću vrijednosti odziva (eng. recall) i preciznosti (eng. precision) [13]:

$$F1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.2}$$

Može poprimiti vrijednosti između 0 i 1, a što je bliže 1, to model manje griješi. Odzivom se izračunava koliki je udio točno predviđenih pozitivnih klasa kao pozitivnih unutar svih klasa koje su pozitivne, odnosno:

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

Visoka vrijednost odziva znači da je model uspio pronaći veliki broj pozitivnih klasa u podacima, iako je možda i neke negativne krivo klasificirao kao pozitivne klase. Preciznost određuje udio točno predviđenih pozitivnih klasa kao pozitivne unutar svih klasa koje su predviđene kao pozitivne, odnosno:

$$Precision = \frac{TP}{TP + FP} \tag{3.4}$$

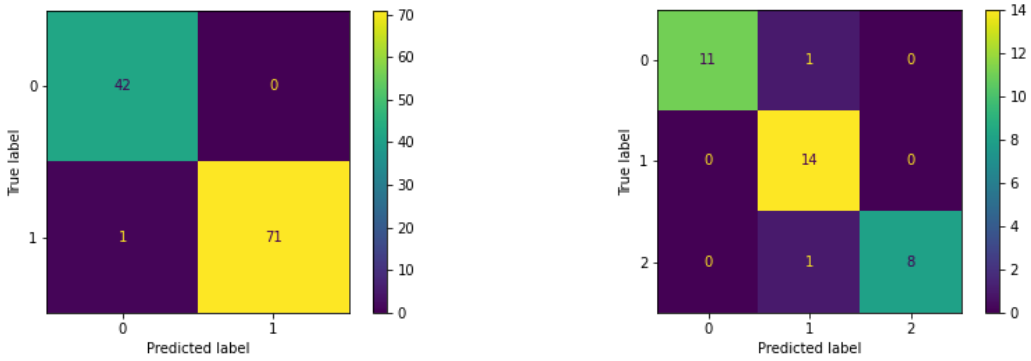
Visoka preciznost znači da, iako možda model nije pronašao sve podatke pozitivne klase, one podatke koje je klasificirao kao pozitivne to vjerojatno i jesu.

Ocjena F1 dobivena je funkcijom *f1_score()* koja je preuzeta iz knjižnice Scikit-learn. Kao parametri šalju se stvarne klase podataka, predviđene klase podataka te parametar *average*. U ovom slučaju, za *average* izabrana je vrijednost *weighted* s obzirom da tako definirana ocjena F1 uzima u obzir situaciju ako su podaci nebalansirani, odnosno ako ima puno više instanci jedne klase nego instanci neke druge klase.

3.5.3 Matrica konfuzije

Matrica konfuzije (eng. confusion matrix) je vizualni prikaz performansa modela. Primjer matrica konfuzije prikazan je na slici 3.1. Na slici 3.1a prikazana je matrica konfuzije kod binarne klasifikacije, a na slici 3.1b prikazana je matrica konfuzije kod višeklasne klasifikacije.

U redovima je prikazano koliko ima stvarnih instanci određenih klasa u podacima, a u stupcima koliko je model predvidio podataka za pojedinu klasu. Iz ovoga prikaza moguće je odrediti vrijednosti TP, TN, FP i FN, iz kojih je onda moguće izračunati točnost i ocjenu F1. Iz binarne matrice, TP, TN, FP i FN vrijednosti moguće je direktno iščitati,



(a) Binarna matrica konfuzije (b) Višeklasna matrica konfuzije

Slika 3.1 Matrice konfuzije

a kod višeklasne matrice, navedene vrijednosti mogu se odrediti principom *jedan protiv svih*. Ovakav princip uzima jednu od klasa i smatra ju kao pozitivnu klasu, a sve ostale zajedno kao negativnu klasu.

Matrica konfuzije prikazana je pozivanje funkcije *ConfusionMatrixDisplay.from_ estimator()* iz knjižnice Scikit-learn. Kao parametri su poslani klasifikator te podaci i stvarne klasne tih podataka iz test seta.

4. Rezultati

U programskom kodu korištena je k-dijelna unakrsna provjera (eng. k-fold cross validation), koja je objašnjena u poglavlju 3.1., gdje k iznosi pet, a svako je mjerenje provedeno tri puta. U tablicama 4.1, 4.2 i 4.3 prikazano je prosječno izmjereno vrijeme, izraženo u sekundama, te pripadajuća standardna devijacija i prosječna izmjerena energija, izražena u džulima, s pripadajućom standardnom devijacijom, za proces odabira podskupa značajki te za klasifikaciju s odabranim podskupom značajki. U pojedinoj tablici prikazana su mjerenja za oba algoritma, Flower algoritam (FA) i Flower Pollination algoritam (FPA), a s obzirom da je mjerenje provedeno za tri različita skupa podataka, rezultati su prikazani u tri odvojene tablice.

Tablica 4.1 Mjerenja za *Wine dataset*

Wine Dataset	Vrijeme (s)	St. dev. vremena (s)	Energija (J)	St. dev energije (J)
FA - odabir značajki	6.2247	0.6351	65.6027	3.0366
FPA - odabir značajki	1.0193	0.4512	7.9950	0.5465
FA - klasifikacija	0.013	0.0001	0.1242	0.0056
FPA - klasifikacija	0.017	0.0077	0.1732	0.0808

Tablica 4.2 Mjerenja za *Cancer dataset*

Cancer Dataset	Vrijeme (s)	St. dev. vremena (s)	Energija (J)	St. dev energije (J)
FA - odabir značajki	6.8065	0.9493	76.4368	6.3003
FPA - odabir značajki	1.1551	0.5143	11.4182	1.4022
FA - klasifikacija	0.0156	0.0005	0.1566	0.0195
FPA - klasifikacija	0.0192	0.0046	0.1777	0.0369

Tablica 4.3 Mjerenja za *Ads dataset*

Ads Dataset	Vrijeme (s)	St. dev. vremena (s)	Energija (J)	St. dev energije (J)
FA - odabir značajki	400.4178	15.8866	1379.7237	106.2416
FPA - odabir značajki	44.5598	1.8405	160.6214	16.0279
FA - klasifikacija	0.1362	0.0177	0.5430	0.0514
FPA - klasifikacija	0.1431	0.0139	0.6671	0.0608

U tablicama 4.4, 4.5 i 4.6 prikazan je broj značajki u odabranom podskupu nakon procesa odabira podskupa značajki te točnost i ocjena F1 nakon provedene klasifikacije podataka s odabranim podskupom značajki. Prikazane su tri odvojene tablice, svaka za jedan od skupova podataka, a u pojedinoj tablici prikazani su rezultati za oba algoritma. Početni broj značajki za *Wine Dataset* je 13, za *Cancer Dataset* je 32, a za *Ads Dataset* on iznosi 1558 značajki.

Tablica 4.4 Broj značajki te iznosi metrika za *Wine dataset*

Wine Dataset	Broj značajki nakon FS-a	Točnost	Ocjena F1
FA	6	0.8997	0.8988
FPA	7	0.9330	0.9303

Tablica 4.5 Broj značajki te iznosi metrika za *Cancer dataset*

Cancer Dataset	Broj značajki nakon FS-a	Točnost	Ocjena F1
FA	12	0.9349	0.9349
FPA	13	0.9491	0.9489

Tablica 4.6 Broj značajki te iznosi metrika za *Ads dataset*

Ads Dataset	Broj značajki nakon FS-a	Točnost	Ocjena F1
FA	773	0.9573	0.9554
FPA	791	0.9579	0.9559

Iz rezultata je vidljivo kako je Flower Pollination algoritam primijenjen za proces odabira podskupa značajki približno šest puta brži od Firefly algoritma za mali i srednje velik skup podataka, a za najveći skup podataka je 9 puta brži. Za isti proces Flower Pollination algoritam utroši 7 do 9 puta manje energije, ovisno o veličini skupa podataka, od Firefly algoritma.

Vrijeme potrebno za klasifikaciju podataka s odabranim podskupom značajki manje je za 24% (*Wine Dataset*), 19% (*Cancer Dataset*), odnosno 5% (*Ads Dataset*), ako je odabir podskupa značajki proveden Firefly algoritmom. Utrošena energija je također manja ako je odabir podskupa značajki proveden Firefly algoritmom, a manja je za 28% kod *Wine Dataseta*, 12% kod *Cancer Dataseta* te 19% za *Ads Dataset*. Navedeni rezultati mogu se povezati s brojem odabranih značajki korištenih u klasifikaciji. Firefly algoritam tijekom procesa odabira značajki odabere manji podskup značajki od Flower Pollination algoritma, u slučaju *Wine* i *Cancer Dataseta* za jednu značajku manje, a u slučaju *Ads Dataseta* 18 značajki manje, zbog čega je manje potrebno vremena i energije za proces klasifikacije.

Međutim, kod klasifikacije podskupom značajki odabranih od strane Firefly algoritma dobivena točnost i ocjena F1 manji su nego kod klasifikacije podskupom značajki odabranih od strane Flower Pollination algoritma. Kod najmanjeg skupa podataka ta razlika iznosi 0.0333 za točnost i 0.0315 za ocjenu F1, kod srednje velikog skupa podataka razlika je 0.0142 za točnost i 0.014 za ocjenu F1, a za najveći skup podataka točnost se razlikuje za 0.0006, a ocjena F1 za 0.0005.

5. Zaključak

Glavni cilj ovoga projekta bio je usporediti performanse, odnosno izmjeriti utrošeno vrijeme i energiju, Firefly i Flower Pollination algoritama na procesima odabira podskupa značajki te klasifikaciji podataka samo s odabranim podskupom značajki.

Iz dobivenih rezultata vidljivo je kako Flower Pollination algoritam troši puno manje vremena i energije od Firefly algoritma, čak i do 9 puta, a pritom ostvaruje veću točnost i ocjenu F1. S druge strane, Firefly algoritam odabire nešto manji podskup značajki od Flower Pollination algoritma.

S obzirom na rezultate, može se zaključiti kako je za proces odabira podskupa značajki u većini slučajeva bolje koristiti Flower Pollination algoritam. U slučaju kada je glavni cilj dobiti što manji podskup značajki, Firefly algoritam je bolji izbor. Također, može se primijetiti kako se razlika u točnosti i ocjeni F1 smanjuje s povećanjem broja značajki skupa podataka, što bi bilo zanimljivo za proučiti u nekim budućim istraživanjima.

Bibliografija

[1] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach.Learn. Res. 3 (2003) 1157–1182.

[2] I. Fister, X.-S. Yang, A Brief Review of Expanding Literature, Springer International Publishing Switzerland, 2014

[3] X.-S. Yang, X. He, Firefly Algorithm: Recent Advances and Applications, Int. J. Swarm Intelligence, 2013, Vol. 1, No. 1, pp. 36–50.

[4] E. Nabil, A Modified Flower Pollination Algorithm for Global Optimization, 2015

[5] X.-S. Yang, Flower pollination algorithm for global optimization, in: Unconventional Computation and Natural Computation 2012, Lecture Notes in Computer Science, Vol. 7445, pp. 240-249, 2012

[6] "Python", s Interneta, `https://www.python.org/`, 29. siječnja 2023.

[7] "Spyder", s Interneta, `https://www.spyder.com/`, 29. siječnja 2023.

[8] "Scikit-learn", s Interneta, `https://scikit-learn.org/stable/`, 29. siječnja 2023.

[9] "UCI Machine Learning Repository", s Interneta, `https://archive.ics.uci.edu/ml/index.php`, 29.siječnja 2023.

[10] "NiaPy knjižnica", s Interneta, `https://github.com/NiaOrg/NiaPy`, 29.siječnja 2023.

[11] "PyRAPL softver", s Interneta, `https://pyrapl.readthedocs.io/en/latest/`, 29.siječnja 2023.

[12] "Točnost", s Interneta, `https://en.wikipedia.org/wiki/Accuracy_and_precision`, 29. siječnja 2023.

[13] "Ocjena F1", s Interneta, `https://towardsdatascience.com/the-f1-score-bec2bbc38aa6`, 29. siječnja 2023.