Potraga za pulsarima primjenom strojnog učenja

Bilić, Marin

Master's thesis / Diplomski rad

2020

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Science / Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:217:327347

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-05-13**

Repository / Repozitorij:

Repository of the Faculty of Science - University of Zagreb

zir.nsk.hr

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
FIZIČKI ODSJEK

Marin Bilić

POTRAGA ZA PULSARIMA PRIMJENOM STROJNOG UČENJA

Diplomski rad

Zagreb, 2020.

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
FIZIČKI ODSJEK

INTEGRIRANI PREDDIPLOMSKI I DIPLOMSKI SVEUČILIŠNI
STUDIJ FIZIKA I INFORMATIKA; SMJER NASTAVNIČKI

Marin Bilić

Diplomski rad

Potraga za pulsarima primjenom strojnog učenja

Voditelj diplomskog rada: izv. prof. dr. sc. Goranka Bilalbegović

Ocjena diplomskog rada: _____

Povjerenstvo: 1. _____

2. _____

3. _____

Datum polaganja: _____

Zagreb, 2020.

Zahvaljujem mentorici koja mi je svojom dostupnošću i idejama uvelike olakšala i omogućila izradu diplomskog rada. Također, htio bih zahvaliti svojoj obitelji, ponajviše svojim roditeljima koji su uvijek bili uz mene i omogućili mi sve što je jednom studentu potrebno. Zahvaljujem se i svojim kolegama i prijateljima koji su mi uljepšali čitavo razdoblje studiranja.

Sažetak

Pulsari su brzo rotirajuće neutronske zvijezde. Ti vrlo zanimljivi objekti fasciniraju astronome još od trenutka kad su prvi put pronađeni. S vremenom su teleskopi postajali sve bolji te se tako i broj otkrivenih pulsara povećavao. Ručno analiziranje podataka dobivenih pomoću teleskopa u potrazi za pulsarima je vrlo naporan i monoton proces koji zahtjeva puno vremena. Međutim, razvojem strojnog učenja je ovaj zadatak postao rješiv pomoću računala. U ovom diplomskom radu analizirani su pripremljeni podaci s kandidatima za pulsare. Njihovom obradom pomoću tri različita algoritma strojnog učenja razvijeni su modeli kategorizacije koji daju veliku preciznost. Korišteni su algoritmi k-najbližih susjeda, potpornih vektora i stabla odlučivanja. Ti algoritmi su analizirani i testirani za više vrijednosti parametara kako bi se dobila najbolja rješenja. Uspoređeni su rezultati primijenjenih algoritama. U metodičkom dijelu je dan osvrt na teme iz astronomije i pulsara u osnovnim i srednjim školama, te je razrađen mogući uvod u strojno učenje za srednjoškolsku nastavu.

Ključne riječi: pulsari, strojno učenje, algoritam k-najbližih susjeda, algoritam potpornih vektora, algoritam stabla odlučivanja

Search for pulsars using machine learning

Abstract

Pulsars are neutron stars with fast rotation speeds. These very interesting objects fascinate astronomers from the moment they were discovered. With time, telescopes are becoming better, therefore the number of discovered pulsars is growing. Analyzing the telescope data by hand is a very tedious and monotone process. However, with development of machine learning this task became easily solvable by using a computer. In this diploma thesis we analyze the prepared data with pulsar candidates. Three different machine learning algorithms are applied and they produced prediction models with high accuracy. Algorithms applied are: k-nearest neighbors, support vectors and decision tree. All algorithms are analyzed and tested for several values of parameters in order to produce the best solution. Results of applied algorithms are compared. In the methodical section, a summary of topics in astronomy and pulsars for middle and high school students is given. In addition, a possible introduction to machine learning for high school students is formulated.

Keywords: pulsars, machine learning, K-nearest neighbors algorithm, support vectors algorithm, decision tree algorithm

Sadržaj

1 Uvod

Jocelyn Bell Burnell je 1967. godine primjetila čudan uzorak na kilometrima dugoj traci na koju je teleskop zapisivao podatke. U tom trenutku je znala da je pronašla nešto novo. Kasnije se pokazalo da je otkrila je novu vrstu nebeskih tijela - pulsare [1]. Astronomi su nastavili pretraživati nebo i tražiti nove pulsare. Zapisi teleskopa su postajali kvalitetniji s vremenom, ali je posao analize tih podataka ostao dugotrajan i zahtjevan proces koji je uzimao mnogo vremena astronomima i njihovim asistentima. Razvojem računala postalo je očito da će mnogi analitički poslovi koji su ljudima vremenski zahtjevni postajati vrlo brzo izvedivi, te da će olakšati analizu podataka. Ipak, za analizu podataka o pulsarima potrebno je obraditi veliku količinu podataka koje teleskopi generiraju. Rješenje se razvilo u obliku sada već posebne grane računalne znanosti - znanosti o obradi velikih količina podataka (engl. *big data*). Ovo je područje eksplodiralo u popularnosti tijekom 21. stoljeća, te su se razvili mnogi algoritmi, alati, čak i posebni programski jezici koji omogućuju lako pristupačnu obradu velike količine podataka [2]. Ti novi algoritmi i alati su se počeli koristiti u komercijalnim aplikacijama i znanstvenim istraživanjima. Jedna od primjena je obrada teleskopskih podataka. U ovom radu analiziramo podatke o potencijalnim kandidatima za pulsare pomoću tri različita algoritma strojnog učenja. Cilj je postizanje što veće točnosti u prepoznavanju pulsara.

U drugom poglavlju ovog rada su opisani pulsari i proces njihovog otkrića. U trećem poglavlju je prikazano nadzirano strojno učenje, te princip rada tri tipa klasifikacijskih algoritama koja se često koriste kod obrade velikih količina podataka. U četvrtom poglavlju je opisana baza podataka koja je korištena za učenje modela, te su algoritmi primijenjeni na te podatke o pulsarima. Algoritmi su analizirani i izabrani su parametri na način da daju najbolje rezultate, te su na kraju uspoređeni kako bi odredili koji je algoritam najbolje koristiti. U dodacima su navedene teme iz astronomije i strojnog učenja koje bi se mogle obraditi u srednjim školama kao zanimljivo dodatno gradivo.

2 Pulsari

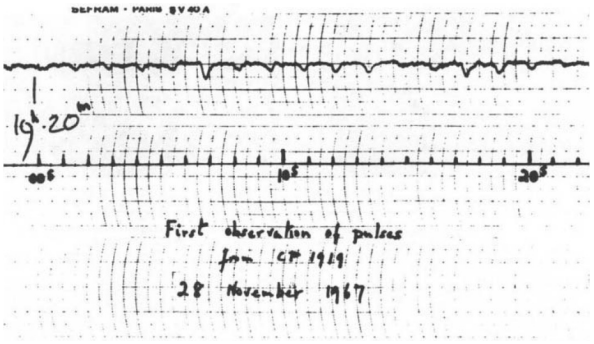
2.1 Otkriće pulsara

Supernova je eksplozija do koje dolazi u zadnjim stadijima evolucije nekih zvijezda [3]. Nakon kolapsa početne zvijezde može nastati neutronska zvijezda. Walter Baade i Fritz Zwicky su predložili prvu teoriju o neutronskim zvijezdama [4]. Sredinom prošlog stoljeća počela se razvijati radio astronomija. Kvazari su bili su jedno od prvih većih otkrića u radio astronomiji [5]. Ime dolazi iz riječi "quasi-stellar", što znači kvazi-zvijezda, jer su astronomi u trenutku otkrića mislili da se radi o zvijezdama [6]. Danas se zna da su kvazari najsjajnije aktivne galaktičke jezgre u kojima se nalaze vrlo masivne crne rupe [5].

Nakon prvog otkrića znanstvenici su organizirali projekte potrage za novim kvazarima. Antony Hewish i Martin Ryle sa Sveučilišta u Cambridgeu su postavili niz od 4096 dipolnih antena koje su radile na radio frekvencijama od 81 Hz (Slika 2.1). Postav se prostirao na $18000\ m^2$, te je mnogim mještanima više izgledao kao vinograd, a ne kao radio teleskop [7].

Slika 2.1: Radio teleskop korišten za otkriće pulsara [1]

Antony Hewish je kao asistenticu zaposlio Jocelyn Bell Burnell, koja je svaki dan odlazila do postava i skupljala kilometrima dugačke zapise teleskopa. Analizirala ih je u nadi da će pronaći još kvazara. Jocelyn je u srpnju 1967. godine prvi put primijetila čudan uzorak koji se pojavio na jednom od zapisa (Slika 2.2). Uzorak je bio specifičan po tome što je prikazivao jednako razmaknute pulseve u vrlo kratkim vremenskim intervalima. To je bila neobjašnjiva pojava u tom trenutku jer se nijedno poznato nebesko tijelo nije ponašalo na takav način. Prva sumnja je bila da se radi o interferenciji od strane nekog izvora sa Zemlje. Međutim, ta se sumnja vrlo brzo pokazala nelogičnom, jer se signal pojavljivao sa iste lokacije na nebu i to točno svakih 23 sata i 54 minute, što odgovara zvjezdanom danu, odnosno periodu rotacije planeta u odnosu na ostale zvijezde [1].



Slika 2.2: Traka koja sadrži zapis o CP1919 pulsaru. Vidljivi su pravilno razmaknuti pulsevi [1].

Hewishova prva ideja je bila da pravilno razmaknuti pulsevi predstavljaju pokušaj vanzemaljaca da uspostave kontakt s nama. Zbog toga je prvo izvor nazvao ”LGM-1”, što je kratica za ”Little Green Men” (”mali zeleni ljudi”) [8]. Nedugo nakon prvog otkrivena su još tri izvora s istim oblikom pulseva. Rezultate su objavili široj znanstvenoj zajednici, bez obzira što još uvijek nije postojalo logično objašnjenje za njih [1]. Već sljedeće godine, Thomas Gold je dao objašnjenje za ponašanje ovih novih izvora. Zaključio je da se radi o neutronskim zvijezdama koje imaju jako magnetsko polje i rotiraju velikim kutnim brzinama, što uzrokuje primjećene pravilne periode. Neutronska zvijezda se tada ponaša ”kao svjetionik”, tako da se na Zemlji signal primijeti svaki put kada se svjetlost emitira prema njoj [9].

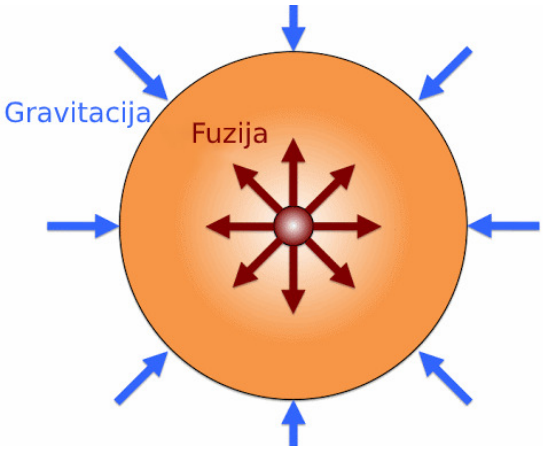
Antony Hewish i Martin Ryle su 1974. godine dobili Nobelovu nagradu iz fizike za otkriće pulsara [10]. Dodjela ove nagrade je uzrokovala nelagodu među nekim članovima znanstvenog društva. Nagrada nije bila dodijeljena i Jocelyn Bell Burnell, koja je prva otkrila signal. Ona je u periodu ovog otkrića radila na doktoratu te je bila asistentica A. Hewisha [11].

2.2 O pulsarima

Pulsari su jako magnetizirane neutronske zvijezde koje rotiraju i pritom emitiraju elektromagnetsku radijaciju [12]. Kako bi u potpunosti shvatili zašto se tako ponašaju te koja su im svojstva, moramo prvo razmotriti kako nastaju.

Svaka zvijezda mora proći kroz određeni životni ciklus. Stabilne zvijezde su takve zbog ravnoteže između radijacijske i gravitacijske sile u njima (Slika 2.3). Zvijezde, zbog svoje velike mase, gravitacijskom silom privlače sav materijal od kojeg su građene prema unutra [14]. Ova sila je toliko jaka da se jezgre vodika u središtu zvijezde ne mogu slobodno

gibati te se spajaju u teže jezgre procesom fuzije. Vodik se spaja s vodikom i nastaje deuterij. Fuzijom tricija s deuterijem nastaje helij. Ovaj proces nuklearne fuzije oslobađa velike količine energije koja pokušava izbiti atome van te se na taj način radijacijska sila suprotstavlja gravitacijskoj [15].

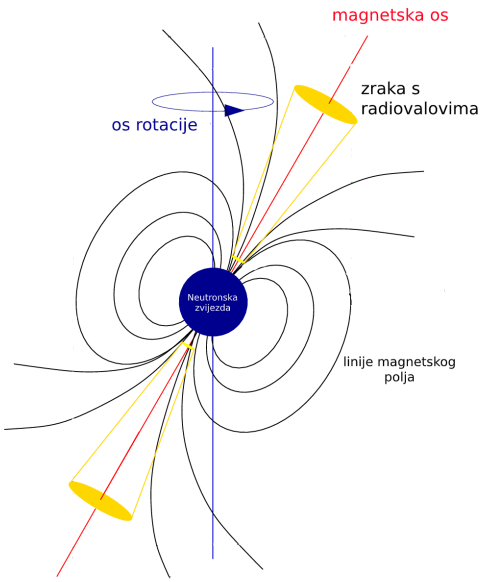


Slika 2.3: Ravnoteža gravitacijske i radijacijske sile u zvijezdama [13]

Kad su ovi procesi u ravnoteži, zvijezda je stabilna. Primjer takve zvijezde je i naše Sunce. Međutim, s vremenom zvijezda potroši sav vodik koji se nalazi u njoj, te nakon toga slijedi sagorijevanje helija u teže elemente poput ugljika i kisika. Zvijezda poput Sunca će se završetkom tog procesa pretvoriti u crvenog diva, te će se raspasti ostavljajući iza sebe ostatak kojeg se naziva bijeli patuljak [16].

Zvijezde čija je masa osam ili više puta veća od Sunca imaju različitu evoluciju. Takva zvijezda se ne zaustavlja kod sagorijevanja ugljika, već nastavlja sa sagorijevanjem neona u kisik, kisika u silicij, te za kraj silicija u željezo. Kada dođe do stadija formiranja željezne jezgre unutar zvijezde, ravnoteža radijacijske i gravitacijske sile prestaje jer željezo ne ulazi u proces nuklearne fuzije [17]. Nakon formiranja željezne jezgre velika masa zvijezde odjednom implodira, odnosno skuplja se u svoje središte uz ogromnu silu na već formiranu željeznu jezgru. Ta sila je toliko jaka da uspije približiti elektrone atomskoj jezgri, te se oni neutraliziraju s protonima. Ono što na kraju ostane su sami neutroni, nukleoni bez naboja koji su gusto rasporedjeni jedan uz drugog. Cijela zvijezda se uruši u svoje središte te dolazi do formiranja supernove - eksplozije koja izbaci 95% materije zvijezde van u svemir. Ostalih 5% ostaje u centru i tvori neutronsku zvijezdu [18]. Gustoća neutronske zvijezde je toliko velika, da bi se masa cijelog planeta Zemlje trebala stisnuti u veličinu prosječnog grada kako bi se postigla ista vrijednost gustoće. Još jedna popularna usporedba je da bi masa svih ljudi na Zemlji stala u volumen od $1\ cm^3$ [19].

Ovaj proces raspada originalne zvijezde je bitan za objašnjenje jakog magnetskog polja kod pulsara, a i visoke brzine rotacije. Razlozi su zakon očuvanja magnetskog toka i zakon očuvanja kutne količine gibanja [20]. Zvijezda je prije supernove imala određeni magnetski tok određen formulom $\Phi = \int \vec{B} \cdot \vec{n} da$. Kada se polumjer jezgre zvijezde smanji za $\sim 10^5$, zbog očuvanja magnetskog toka, jakost magnetskog polja poraste za $\sim 10^{10}$ jer površina po kojoj integriramo ima kvadratnu ovisnost o polumjeru. Magnetsko polje na površini neutronskih zvijezda je (10^4 -10^{11}) T. Mlade neutronske zvijezde imaju vrlo jaka magnetska polja, ali s vremenom jakost opada. Rezultantni oblik polja je magnetski dipol, kao i kod Zemlje. Također, kao i kod Zemlje, os rotacije i magnetska os se ne poklapaju već je magnetska os nagnuta pod nekim kutem α (Slika 2.4).



Slika 2.4: Skica neutronske zvijezde i njenog magnetskog polja [20]

Što se tiče velike brzine rotacije pulsara oko svoje osi, ona proizlazi iz očuvanja kutne količine gibanja. Željezna jezgra prije supernove ima kutnu količinu gibanja

$$L = I\omega = \frac{2}{5} M R^2 \omega. \tag{2.1}$$

Njena kutna količina gibanja nakon supernove mora ostati ista. Polumjer jezgre se jako smanjuje kao što je već navedeno, što uzrokuje veliki rast kutne brzine ω.

Brza rotacija magnetskog polja pulsara uzrokuje elektromagnetsku radijaciju, koja je glavni potrošač energije pulsara. Snaga elektromagnetskog zračenja rotirajućeg magnetskog

dipola je dana formulom [20]

$$P_{\text{rad}} = \frac{2}{3}\frac{(\ddot{m}_{\perp})^2}{c^3}, \tag{2.2}$$

gdje je $m_{\perp}$ okomita komponenta magnetskog dipolnog momenta. Ako pulsar aproksimiramo kao sferu radijusa R čija je jakost magnetskog polja B, tada je iznos magnetskog dipolnog momenta

$$m = BR^3, \tag{2.3}$$

a okomita komponenta je

$$m_{\perp} = BR^3 \sin\alpha. \tag{2.4}$$

Za pulsar koji rotira kutnom brzinom ω vrijedi

$$m = m_0 \exp(-i\omega t)$$

$$\dot{m} = -i\omega m_0 \exp(-i\omega t)$$

$$\ddot{m} = \omega^2 m_0 \exp(-i\omega t) = \omega^2 m. \tag{2.5}$$

Označimo li period rotacije pulsara sa T, te uvrstimo izraze (2.4) i (2.5) u (2.2), konačan izraz za snagu zračenja rotirajućeg magnetskog dipola je

$$P_{\text{rad}} = \frac{2}{3}\frac{m_{\perp}^2\omega^4}{c^3} = \frac{2}{3c^3}(BR^3\sin\alpha)^2\left(\frac{2\pi}{T}\right)^4. \tag{2.6}$$

Snaga elektromagnetskog zračenja je dovoljno velika da utječe na ukupnu energiju pulsara, što kao posljedicu ima usporavanje njegove rotacije. To znači da period pulsara T nije konstantan u vremenu već se mijenja, dakle $\dot{T} > 0$. Vezu između perioda i gubitka rotacijske energije možemo lako dobiti. Izrazi za kutnu brzinu i promjenu kutne brzine su

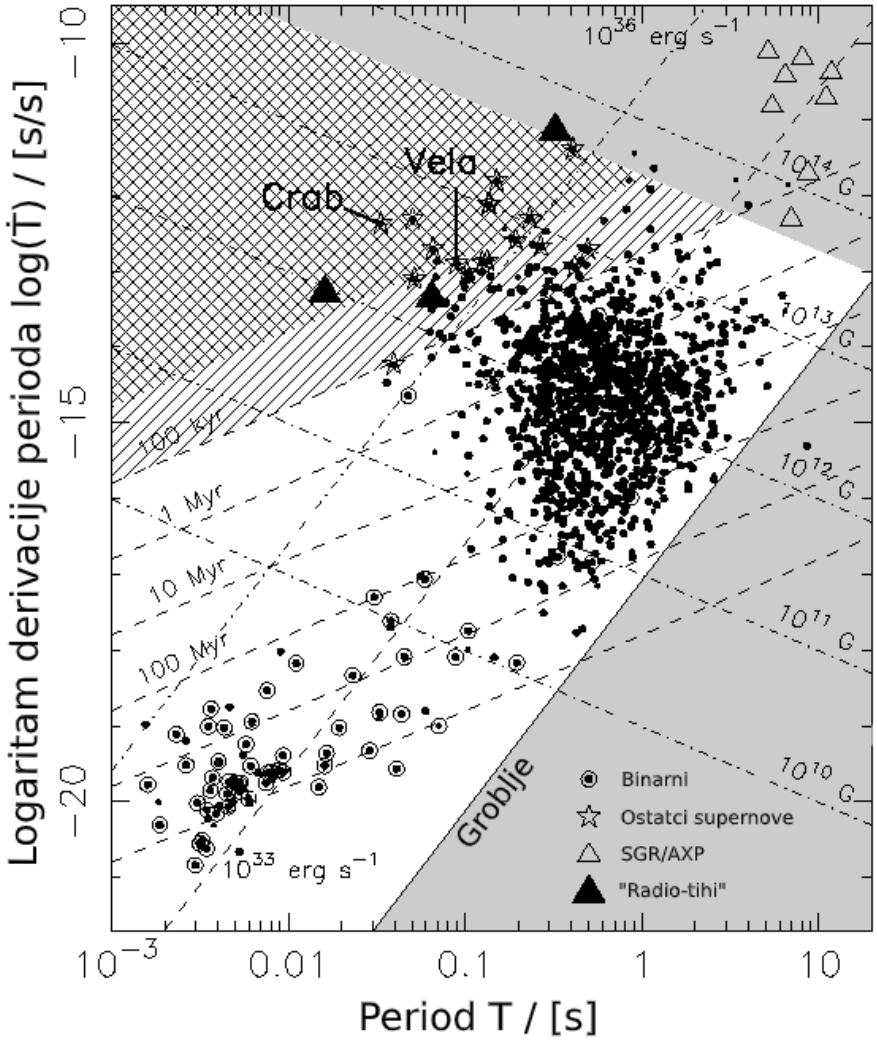
$$\omega = \frac{2\pi}{T}, \qquad \text{dakle} \qquad \dot{\omega} = 2\pi\left(-\frac{\dot{T}}{T^2}\right). \tag{2.7}$$

Kada ovo uvrstimo u izraz za promjenu rotacijske energije dobijemo

$$\frac{dE_{\text{rot}}}{dt} = \frac{d}{dt}\left(\frac{1}{2}I\omega^2\right) = I\omega\dot{\omega} = \frac{-4\pi^2 I\dot{T}}{T^3} \tag{2.8}$$

Ovaj konačan rezultat nam je bitan zbog toga što se sa Zemlje mogu dobiti jako precizna

mjerenja za T i $\dot{T}$ [20, 21]. Upravo se po tim mjerenjima pulsari kategoriziraju. Postoje



Slika 2.5: Kategorizacijski T-$\log \dot{T}$ graf za pulsare [21]

tri glavne kategorije pulsara, a to su: magnetari, rotacijski pogonjeni pulasari i akrecijski pogonjeni pulsari [22].

- Magnetari - najmlađi pulsari s ekstremno jakim magnetskim poljem. Većina njihovog elektromagnetskog zračenja dolazi iz raspadanja magnetskog polja, a ne iz rotacije. Na Slici 2.5 se nalaze gore lijevo i označeni su zvjezdicama. Oko njih se još uvijek nalaze ostatci supernove.

- Rotacijski pogonjeni pulsari - većina do sad otkrivenih pulsara. Većina elektromagnetskog radio zračenja dolazi iz gubitka njihove rotacijske energije. Na Slici 2.5 se nalaze u sredini i označeni su crnim točkama.

- Akrecijski pogonjeni pulsari - pulsari koji energiju za emitiranje rendgenskih valova

dobivaju od susjedne zvijezde koja je crveni div. Ti pulsari svojim gravitacijskim poljem privlače materiju susjedne zvijezde i tako stvaraju akrecijski disk koji daje dodatnu energiju pulsaru. To su pulsari koji su pred krajem svog postojanja, a na Slici 2.5 se nalaze dolje lijevo te su označeni dodatnim krugom.

Na Slici 2.5 su također prikazane dodatne kategorizacije. Sivom bojom je označeno "groblje". To je područje u kojem pulsari teorijski ne bi smjeli postojati, međutim neispunjenim trokutom su prikazani pulsari koji su ipak otkriveni u tom području, a nazivaju se "Soft Gamma Repeaters" (SGR) i "Anomalous X-Ray Pulsars" (AXP). Jednostruko isprekidanim linijama je označeno područje pulsara koji su stari (10-100) kyr i nalikuju Vela pulsaru. Vela pulsar je ostatak Vela supernove i emitira u radio, optičkom, rendgenskom i γ području elektromagnetskog spektra [1]. Dvostruko isprekidanim linijama je označeno područje pulsara koji nalikuju Crab pulsaru. Crab pulsar je ostatak supernove čija se eksplozija 1054. godine vidjela za Zemlje [1]. Također, ispunjenim trokutom su na Slici 2.5 prikazani pulsari koji ne emitiraju radio valove, a teorijski bi trebali [21].

2.3 Potraga za pulsarima

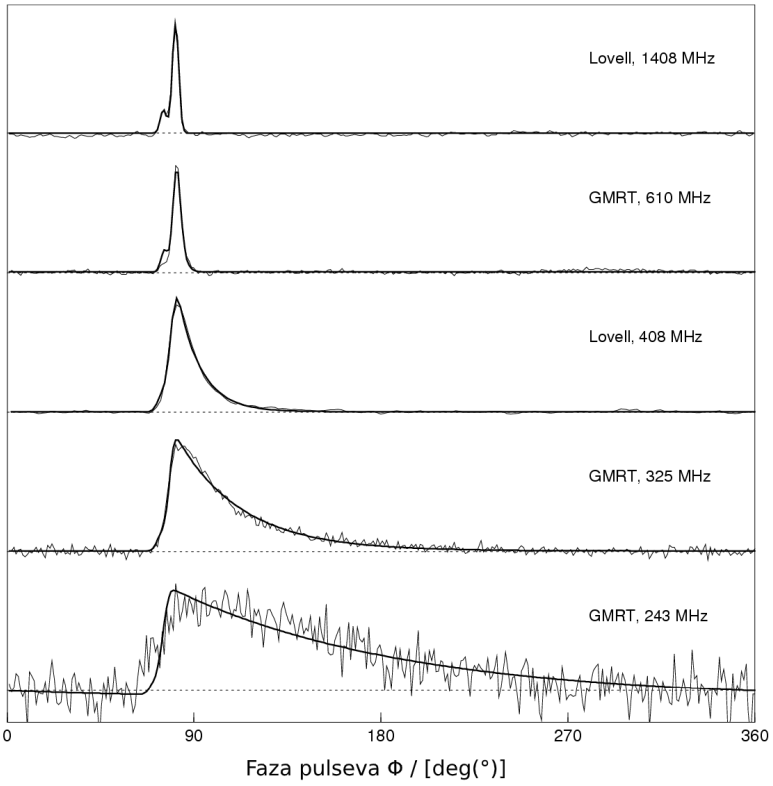
Nakon otkrića 1967. godine, uslijedila su istraživanja u kojima su otkrivani novi pulsari. Prvi binarni pulsar je otkriven 1974. godine. To je pulsar koji kruži oko druge neutronske zvijezde [23]. Nedugo nakon toga, 1982. godine, je otkriven prvi milisekundni pulsar. Period rotacije tog pulsara je samo 1.6 milisekunda [24]. Do danas je otkriveno nešto više od 2800 pulsara [25, 26]. Ono što se kroz vrijeme mijenjalo su rasponi frekvencija valova koji su korišteni za opažanje pulsara, kao i sve kvalitetniji teleskopi. Također, razvojem računala su se razvijale metode za analizu prikupljenih podataka. U početku su znanstvenici gledali određene karakteristične grafove za izvore koji su bili kandidati, te su na taj način procjenjivali je li određeni izvor pulsar ili ne [21]. Kako bi shvatili što točno predstavljaju prikupljeni podatci i njihovi grafovi, moramo prvo razumjeti što se događa s radiovalovima dok putuju kroz svemir.

Svemir nije savršeni vakuum kroz koji radiovalovi neometano putuju. Radiovalovi se na svom putu u međuzvjezdanom mediju (engl. Interstellar Medium, ISM) raspršuju na elektronima. Također, distribucija elektrona u ISM-u nije uniformna, tako da će se putovanja radiovalova iz različitih izvora razlikovati. Faktor koji nam opisuje disperziju u ISM-u se

naziva ”mjera disperzije” (engl. Dispersion Measure) i definira se kao

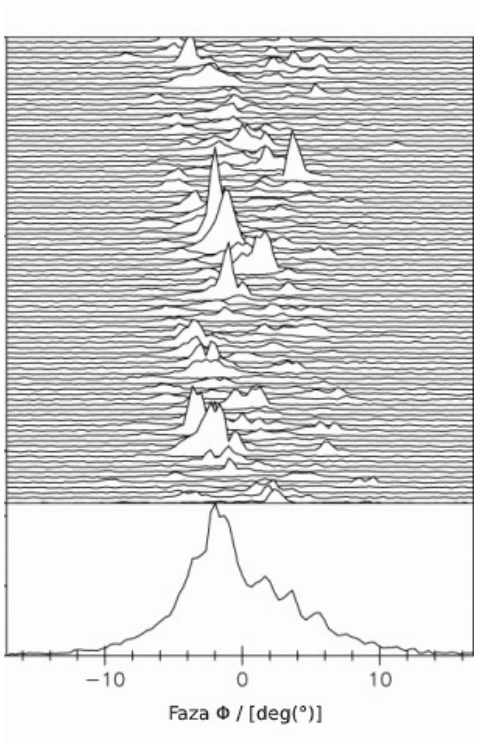
$$\mathrm{DM} = \int_0^{\mathrm{d}} \mathrm{n_e dl}, \tag{2.9}$$

gdje je n_e gustoća broja elektrona, a integracija se vrši po dužini putanje vala. Također, ISM više ometaju niske frekvencije od visokih, što je vidljivo na Slici 2.6.

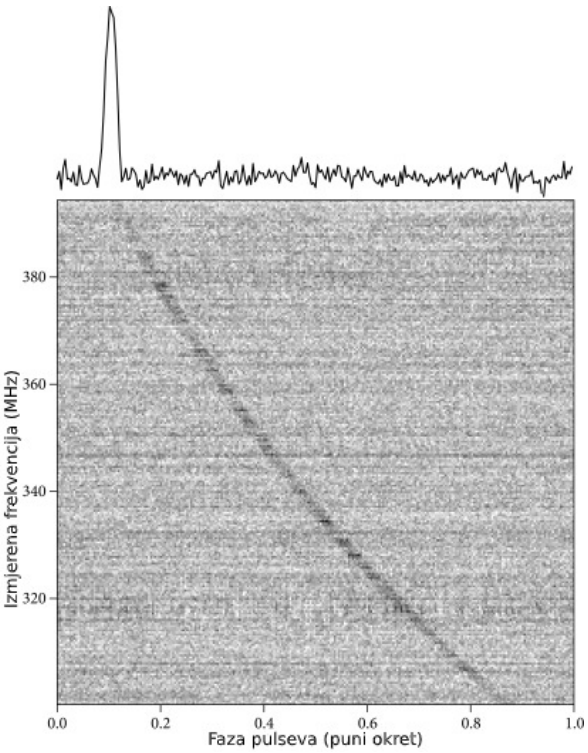


Slika 2.6: Različite količine pozadinskog šuma za različite frekvencije [21]

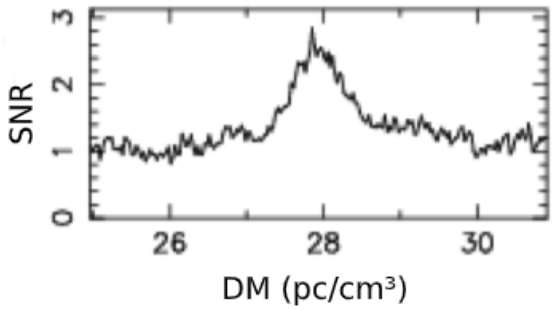
Zbog ovoga astronomi prikupljaju podatke sa više frekvencija, te ih na kraju spajaju u jednu krivulju koja se naziva ”integrirani srednji profil” (engl. Integrated Mean Profile), vidljiv na Slici 2.7. Također, zbog djelovanja ISM-a na radiovalove, puls koji je generirao pulsar će doći u različitim vremenima za različite frekvencije što je vidljivo na Slici 2.8. Ovo se rješava tako što se aproksimira vrijednost DM-a, ali ta vrijednost nije sigurna, tako da je Slika 2.9 također od velike važnosti. Naziva se DM-SNR (engl. Dispersion-Measure-Signal-to-Noise-Ratio) graf, a predstavlja ovisnost omjera signala i pozadinskog šuma o vrijednosti DM-a [28]. Grafovi na Slikama 2.7 i 2.8 su najznačajniji za identifikaciju pulsara. Prije su ih ljudi ručno analizirali. Novija metoda analize ovih grafova je strojno učenje koje omogućava preciznu obradu velike količine podataka u kratkom vremenu.



Slika 2.7: Određivanje najboljeg srednjeg integriranog profila [27]



Slika 2.8: Disperzija valova u ovisnosti o frekvenciji [28]



Slika 2.9: DM-SNR krivulja [27]

3 Osnove strojnog učenja

3.1 O strojnom učenju

Cilj poznate tvrtke Google je ”organizirati sve svjetske informacije i učiniti ih lako dostupnima svima” [29]. Ovo nije nimalo lak pothvat, s obzirom da je rast količine informacija koje su dostupne na internetu sve veći i veći. Ukupna količina podataka na internetu je bila 33 zetabajta (engl. zettabyte, 2^{70} bajta) 2018. godine, a predviđanja su da će taj broj doseći 175 zetabajta do 2025. godine [30].

Obrada ovako velike količine podataka zahtjeva konstantnu inovativnost u području njihove klasifikacije. Upravo je to i cilj većine algoritama strojnog učenja - što preciznije klasificirati podatke uz minimalnu potrebu za ljudskom pomoći. Razvoj ovih algoritama je potaknuo njihovu prilagodbu i primjenu u raznim područjima života, te se tako pored primarnog cilja klasifikacije podataka razvijala i ”umjetna inteligencija” koja je omogućila računalima da uče neke procese analize bolje nego ljudi. Tako možemo sve češće vidjeti novosti o tome kako je računalo pobijedilo najbolje svjetske natjecatelje u šahu ili igri Go, te kako vozila koja upravljaju sama sobom pokazuju bolju sposobnost vožnje od ljudi. Ovakve vijesti su podigle popularnost strojnog učenja koje se danas koristi u primjenama kao što su medicinska dijagnostika, pametna organizacija naše elektroničke pošte, bolja procjena tržišta, ili prepoznavanje govora [31, 32].

Metode strojnog učenja postaju sve dostupnije i lakše za primjenu. Matematička i programerska zajednice su kreirale intuitivne alate i napredne pakete koji se mogu lako koristiti u već postojećim programskim jezicima. Popularne metode korištenja algoritama strojnog učenja su paketi napisani za programske jezike R (paket *caret* [33]) i Python (paket *scikit-learn* [34]). U ovom radu se koristi Python i *scikit-learn* paket napisan za njega.

Potrebno je prepoznati kada je strojno učenje dobar alat za rješavanje problema. U nekim situacijama mogu se postići bolji rezultati korištenjem drugih metoda. Problem kod strojnog učenja su podaci nad kojima se vrši učenje. Njih kao prvo mora biti puno, te moraju biti pravilno kategorizirani, jer u suprotnom algoritam ima tendenciju krivog zaključivanja [35]. Tipični problemi koji se pojavljuju kod kategorizacijskih algoritama strojnog učenja su podnaučenost (engl. *underfitting*) i prenaučenost (engl. *overfitting*) modela koji su vezani uz krivo modeliranje aproksimirane funkcije. Ovi problemi će se detaljnije razraditi u idućem potpoglavlju vezanom uz konkretne implementacije kategorizacijskih algoritama.

Zajedničko za sve vrste strojnog učenja je to da pokušavaju doći do određenog cilja

generaliziranjem, odnosno organizacijom dosadašnjih iskustava. To primjerice mogu biti već klasificirani podaci zvani skup za učenje (engl. *training set*) koji algoritam koristi za učenje, ili pravila ponašanja koje algoritam isprobava kako bi zaključio na temelju iskustva koja kombinacija koraka je najbolja za rješavanje problema. Na ovaj način možemo klasificirati algoritme strojnog učenja [32]:

- **Nadzirano strojno učenje** (engl. *supervised learning*) - Algoritam će učiti iz već kategoriziranih podataka, na temelju kojih se generira aproksimirana funkcija koja će za neki novi skup podataka dati rezultat.

- **Nenadzirano strojno učenje** (engl. *unsupervised learning*) - Algoritam će sam tražiti zajedničke karakteristike na skupu podataka koji nisu organizirani i označeni, te će na taj način formirati sposobnost da karakterizira novi podatak kao pripadnika određene grupe podataka u postojećem skupu sa sličnim osobinama.

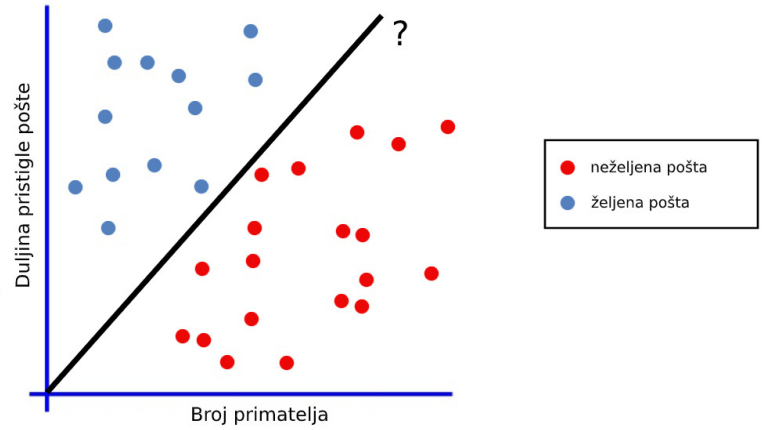
- **Podržano strojno učenje** (engl. *reinforcement learning*) - Algoritam će dati najbolji idući korak temeljen na podacima koje trenutno ima, tako što će isprobati moguće iduće korake te tražiti korak za koji dobije najveću nagradu, odnosno korak s najboljim ishodom.

Podaci koji se koriste u ovom radu za istraživanje pulsara su već kategorizirani, tako da ćemo u sljedećem poglavlju detaljnije razraditi princip rada algoritama nadziranog strojnog učenja koje ćemo koristiti.

3.2 Nadzirano strojno učenje

Kao što je već spomenuto u prethodnom poglavlju, cilj nadziranog strojnog učenja je naučiti računalo da stvori model iz danih podataka po kojem će poslije kategorizirati nove podatke. Kao primjer pokušajmo riješiti problem kategorizacije pristigle elektroničke pošte na željenu i neželjenu. Potrebni su nam podaci o primljenoj pošti. Recimo da imamo podatak o duljini sadržaja poruke, te broj primatelja. Kada bi ta dva podatka postavili kao x i y os na grafu, dobili bi rezultat prikazan na Slici 3.1. Vidimo da je sva pristigla pošta koja je imala puno primatelja, a bila je kratkog sadržaja, kategorizirana kao neželjena. U suprotnom slučaju, kada je broj primatelja bio mali a duljina sadržaja velika, pristigla pošta nije bila neželjena. Kada bi sami trebali odrediti granicu između željene i neželjene pošte, vjerojatno bi povukli

ravnu liniju negdje u području između crvenih i plavih točaka. Na taj način bi kategorizirali novu poštu tako što bi je stavljali ”iznad” ili ”ispod” povučene crte. Međutim, želimo automatizirati ovaj proces, odnosno naučiti računalo da samo odredi najbolje rješenje.

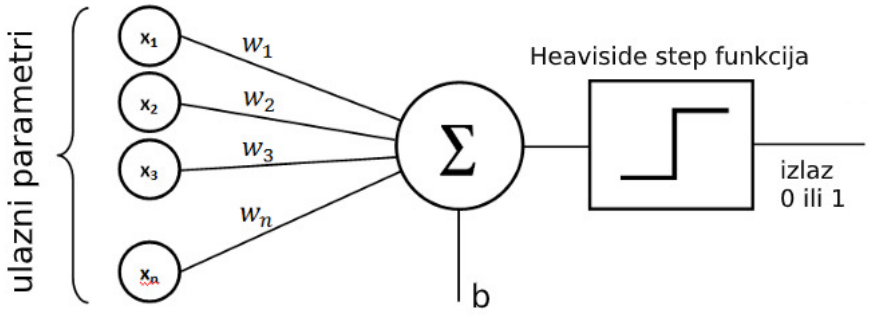


Slika 3.1: Prikaz podataka o pristigloj pošti u ovisnosti o dva parametra

Rješenje ovog problema ponudio je američki psiholog Frank Rosenblatt još 1957. godine [36]. Njegov algoritam funkcionira za n dimenzionalni problem, a rješenje je uvijek $n-1$ dimenzionalno. Naš primjer sa pristiglom poštom je dvodimenzionalan problem jer ima dva parametra, a rješenje je jednodimenzionalno jer je pravac. Uzmimo da imamo n ulaznih parametara $[x_1, x_2, ..., x_n]$, te set težinskih parametara $[w_1, w_1, ..., w_n]$. Definicija rezultantne funkcije je:

$$f(x) = \begin{cases} +1 & \text{ako } \sum_{i=1}^{n} x_i w_i > b \\ 0 & \text{u ostalim slučajevima} \end{cases} \tag{3.1}$$

gdje je b pristranost (engl. bias). Kako bi bili sigurni da će izlazna vrijednost funkcije biti uvijek 0 ili 1, na sumu primjenjujemo Heavisideovu step-funkciju $H(x)$ koja će dati 0 za sve vrijednosti $x < 0$, te 1 za sve ostale vrijednosti. Taj proces je prikazan na Slici 3.2.



Slika 3.2: Shema Rosenblattovog perceptrona [37]

Računalu dajemo m kategoriziranih podataka p zajedno s njihovim kategorizacijama y, dakle $\{\mathbf{p_i}, y_i\}_{i=1}^{m}$. Svaki podatak p_i sadrži parametre $x_{1,\ldots,n}$, a računalo bi samo trebalo odrediti sve težinske faktore $w_{1,\ldots,n}$ kao i parametar pristranosti b. Način na koji se ovo postiže je sljedeći [37]:

Algoritam 1: Rosenblattov perceptron

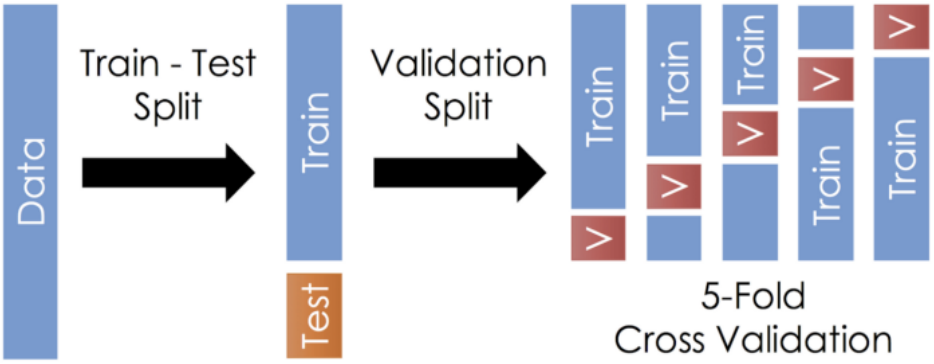
dodijelimo nasumične vrijednosti svim $w_{1,\ldots,n}$ i parametru pristranosti b;
dok *nemamo konvergenciju* **čini**
- ### prolazimo kroz sve ulazne vrijednosti;
za *j = 1,m* **čini**
usporedba pravog rezultata s trenutnom pretpostavkom;
pogreska = $y_j - H(suma\_svih(p_j w_j) + b)$;
ako je model krivo procijenio rezultat;
ako *pogreska != 0* **onda**
ažuriramo vrijednost težinskih faktora $w_{1,\ldots,n}$;
$\mathbf{w} = \mathbf{w} + pogreska \cdot p_j$;
ažuriramo parametar b;
$b = b + pogreska$;
kraj
kraj
ispitujemo imamo li konvergenciju;
kraj

Rezultat ovog procesa su težinske vrijednosti $w_{1,\ldots,n}$ i parametra pristranosti b koji definiraju hiper-ravninu (engl. *hyperplane*) koja dijeli ulazne vrijednosti na dvije skupine. Dodavanjem nove ulazne vrijednosti koja nije već prije kategorizirana, određuje se njena pripadnost jednoj od grupa tako što se za njene ulazne parametre računa rješenje funkcije koja sada ima definirane težinske parametre i parametar pristranosti b.

Ovaj algoritam ima više problema, a najveći od njih je taj što se za učinkovit rad mora osigurati linearna separabilnost početnih podataka. Također, jednom kada se postigne konvergencija težinske vrijednosti se prestaju prilagođavati podacima, što znači da nećemo uvijek dobiti najbolje moguće rješenje za neki problem [38]. Ovaj algoritam je opisan kako bi se pokazalo kako je izgledao početak nadziranog strojnog učenja te kako bi se na jednostavan način prikazao princip rada jednog algoritma. Kod klasifikacijskih problema strojnog učenja rezultat je diskretan, npr. elektronička pošta je ili željena, ili neželjena. Rezultati

regresijskih problema strojnog učenja imaju neprekidne vrijednosti.

Ono što je zajedničko svim algoritmima nadziranog učenja je način na koji se oni treniraju na ulaznim podacima. Ako se raspolaže sa skupom kategoriziranih ulaznih podataka, neće se svi koristiti za treniranje sustava. Određeni dio se koristi za provjeru valjanosti naučenog modela. Proces je prikazan na Slici 3.3.

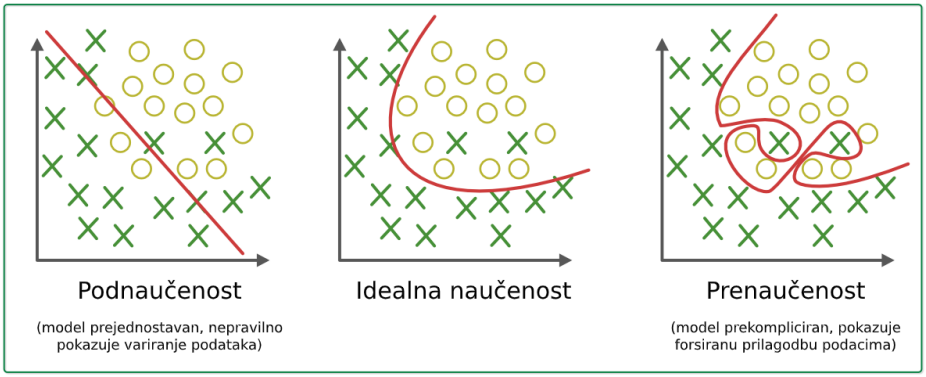


Slika 3.3: Prikaz podjele ulaznih podataka kako bi se postigla najbolja naučenost modela [39]

Osim odvajanja određenog dijela podataka za testiranje, također se radi i unakrsna provjera (engl. *cross-validation*). To je proces kod kojeg se model približava najboljem rezultatu kroz učenje i provjeru na različitim dijelovima skupa podataka. Razlog za ovo je taj što mi ne možemo garantirati da smo uzeli najbolji reprezentativni primjer ulaznih podataka. Npr. gradimo model na temelju anketiranja određenog dijela populacije i odlučimo odvojiti 50% podataka na učenje modela. Može se dogoditi da će u tih 50% podataka biti više ženske populacije u usporedbi s muškom te ćemo dobiti nepravilnu raspodjelu podataka čime riskiramo krivu naučenost modela. Zbog ovog je najčešće najbolje odvojiti $> 70\%$ ulaznih podataka na učenje, a zatim primijeniti višestruku unakrsnu provjeru na tim podacima, kako bi dobili što bolju naučenost modela [40].

Kod odabira početnih postavki za učenje modela, potrebno je razmišljati o tome kakvim podacima raspolažemo, te koliko podataka imamo na raspolaganju. Dva najčešća problema koja se pojavljuju kod učenja nekog modela su podnaučenost (engl. *underfitting*) i prenučenost (engl. *overfitting*). Primjeri ovakvih situacija su dani na Slici. 3.4. Podnaučenost je najčešće problem manjka podataka nad kojima se uči, dok je prenaučenost puno češći problem kojeg je često problematično izbjeći, jer se tijekom učenja modela situacija čini idealnom. Tek se testiranjem na novim podacima može pokazati da model neke podatke ne kategorizira kako bi trebao [41].

Uzimajući u obzir sve navedene probleme, potrebno je odgovoriti na pitanje provjere



Slika 3.4: Prikaz podnaučenosti i prenaučenosti nekog modela [41]

pravilne naučenosti modela. Ovo se najbolje pokazuje analizirajući testne podatke, odnosno provjerom naučenosti modela s podacima čiju točnu kategorizaciju znamo. Postoje tri numeričke vrijednosti koje su indikatori naučenosti modela, te je cilj maksimizirati svaku od njih. Sve tri se baziraju na četiri kategorizacijske vrijednosti. U slučaju da je model točno predvidio neku kategoriju testnog podatka, bila ona negativna ili pozitivna, tada govorimo o *istinskom pozitivu* ili *istinskom negativu* (engl. *True Positive - TP* i *True Negative - TN*). Ukoliko je model krivo kategorizirao neku testnu vrijednost, tada govorimo o *lažnom pozitivu* i *lažnom negativu* (engl. *False Positive - FP* i *False Negative - FN*). Na temelju ovog definiraju se tri mjere naučenosti modela [42]:

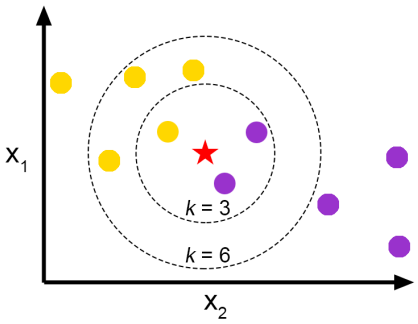
- **Točnost (engl. *accuracy*)** - definira se kao $\frac{TP+TN}{TP+TN+FP+FN}$ te predstavlja omjer pogođenih podataka u odnosu na sve podatke.
- **Preciznost (engl. *precision*)** - definira se kao $\frac{TP}{TP+FP}$ te pokazuje koliko je model precizan u pogađanju određene kategorije
- **Odziv (engl. recall)** - definira se kao $\frac{TP}{TP+FN}$ te pokazuje koji udio pogođenih pozitivnih podataka je točno klasificiran. Ovo je najvažniji parametar kod problema gdje je cilj pronaći što više podataka koji pripadaju određenoj kategoriji, kao što je naš problem pronalaska pulsara.

U nastavku će se preciznije razraditi tri klasifikacijska algoritma koja ćemo koristiti s ciljem kategorizacije i pronalaska potencijalnih kandidata za pulsare.

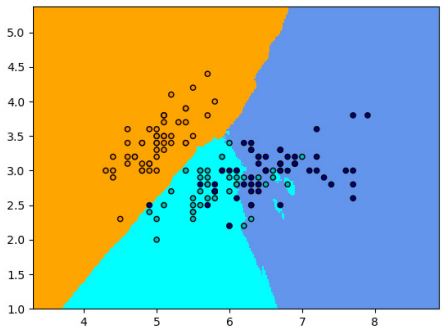
3.3 Algoritam k-najbližih susjeda (KNN)

Metoda k-najbližih susjeda (KNN, engl. *K-Nearest Neighbours*) je jedan od jednostavnijih algoritama za razumijevanje i implementaciju, te se kao takav često koristi. Kada se ispituje

kategorija nekog novog podatka, tada algoritam provjerava postojeće kategorizirane podatke, te traži k onih koji su mu najbliži po ulaznim parametrima. Zatim algoritam gleda kojoj kategoriji pripada najviše susjednih podataka, te na temelju toga dodjeljuje tu kategoriju novom podatku. Ovo je prikazano na Slici 3.5a gdje vidimo da kategorizacija određenog podatka ovisi o tome koliki faktor k koristimo, odnosno koliko najbližih susjeda uzimamo u obzir. Također, na Slici 3.5b su prikazana kategorizacijska područja, gdje je vizualizirano u koju kategoriju će se svrstati novi podatak. Naravno, primjeri na slikama su dvodimenzionalni jer sadrže samo dva ulazna parametra, ali algoritam je primjenjiv na n - dimenzionalne probleme [32].



(a) Prikaz kategorizacije ovisno o odabiru faktora k. Za $k = 3$ će podatak biti svrstan u ljubičastu kategoriju, dok će za $k = 6$ biti svrstan u žutu.



(b) Prikaz 3-kategorizacijske klasifikacije (k=15)

Slika 3.5: Princip rada KNN algoritma [43, 44]

Ovaj algoritam, osim već objašnjenog faktora k, sadrži više mogućih modifikacija koje je potrebno prilagoditi problemu koji rješavamo, kako bi optimizirali rezultantnu kategorizaciju, kao i brzinu izvođenja, a neke od njih su [45]:

- **Težinska udaljenost** - ovaj način izvođenja neće tretirati sve susjedne podatke jednako, već će pridodati veću vrijednost onim podacima koji su bliži. Ta vrijednost opada sa $\frac{1}{d}$ gdje je d udaljenost od podatka koji se kategorizira.

- **Način izračuna udaljenosti** - Osim euklidskog načina izračuna udaljenosti, koji bi se mogao interpretirati kao "zračna udaljenost" između dvije točke na karti, koristi se i manhattanski način koji bi se mogao interpretirati kao najkraća udaljenost ako se do točke dolazi samo pod pravim kutevima, kao da imamo ceste koje su napravljene pod pravim kutevima. Iako se euklidski način čini logičnijim za korištenje, kod višedimenzionalnih podataka zbog optimizacije brzine rada algoritma se ipak koristi manhattanski način računanja duljine.

- **Način sortiranja** - kod malog broja podataka se svi podaci mogu sortirati po udaljenosti od novog podatka, međutim povećanjem broja podataka značajno opada brzina izvođenja algoritma. Stoga je pri većim količinama podataka poželjno koristiti konstrukciju stabla, koja efektivno dijeli početne podatke na područja, tako da se pri dodavanju novog podatka brzo pronalazi područje u koje podatak vjerojatno pripada te se nakon toga ponovno izvodi obično sortiranje podataka u tom području.

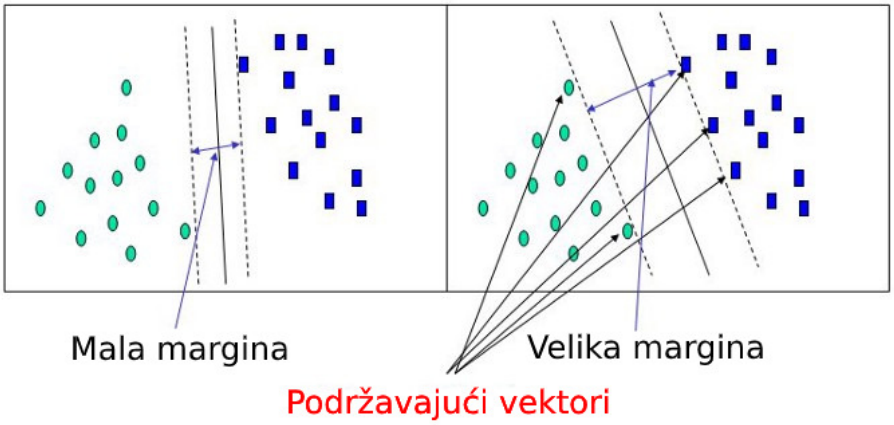
- **Veličina lista** - veličina koja je vezana uz već spomenutu konstrukciju stabla, a predstavlja broj ulaznih podataka kod kojeg će stati konstrukcija dodatnih grananja stabla. Ovu veličinu treba postaviti na način da se što više smanji vrijeme kategorizacije novog podatka, odnosno treba odrediti za koliko podataka je brže odraditi sortiranje podataka u grani nego nastaviti s konstrukcijom dodatnih grana.

Ovaj algoritam je primjenjiv u većini situacija, međutim postoje određene mane na koje treba obratiti pozornost. Postoje algoritmi koji su efikasniji kod kategorizacije jako velikih količina novih podataka. Naime, KNN algoritam je vrlo brz kod učenja modela jer treba eventualno konstruirati stablo, ali je spor kod kategorizacije jer za svaki novi podatak vrši izračun udaljenosti. Također, algoritam nema mogućnost davanja veće važnosti određenom parametru, već sve parametre tretira samo kao dimenzije pomoću kojih računa udaljenosti [46].

3.4 Algoritam potpornih vektora (SVM)

Metoda potpornih vektora (SVM, engl. *Support Vector Machine*) je po svom načinu rada slična spomenutom Rosenblatt-ovom perceptronu, jer se u svojoj osnovnoj implementaciji koristi za klasifikaciju podataka sa samo dvije kategorije, te po sličnom principu traži hiper-ravnine koje će podijeliti dvije grupe ulaznih podataka. Međutim, ključna razlika je u tome što za razliku od perceptrona, SVM algoritam ne zahtjeva linearnu separabilnost podataka, što rješava koristeći "meku marginu" (engl. *soft-margin*) te kernel trikove. Na Slici 3.6 je prikazan princip rada SVM algoritma.

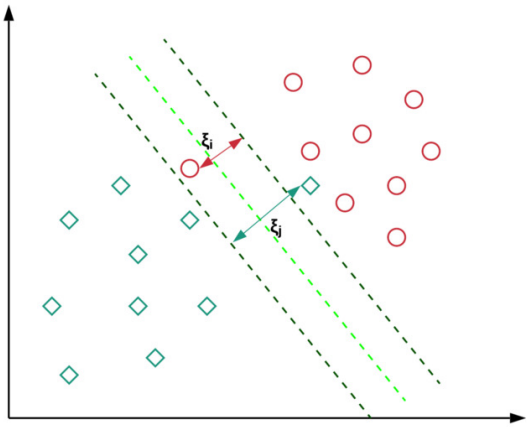
Kao što je vidljivo na Slici 3.6, algoritam neće tražiti jednu hiper-ravninu s kojom će podijeliti ulazne podatke, već će zapravo tražiti dvije paralelne hiper-ravnine i to na način da te dvije hiper-ravnine budu što više udaljene jedna od druge. Veličina koja predstavlja udaljenost te dvije paralelne hiper-ravnine naziva se margina, dakle cilj algoritma je pronalazak maksimalne margine. Ovaj proces funkcionira kada imamo linearnu separabilnost poda-



Slika 3.6: Prikaz osnovnog načina rada SVM algoritma [47]

taka, te je po postupku traženja hiper-ravnina vrlo sličan već objašnjenom Rosenblattovom perceptronu. Razlika je u tome što konačno rješenje nisu same hiper-ravnine, već potporni vektori (engl. *support vectors*) koji su ulazni podaci koji su najbliži hiper-ravninama te ih na taj način i definiraju [47].

Ono po čemu je algoritam još poseban je već spomenuta ”meka margina”, koja dozvoljava i krivo klasificirane ulazne podatke koji narušavaju linearnu separabilnost podataka, što se rješava tako da se dopušta određena pogreška kod klasifikacije ulaznih podataka koji su blizu hiper-ravnina. Takav slučaj, kao i iznos pogreške prikazani su na Slici 3.7.



Slika 3.7: Prikaz izračuna hiper-ravnina kod krivo klasificiranih podataka koji narušavaju linearnu separabilnost podataka [48]

Cilj SVM algoritma u ovom slučaju je traženje takvih vrijednosti w, b koje će minimizirati

sljedeći izraz:

$$arg\ \underset{w,b}{min}\ \frac{1}{2}\left\|\vec{w}\right\|^2 + C\sum_{i=1}^{n}\xi_i, \tag{3.2}$$

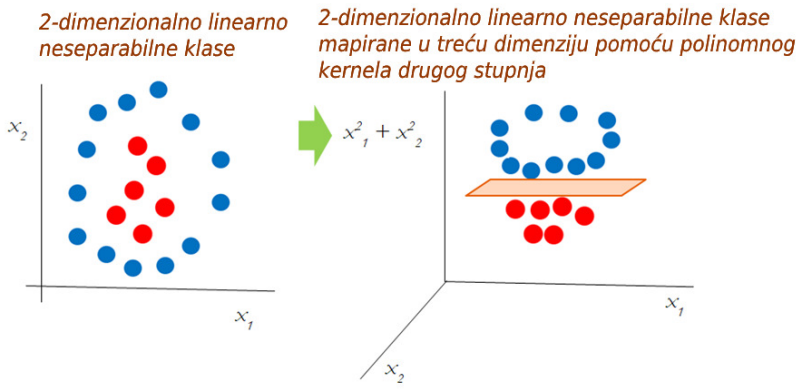
odnosno

$$arg\ \underset{w,b}{min}\ \frac{1}{2}\left\|\vec{w}\right\|^2 + C\sum_{i=1}^{n}max\{0, 1 - y_i(w \cdot x_i + b)\}. \tag{3.3}$$

Prvi član u izrazu je normalizacija težinskog vektora $\vec{w}$, čija minimizacija predstavlja traženje najveće margine, jer je margina b definirana kao $b = \frac{2}{\|\vec{w}\|}$. Dakle što više uspijemo smanjiti normalizaciju vektora $\vec{w}$, to ćemo dobiti veću marginu. Drugi član u izrazu je vezan za rješavanje problema nelinearne separabilnosti zbog krive klasifikacije ulaznih podataka. Član C naziva se regularizacijski parametar i njega određujemo pri učenju modela. Postavimo li preveliku vrijednost tog parametra, tada će se algoritam više fokusirati na problem maksimiziranja margine, te ćemo kao rješenje dobiti hiper-ravninu koja ignorira moguće pogrešne podatke. Postavimo li premalu vrijednost tog parametra, algoritam će ignorirati točnu klasifikaciju ulaznih podataka, te će se samo fokusirati na minimizaciju pogrešnih podataka, što opet neće dati idealno rješenje. Član ξ_i u SVM-u je funkcija gubitka zglobnice (engl. *hinge loss function*). Ta funkcija će biti > 0 samo kada se neki od ulaznih podataka nalazi sa pogrešne strane hiper-ravnine. Također, oni podaci koju se nalaze daleko na krivoj strani hiper-ravnine će imati manji utjecaj na promjenu modela od onih pogrešnih podataka koji su blizu. Logično je da je podatak koji se nalazi "duboko" na krivoj strani klasificiranih ulaznih podataka, krivo klasificiran te bi on trebao imati manji utjecaj na izračun točne hiper-ravnine [49].

Još jedan način rješavanja linearne neseparabilnosti podataka su također već spomenuti kernel trikovi. Korištenje metode meke margine nam je pomoglo kod podataka koji su i dalje mogli biti pravilno podijeljeni pomoću hiper-ravnine čija je dimenzija $n - 1$, gdje je n broj ulaznih parametara. Složeniji problem nastaje kada su nam podaci organizirani kao na lijevoj strani Slike 3.8. Očito je da bi kružnica bila idealno rješenje za podjelu ovih podataka. SVM to rješava na način da ulaznim podacima dodjeljuje dodatnu dimenziju koju računa pomoću određene kernel funkcije te na taj način uspjeva pronaći hiper-ravninu koja će uspješno kategorizirati podatke. Kod primjera na Slici 3.8 ulazni podaci su preslikani u treću dimenziju pomoću funkcije čija je vrijednost jednaka $x_1^2 + x_2^2$. Na ovaj način su točke koje su bile bliže centru u trećoj dimenziji preslikane niže od točaka koje su bile dalje od centra. Nakon ovog postupka algoritam lako pronalazi hiper-ravninu koja dijeli ulazne podatke, što

je vidljivo na desnoj strani slike 3.8. Kernel funkcija može biti bilo koja funkcija koja će preslikati podatke u višu dimenziju. Najčešće korištene kernel funkcije je već spomenuta polinomna kernel funkcija, te uz nju još radijalna i hiperbolična kernel funkcija [50].



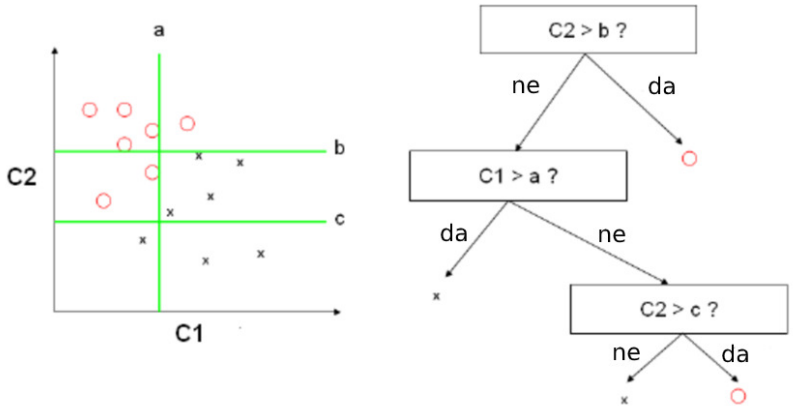
Slika 3.8: Prikaz izračuna hiper-ravnine nakon primjene polinomne kernel funkcije drugog stupnja [50]

Potencijalni problemi kod korištenja SVM algoritma su situacije gdje imamo loše balansirane podatke, odnosno gdje u jednoj od kategorija ima značajno više podataka. Također, algoritam je u svojoj osnovnoj implementaciji namijenjen samo za binarnu klasifikaciju. Postoje implementacije SVM algoritma koje funkcioniraju i za podatke koji imaju više kategorija, ali u ovim slučajevima je bolje koristiti neki drugi algoritam [51].

3.5 *Klasifikacijsko stablo odlučivanja (DTC algoritam)*

Stabla odlučivanja se koriste za klasifikacijske i regresijske probleme strojnog učenja. Algoritmi stabala odlučivanja uče preko niza pitanja i odgovora. Klasifikacijsko stablo odlučivanja (engl. *Decision Tree Classifier, DTC*) algoritam je po svojoj prirodi jednostavan za razumijevanje i implementaciju. Bazira se na konstrukciji klasifikacijskog stabla, kao što je prikazano na Slici 3.9.

Pri procesu učenja, odnosno konstrukcije stabla, algoritam će tražiti vrijednosti ulaznih parametara koje najbolje dijele skup podataka, te će pomoću njih konstruirati mjesto grananja. Mjesto grananja je ono gdje se odlučuje u koju granu će podaci biti kategorizirani. Mjesto u stablu gdje se ne događaju dodatna grananja, već se podatku dodjeljuje finalna vrijednost, naziva se list. U primjeru sa Slike 3.9, algoritam je pronašao vrijednost parametra $C2 = b$ koja dijeli ulazne podatke na dva područja. Za jedno područje smo sigurni da je ulazni podatak kružić i nalazi se iznad crte b, te se zbog toga odmah konstruira list. Za drugo područje su potrebna dodatna odlučivanja i nalazi se ispod crte b. Konstrukcija stabla se



Slika 3.9: Prikaz konstruiranog stabla za neki skup ulaznih podataka s dva parametra C1 i C2 [52]

zaustavlja onog trenutka kada su konstruirani svi listovi i više nema potrebe za nastavkom kategorizacije odnosno dodatnim grananjem. Kada je stablo konstruirano, kategorizacija novih podataka vrši se na način da se taj podatak spušta niz stablo sve dok ne dođe do lista i u tom trenutku mu se dodjeljuje vrijednost definirana tim listom [53].

Najkompleksniji dio algoritma je odluka o vrijednosti ulaznog parametra koja daje optimalnu konstrukciju mjesta grananja, odnosno najbolje dijeli podatke. Zbog toga se najčešće koristi Gini nečistoća (engl. *Gini impurity*) koja je definirana je kao:

$$I_G(p) = 1 - \sum_{i=1}^{J} p_i^2 \quad \xrightarrow{\text{za } J=2} \quad I_G(p) = 1 - (p_1^2 + p_2^2), \tag{3.4}$$

gdje je p_i vjerojatnost pojavljivanja pojedine rezultantne kategorije u narednom podstablu, a J ukupan broj postojećih kategorija. Kao primjer, uzmimo da imamo samo dvije rezultantne kategorije što je pokazano na desnoj strani jednadžbe 3.4. Npr. za određenu vrijednost ulaznog parametra možemo dobiti grananje kod kojeg ćemo u podgranama imati ukupno tri rezultata iz prve kategorije $n_1 = 3$, te tri rezultata iz druge kategorije $n_2 = 3$. Tada ćemo za vrijednost Gini nečistoće dobiti:

$$I_G(p) = 1 - \left(\left(\frac{n_1}{n_1 + n_2} \right)^2 + \left(\frac{n_2}{n_1 + n_2} \right)^2 \right) = 1 - (0.5^2 + 0.5^2) = 0.5 \tag{3.5}$$

Ovo je maksimalna moguća vrijednost Gini nečistoće, jer nam ova grana sadrži jednak broj podataka iz obje kategorije. Ukoliko imamo slučaj da nam grana sadrži samo podatke iz jedne kategorije, tada vrijednost Gini nečistoće postaje $I_G = 0$, te nemamo daljnje grananje,

već se ta grana pretvara u list. Algoritam će tražiti vrijednost parametra za koju će mu iznos Gini nečistoće za pojedinu granu biti minimalan [54]. Uz Gini nečistoću se može koristiti i entropija. To je još jedan opis kvalitete grananja, a definirana je kao:

$$Entropija = -\sum_{i} p_i \log_2 p_i, \tag{3.6}$$

gdje su vrijednosti p_i vjerojatnosti pojavljivanja pojedine rezultantne podkategorije u idućem podstablu, kao i kod Gini nečistoće. Ova dva načina daju vrlo slične rezultate, ali se Gini nečistoća više koristi zbog brzine izvođenja algoritma, jer ne sadrži operaciju logaritmiranja koja zahtjeva veće vrijeme izračuna od operacije kvadriranja [55].

Jedan od problema korištenja stabla odlučivanja za klasificiranje podataka je česta pojava prenaučenosti modela. U tom slučaju se naknadno primjenjuju tehnike "obrezivanja" stabla (engl. *pruning*) gdje se namjerno uklanjaju dijelovi stabla koji ne pridonose previše procesu odlučivanja [56]. Također, dodatan problem je taj što male promjene ulaznih podataka mogu uzrokovati potrebu za rekonstrukcijom kompletnog stabla [57].

4 Istraživanje pulsara primjenom strojnog učenja

4.1 Korišteni alati i metode

U ovom poglavlju ćemo primijeniti prethodno opisano na skup podataka koji sadrži informacije o potencijalnim pulsarima. Koristimo tri opisana algoritma strojnog učenja. Za ovaj zadatak su primijenjeni paketi i programi koji su besplatni i svima dostupni na internetu, a napisani su s ciljem da budu što jednostavniji za krajnjeg korisnika. Svaki od paketa je otvorenog tipa (engl. *open-source*) što znači da ga održava velika zajednica programera i svatko se može priključiti kako bi surađivao na kodu. Programi, programski jezik i paketi koji su korišteni u ovom radu su kratko opisani u ovom potpoglavlju.

- **Python** - Python je programski jezik čija popularnost raste, primarno zbog jednostavnosti korištenja i sintakse. Kodovi napisani u njemu su vrlo čitljivi jer nema potrebe za korištenjem primjerice "{}" znakova za označavanje blokova koda kao u nekim drugim programskim jezicima, te su određeni dijelovi sintakse napisani s ciljem da čitanje koda podsjeća na čitanje teksta. Posljedica ovog je da se Python koristi i kod nas u srednjim školama za uvod u programiranje. Ovo ne znači da Python ima ograničenu funkcionalnost, čak naprotiv, ovaj programski jezik se koristi u svim područjima programiranja, od skriptiranja i zahtjevnih matematičkih analiza do izrade igrica ili web stranica. Ovo mu omogućuju mnogi paketi koju su za njega napisani, a vrlo se lako implementiraju i imaju odličnu dokumentaciju [58].

- **Jupyter bilježnica** - web aplikacija koja je dio većeg projekta zvanog "Jupyter Project" i primarno se koristi u kombinaciji s Python programskim jezikom. Vrlo je popularna kod podatkovnih znanosti jer omogućuje sekvencijalno izvođenje koda, kao i prikaz vizualizacija poput grafova i jednadžbi. Sekvencijalno izvođenje koda je korisno jer se na taj način može lako pojasniti i vizualizirati svaki međukorak kako bi se lakše shvatio proces izvođenja [59].

- ***numpy* i *pandas* paketi** - *numpy* je paket koji je koristan zbog posebne implementacije listi. Glavna klasa u tom paketu je *ndarray* koja predstavlja n-dimenzionalnu listu, a posebna je po tome što radi 50x puta brže od klasične liste u Pythonu. Razlog je taj što su *numpy* liste locirane kontinuirano na jednom mjestu u računalnoj memoriji te procesi ne moraju dohvaćati elemente sa različitih memorijskih lokacija. Ova funkcionalnost je u pozadini implementirana pomoću "C" programskog jezika koji omogućava

ovakvu manipulaciju memorijom. Naravno, kod podatkovnih znanosti često se radi s velikim i kompleksnim podacima te je ovakva optimizacija potrebna za njihovu brzu obradu [60]. Paket *pandas* također ima pozadinu u "C" programskom jeziku, a glavna funkcionalnost mu je *DataFrame* klasa koja omogućava lako učitavanje, pohranu, manipuliranje i spremanje različitih vrsta podataka. Klasa sadrži metode koje olakšavaju manipulaciju podataka potrebnu za mnoge probleme u podatkovnim znanostima [61].

- ***scikit-learn* paket** - kompleksan paket koji obuhvaća većinu poznatih algoritama strojnog učenja čime omogućuje laku implementaciju i korištenje. Sadrži implementaciju sva tri klasifikacijska algoritma koja će se koristiti u ovom radu, zajedno s mnogim dodatnim parametrima koji omogućuju prilagodbu izvođenja algoritama [62].

- ***matplotlib* i *seaborn* paketi** - *matplotlib* je paket koji je namijenjen grafičkoj vizualizaciji podataka, odnosno generiranju grafova [63]. Najbolje ga je koristiti u kombinaciji s već spomenutim *numpy* paketom. Paket *seaborn* je ekstenzija *matplotlib* paketa, a specijaliziran je za statistički prikaz podataka te se kao takav često koristi u podatkovnim znanostima za vizualizaciju velikih količina podataka, kao i analizu rezultata [64].

4.2 Uvid u HTRU-1 bazu podataka

Za primjenu algoritama na istraživanje pulsara prvo su nam potrebni podaci o kandidatima za pulsare. Različiti oblici ovih podataka su javno dostupni, te se mogu preuzeti u svrhu istraživanja. Veliko istraživanje urađeno 2010. godine koristilo je "Parkes Multibeam Receiver" teleskop koji se nalazi u Novom Južnom Wales-u u Australiji i primarni cilj je bio pronalazak kandidata za pulsare. Naziv istraživanja je "High Time Resolution Universe Pulsar Survey" (HTRU) te su pomoću već navedenog teleskopa prikupili 875 terabajta podataka i milijune potencijalnih kandidata za pulsare [65]. Ove podatke se dodatno filtriralo 2014. godine u istraživanju koje je izoliralo samo kandidate za pulsare iz pojasa srednje širine (engl. *medium latitude*) te su se na taj način generirali prvi javno dostupni klasificirani podaci o pulsarima. Taj skup podataka (HTRU-1) sadrži 1196 kandidata za pulsare, te 89995 kandidata koji nisu pulsari [66].

Svaki od kandidata za pulsare iz te baze podataka sadrži 30 karakterističnih vrijednosti koje ga opisuju, te finalnu klasifikacijsku vrijednost koja ima iznos 0 ili 1, gdje je 1 oznaka kandidata koji je pulsar, a 0 oznaka kandidata koji nije pulsar [67]. Neke od najbitnijih karakterističnih vrijednosti su [66]:

- Srednja vrijednost (engl. *mean*) integriranog profila

$$\bar{P} = \frac{1}{n}\sum_{i=1}^{n} p_i \tag{4.1}$$

- Standardna devijacija (engl. *standard deviation*) integriranog profila

$$\sigma_P = \sqrt{\frac{\sum_{i=1}^{n}\left(p_i - \bar{P}\right)^2}{n-1}} \tag{4.2}$$

- Koeficijent spljoštenosti (engl. *excess kurtosis*) srednjeg profila

$$k_P = \frac{\frac{1}{n}\left(\sum_{i=1}^{n}\left(p_i - \bar{P}\right)^4\right)}{\left(\frac{1}{n}\left(\sum_{i=1}^{n}\left(p_i - \bar{P}\right)^2\right)\right)^2} - 3 \tag{4.3}$$

- Koeficijent asimetrije (engl. *skewness*) integriranog profila

$$S_P = \frac{\frac{1}{n}\left(\sum_{i=1}^{n}\left(p_i - \bar{P}\right)^3\right)}{\left(\sqrt{\frac{1}{n}\left(\sum_{i=1}^{n}\left(p_i - \bar{P}\right)^2\right)}\right)^3} \tag{4.4}$$

- Srednja vrijednost (engl. *mean*) DM-SNR (engl. dispersion measure signal to noise ratio) krivulje

$$\bar{D} = \frac{1}{n}\sum_{i=1}^{n} d_i \tag{4.5}$$

- Standardna devijacija (engl. *standard deviation*) DM-SNR krivulje

$$\sigma_D = \sqrt{\frac{\sum_{i=1}^{n}\left(d_i - \bar{D}\right)^2}{n-1}} \tag{4.6}$$

- Koeficijent spljoštenosti (engl. *excess kurtosis*) DM-SNR krivulje

$$k_D = \frac{\frac{1}{n}\left(\sum_{i=1}^{n}\left(d_i - \bar{D}\right)^4\right)}{\left(\frac{1}{n}\left(\sum_{i=1}^{n}\left(d_i - \bar{D}\right)^2\right)\right)^2} - 3 \tag{4.7}$$

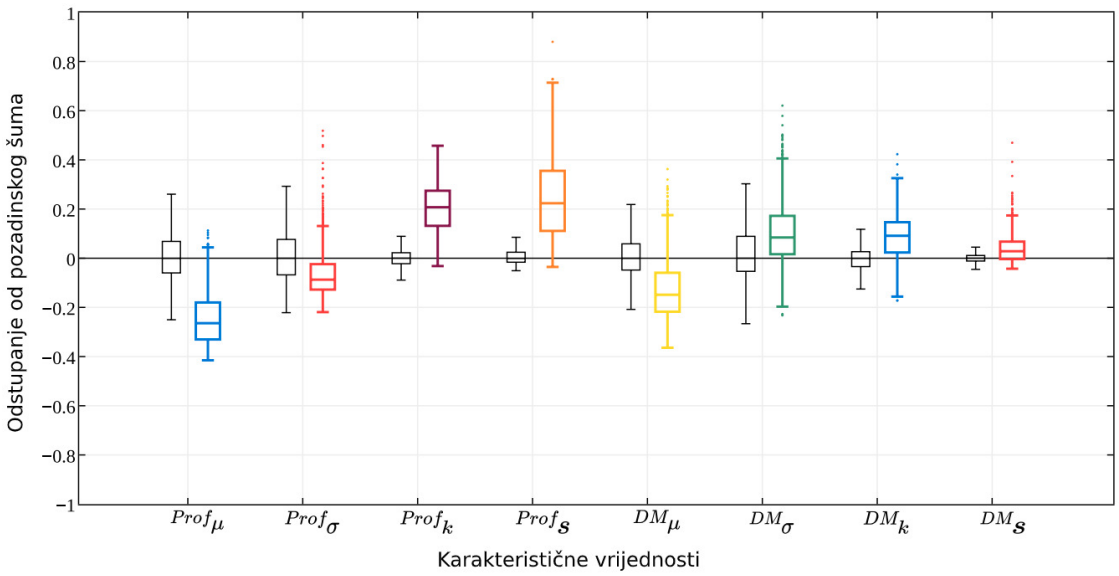
- Koeficijent asimetrije (engl. *skewness*) DM-SNR krivulje

$$S_D = \frac{\frac{1}{n}\left(\sum_{i=1}^{n}\left(d_i - \bar{D}\right)^3\right)}{\left(\sqrt{\frac{1}{n}\left(\sum_{i=1}^{n}\left(d_i - \bar{D}\right)^2\right)}\right)^3} \tag{4.8}$$

- Najbolja vrijednost perioda
- Najbolja vrijednost mjere disperzije (DM)
- Najbolja vrijednost omjera jačine signala naspram pozadinskog šuma (SNR)
- Širina pulsa

Uz navedene podatke tu su još i različite χ^2 vrijednosti vezane uz profile pulseva te statističke vrijednosti vezane uz frekvencije pri kojima su prikupljeni podaci. Vidljivo je da su prvih osam navedenih podataka zapravo četiri iste statističke vrijednosti za dvije krivulje, srednji integrirani profil i DM-SNR, koje su najbitnije za klasifikaciju pulsara. Razlog zašto su prve vrijednosti najbitnije za točnu klasifikaciju kandidata za pulsare je vidljiv na Slici 4.1. Na kutijastom dijagramu su prikazane vrijednosti odstupanja od pozadinskog šuma za svih 90000 kandidata za pulsare. Pošto se konkretne vrijednosti odstupanja previše razlikuju za različite karakteristične vrijednosti, te vrijednosti su skalirane na vrijednost koja je između -1 i 1 kako bi se bolje prikazao odnos kandidata koji su klasificirani kao pulsari i onih podataka koji su klasificirani da nisu pulsari. Karakteristične vrijednosti podataka koji su klasificirani kao pulsari su na grafu prikazani obojeno, dok su kandidati iz druge grupe prikazani crnom bojom. Iz dijagrama je vidljivo da kandidati koji su već klasificirani kao pulsari odstupaju od pozadinskog šuma, dok onim kandidatima koji su već klasificirani kao ne-pulsari srednje vrijednosti kutijastog dijagrama više manje leže na nultoj vrijednosti, što znači da ne odstupaju od pozadinskog šuma [66].

Kada smo analizirali algoritme koje ćemo koristiti za analizu podataka, spomenuli smo i njihove mane, odnosno situacije kod kojih će algoritmi dati loše rezultate. Jedna od tih situacija koja je zajednička većini algoritama strojnog učenja su nebalansirani podaci. Ovo znači da podaci nad kojima se vrši učenje modela imaju nejednak broj ulaznih podataka koji dijele istu kategoriju. Ako pogledamo HTRU-1 bazu podataka, vidjet ćemo da se podaci sastoje od od samo 1196 kandidata koji su pulsari, od ukupno 91192 kandidata, što znači da nam kategorija pulsara čini 1.31% podataka. Ovo znači da prije korištenja baze poda-

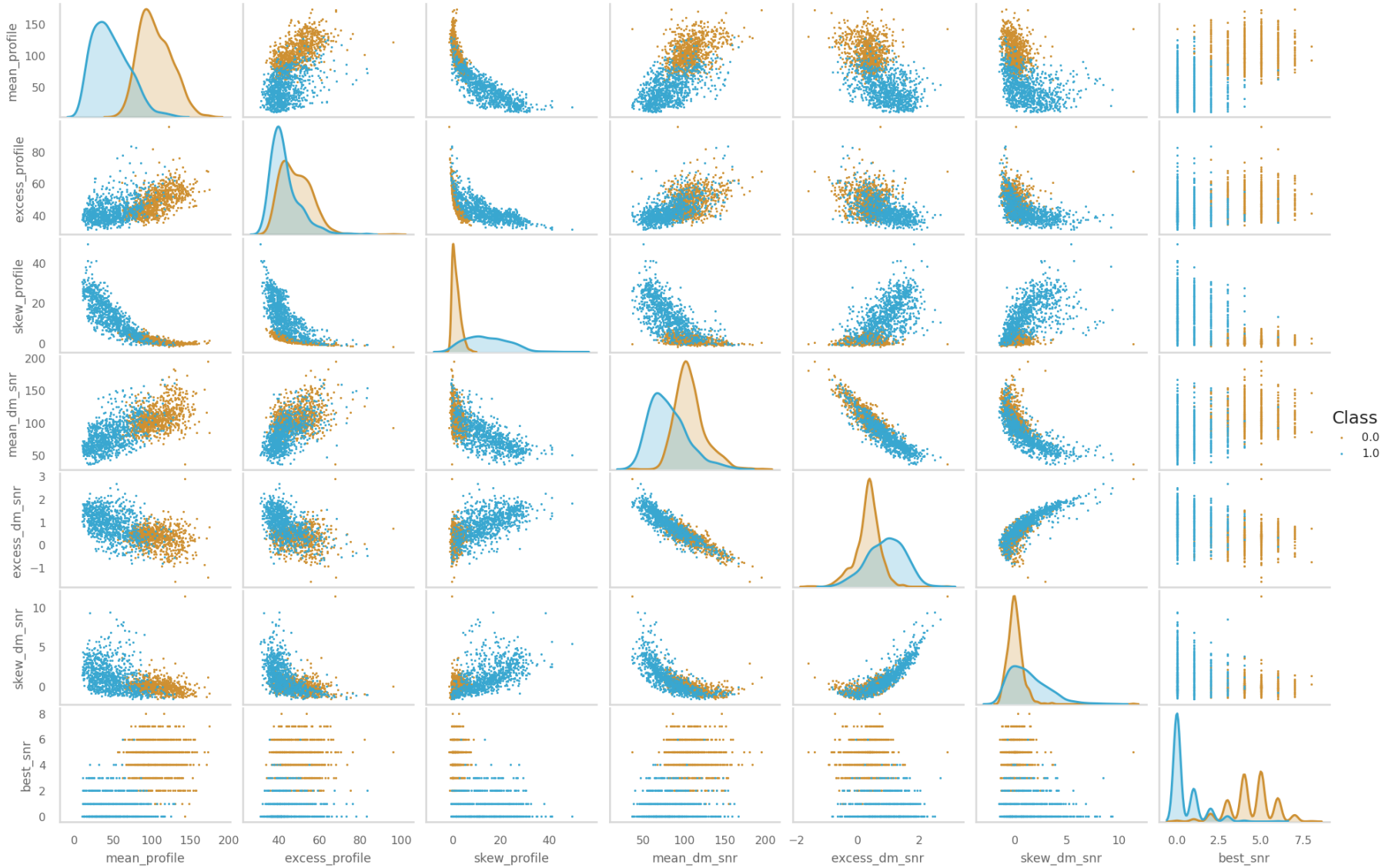


Slika 4.1: Prikaz odstupanja svake od osam karakterističnih vrijednosti od pozadinskog šuma. Veličine definirane u formulama (4.1)-(4.8) izabrane su kao značajke (engl. features) tipa profila (Prof) ili DM (engl. dispersion measure). Obojenim kutijastim dijagramima su prikazani podaci koji su klasificirani kao pulsari, dok su neobojanim dijagramima prikazani kandidati koji nisu klasificirani kao pulsari [66]

taka za učenje modela, potrebno je napraviti balansiranje podataka, što je proces eliminacije nausmičnih kandidata koji nisu kategorizirani kao pulsari.

Kao prvo, podaci su podijeljeni na one koji se koriste za treniranje i na one koji se koriste za testiranje, i to na način da se uzelo nasumičnih 25% podataka od ukupnog broja podataka kako bi ih se poslije koristilo za testiranje naučenog modela. Od preostalih (75%) podataka koje koristimo za učenje modela, odvojeni su oni koji su pulsari i oni koji nisu, te od onih koji nisu je nasumično oduzeto toliko kandidata da na kraju ostanemo s jednakim brojem kandidata u obje grupe. Na kraju spajamo te dvije grupe podataka u jednu i miješamo kandidate u rezultantnoj grupi koja će nam služiti za učenje modela. Na ovaj način se postiglo balansiranje podataka, te je rezultantni broj podataka nad kojima će se učiti modeli 1746, gdje svakoj kategoriji pripada 873 kandidata za pulsare.

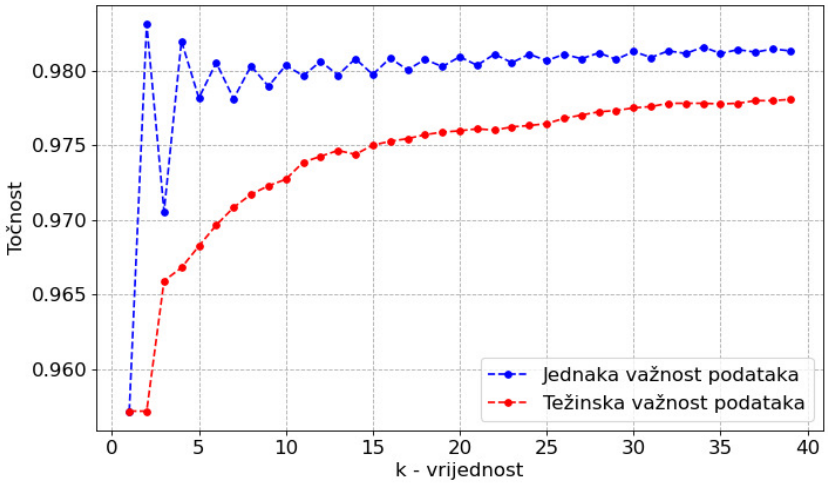
Prikazat ćemo međusobnu ovisnost balansiranih podataka koristeći *seaborn* paket za Python koji sadrži metodu *pairplot()*. Ta metoda omogućuje vizualizaciju podataka iz obje kategorije na takav način da se lako mogu vidjeti razlike u podacima [68]. Grafovi su prikazani na Slici 4.2.



Slika 4.2: Graf parnih relacija između odabranih karakteristika među podacima (formule 4.1, 4.3-4.5, 4.7, 4.8). Na nekim grafovima se može vidjeti jasna podjela dvije klase podataka, dok se kod nekih podaci preklapaju. Plava boja predstavlja pulsare tj. podatke označene kao klasa 1. Najbolji indikator separabilnosti kategorija su grafovi na dijagonali koji pokazuju razlike u samo jednoj karakterističnoj vrijednosti.

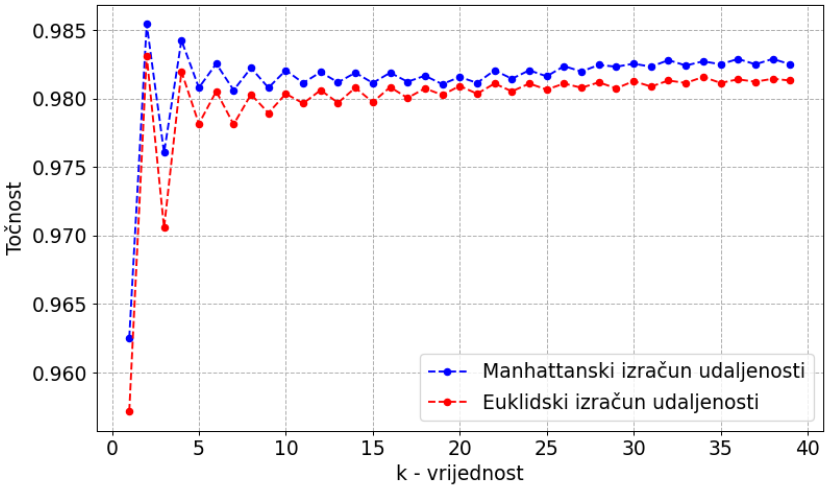
4.3 *Rezultati KNN algoritma*

Algoritam k-najbližih susjeda sadrži tri parametra kojima se postiže najbolja preciznost modela. To su parametar k koji nam govori koliko se susjednih podataka uzima u obzir pri klasifikaciji novog podatka, parametar koji će definirati način izračuna udaljenosti, te parametar koji definira hoće li svi podaci imati jednaku važnost ili će se uzimati težinska udaljenost. Testirat ćemo dva načina izračuna udaljenosti, a to su manhattanski i euklidski način [69]. Kao prvo, provjerit ćemo ima li smisla uzimati težinski način računanja udaljenosti u modelu tako što ćemo izračunati rezultate za sve vrijednosti k. Preciznost modela uzimamo kao glavni kriterij u što boljoj procjeni modela. Rezultati za sve kombinacije ova dva parametra su prikazani na Slici 4.3. Vidljivo je da kod podataka o pulsarima nema smisla koristiti težinski način dodjeljivanja vrijednosti udaljenostima.



Slika 4.3: Ovisnost točnosti modela za različite k vrijednosti i za dva različita načina dodjeljivanja vrijednosti udaljenostima.

Provjerit ćemo koji način izračuna udaljenosti je bolji, što je prikazano na Slici 4.4. Vidljivo je da je manhattanski način izračuna udaljenosti u svim slučajevima bolji od euklidskog. Također je vidljivo da za vrijednost $k = 2$ imamo najbolju preciznost modela. Međutim postoji problem kod odabira ove vrijednosti parametra k. Iz grafa je vidljivo da algoritam konzistentno daje bolje rezultate za parne vrijednosti parametra k. Ovo je rezultat binarne klasifikacije podataka. Zbog toga je često rezultat izjednačen kod klasifikacije novih podataka. Primjerice, za $k = 2$ često je novi podatak okružen s jednim susjedom koji je pulsar i sa jednim koji nije. Tada algoritam naizmjenično dodjeljuje različitu rezultantnu vrijednost tom podatku, što nije preporučljivo [69]. Ovu situaciju je lako izbjeći tako što ćemo uzeti ne-



Slika 4.4: Ovisnost točnosti modela za različite k vrijednosti i za dva različita načina izračuna udaljenosti.

parnu vrijednost parametra k, tako da uvijek imamo većinskog pobjednika među susjednim podacima. Zbog toga ćemo odabrati vrijednost $k = 5$ za dobivanje konačnih rezultata.

Što se tiče performansi izvođenja algoritma, odabrana je zadana vrijednost u scikit-learn paketu, a to je konstrukcija stabla sa veličinom listova $n = 30$. Ova vrijednost se pokazala dobrom za našu primjenu. Ovakva konstrukcija stabla je dala veću brzinu izvođenja od slučaja kada se stablo ne konstruira. Sve veličine listova veće od $n > 15$ su se pokazale jednako dobre u vremenu izvođenja.

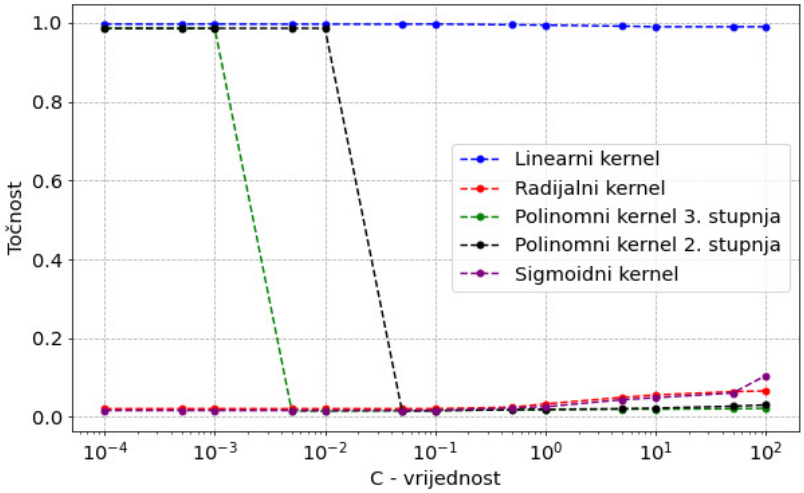
Izračunate vrijednosti za točnost, odziv i preciznost su prikazane u Tablici 4.1. Razlog za nisku vrijednost preciznosti je taj što skup podataka koji koristimo za testiranje modela nije balansiran kao što je slučaj kod podataka za učenje modela. Zbog ovog se kod testiranja modela pojavljuje velik broj lažnih pozitiva (*false positives* - FP) jer je broj kandidata koji nisu pulsari veći od onih koji jesu. Ovo značajno utječe na vrijednost u nazivniku izraza za preciznost koji je $\frac{TP}{TP+FP}$. Kada se balansiraju i testni podaci, tada preciznost dostigne vrijednost 98.7%. Zbog ovog ćemo ignorirati niske vrijednosti mjere preciznosti jer nam nisu toliko bitne za ono što mi želimo postići učenjem ovih modela.

Točnost	98.23%
Odziv	97.21%
Preciznost	45.31%

Tablica 4.1: Rezultatne vrijednosti za točnost, odziv i preciznost KNN algoritma za vrijednost $k = 5$

4.4 *Rezultati SVM algoritma*

Podaci o pulsarima nad kojima se obavlja učenje modela su kompleksni i stoga nije za očekivati linearnu separabilnost podataka. Kada nemamo linearnu separabilnost, tada je potrebno koristiti kombinaciju pravilnog kernela te je potrebno odrediti najbolji iznos C vrijednosti. Za početak ćemo učiti model pomoću različitih kernela, a to su linearni, polinomni, radijalni i sigmoidni. Svaki od njih ćemo testirati za više C vrijednosti. Rezultati su prikazani na Slici 4.5. Kao podsjetnik, C je vrijednost koja definira razinu ignoriranja krivo klasificiranih podataka koji su blizu hiper-ravnine. Niske vrijednosti parametra C znače da dozvoljavamo krivo klasificirane vrijednosti, što rezultira u široj margini hiper-ravnine. Vidljivo je da je najbolji odabir linearne kernel funkcije za sve C vrijednosti, tako da ćemo taj kernel koristiti kod učenja modela.

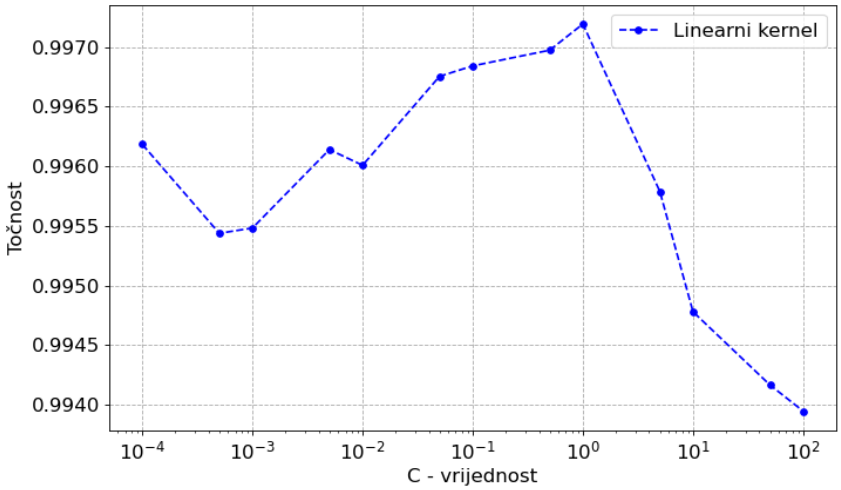


Slika 4.5: Ovisnost točnosti modela za različite C vrijednosti i za pet različitih kernel funkcija.

Sljedeći problem je izbor najbolje C vrijednosti. Graf koji prikazuje ovisnost točnosti linearnog kernela o C vrijednosti je prikazan na Slici 4.6. Točnost modela je dobra za sve vrijednosti C, tako da ćemo koristiti vrijednost $C = 1$ jer je to zadana vrijednost kod korištenja algoritma koja ima umjerenu toleranciju pogrešnih podataka.

Točnost	99.61%
Odziv	99.07%
Preciznost	78.62%

Tablica 4.2: Rezultatne vrijednosti za točnost, odziv i preciznost SVM algoritma za vrijednost $C = 1$



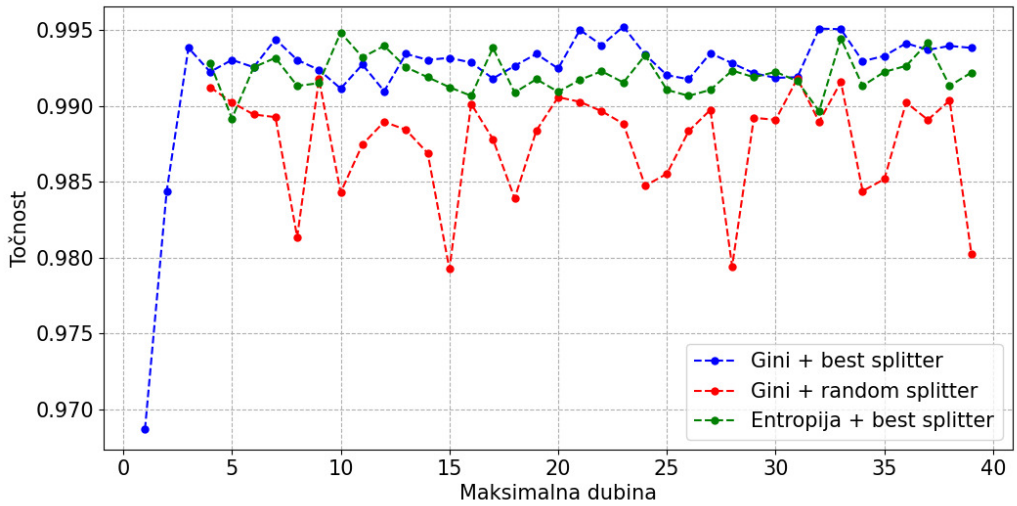
Slika 4.6: Ovisnost točnosti modela kod korištenja linearnog kernela za različite C vrijednosti.

Konačni rezultati za model učenja pomoću ovog algoritma su dani u Tablici 4.2. Kao i kod prethodno analiziranog algoritma, nizak rezultat za preciznost dolazi od nebalansiranog skupa podataka za testiranje.

4.5 Rezultati DTC algoritma

Klasifikacijsko stablo odlučivanja je algoritam koji uzastopno dijeli podatke na sve manje i manje grupe. U krajnjem slučaju algoritam će podijeliti podatke do te granice da neće postojati grupa koja sadrži dvije različite klasifikacije podataka. Dakle, dvije najbitnije stvari kod ovog algoritma su način podjele podataka i određivanje kada bi algoritam trebao stati s dodatnim grananjima. Što se tiče načina grananja, dva parametra su bitna, a to su određivanje kriterija (može biti Gini nečistoća ili entropija) i odabir karakteristike po kojoj se dijeli (odabir najbolje karakteristike ili odabir nasumične karakteristike). Za kontrolu zaustavljanja grananja ćemo koristiti maksimalnu dubinu stabla.

Graf koji prikazuje odnos ovih vrijednosti prikazan je na Slici 4.7. Na grafu je vidljivo da je najbolje koristiti način odabira grananja koji nije nasumičan, već automatski bira najbolju vrijednost. Zbog čistoće izgleda grafa nije prikazana kombinacija entropije i nasumičnog djelitelja jer je ta kombinacija bila jednako loša u točnosti kao i Gini varijanta. Ono što je manje očito je činjenica da je Gini nečistoća bolji izbor od entropije, što se pokazalo konzistentnom analizom prosječne vrijednosti točnosti, koja je uvijek bila marginalno bolja od entropije. Zaključak je da odabir Gini nečistoće nije ključan u procjeni naučenosti modela.



Slika 4.7: Ovisnost točnosti modela o načinima grananja za različite maksimalne dubine stabla.

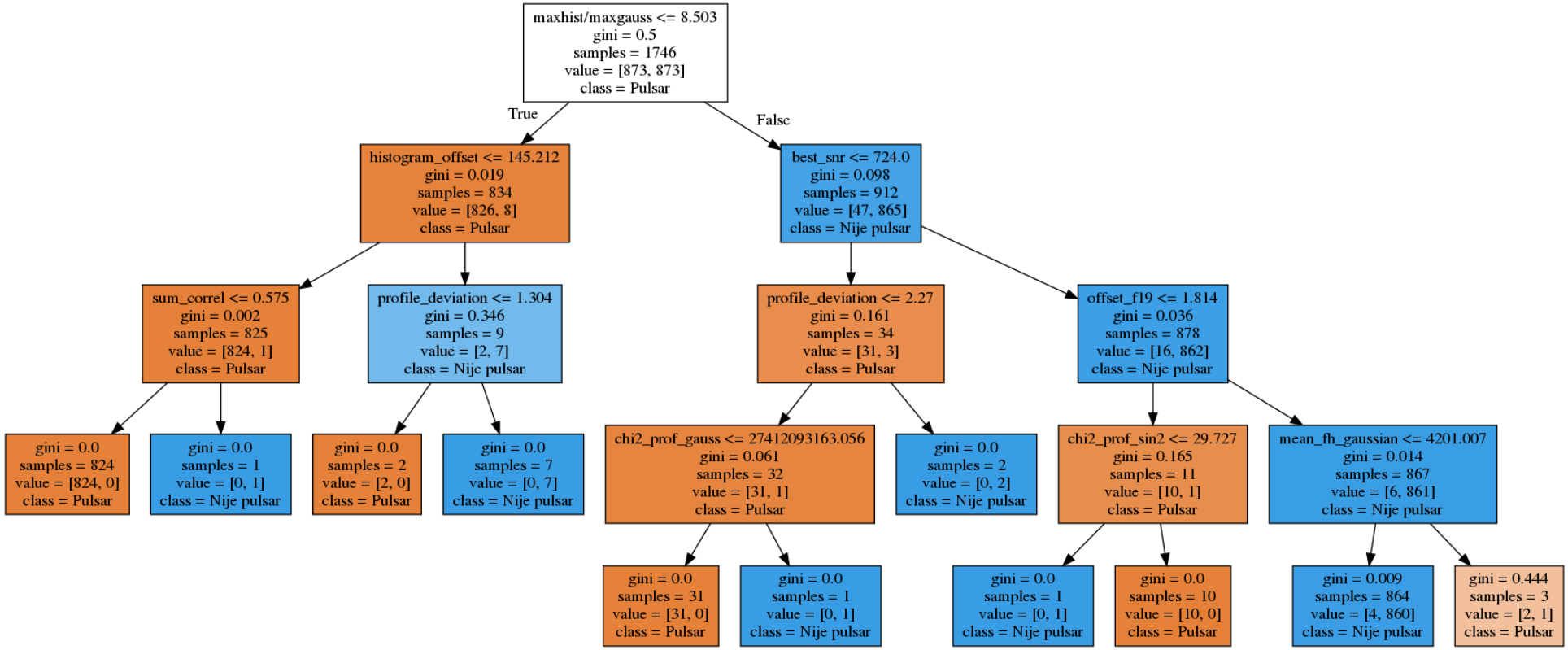
Što se tiče maksimalne dubine stabla, često se maksimalna vrijednost točnosti dobila kod ograničavanja dubine stabla na vrijednost između 15 i 20. Pretpostavka je da se kod tih vrijednosti sprečava prenaučenost modela tako što se ograničava daljnja konstrukcija stabla. Još jedna zanimljiva činjenica koja nije prikazana na grafu je ta da se za dubinu stabla $n = 1$ postiže točnost od čak 94.7%, što znači da su se samo jednim grananjem podaci vrlo dobro podijelili na dvije klasifikacijske grupe. Zaključak je da ćemo kao parametre koristiti Gini nečistoću u kombinaciji s najboljim načinom odabira parametra uz maksimalnu dubinu stabla $n = 19$. Izračunati rezultati za točnost, odziv i preciznost su dani u Tablici 4.3. Također, vizualizacija konstruiranog stabla za dubinu $n = 4$ je prikazana na Slici 4.8

Točnost	99.02%
Odziv	99.38%
Preciznost	59.23%

Tablica 4.3: Rezultatne vrijednosti za točnost, odziv i preciznost DTC algoritma.

4.6 Usporedba rezultata

Nakon određivanja najboljih vrijednosti za sva tri algoritma, analizirat ćemo ih zajedno. Sve tri mjere uspješnosti naučenog modela su dane na jednom mjestu u Tablici 4.4. Ponovno, razlog za niski postotak preciznosti algoritama je nebalansirani skup testnih podataka. Gledajući Tablicu 4.4 zaključili bi da algoritam potpornih vektora (SVM) ima najbolje rezultate. Sva tri algoritma se koriste u primjenama strojnog učenja. Međutim, za naš specifičan problem



Slika 4.8: Prikaz konstruiranog stabla dubine $n = 4$. Točnost ovog modela je 98.9%.

	KNN algoritam	SVM algoritam	DTC algoritam
Točnost	98.23%	99.61%	99.02%
Odziv	97.21%	99.07%	99.38%
Preciznost	45.31%	78.62%	59.23%

Tablica 4.4: Rezultatne vrijednosti za točnost, odziv i preciznost sva tri analizirana algoritma

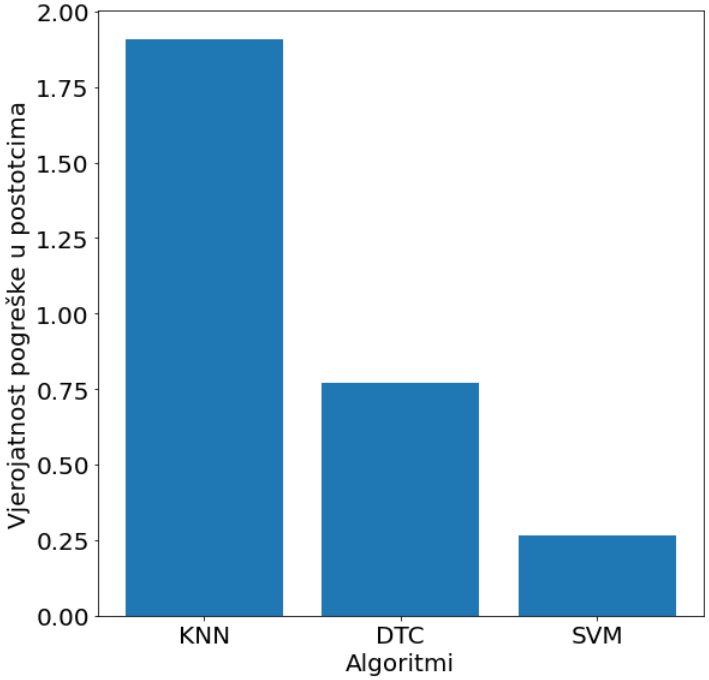
bilo bi dobro analizirati i neke dodatne indikatore naučenosti modela. Naime, analizirajući kandidate za pulsare, cilj nam je postići da nam algoritam točno pogodi samo one kandidate koji su pulsari. Drugim riječima, važno nam je da ne klasificiramo kandidata koji nije pulsar kao pulsar. Također i obrnuto nam je jednako važno, ne želimo klasificirati kandidata koji je zapravo pulsar kao da nije pulsar. Zbog toga gledamo definirane indikatore koji kažnjavaju ovakve slučajeve. Dva indikatora koja nam ukazuju na ovo su već analizirani "odziv" koji se također označava i kao TPR (engl. *True Positive Rate*), te "vjerojatnost pogreške" koji se označava sa FPR (engl. *False Positive Rate*). Definicije ove dvije vrijednosti su:

$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{FP + TN}. \tag{4.9}$$

Kao primjer, klasificiramo deset kandidata za pulsare, od kojih su prave vrijednosti takve da je pet kandidata pulsar, a pet nije. Naš naučeni model je otkrio pet kandidata koji su pulsari, međutim krivo je prepoznao jednog kandidata tako što ga je klasificirao kao pulsara, a on to zapravo nije. Drugim riječima pogodili smo četiri kandidata za pulsare, a "izgubili" smo jedan pravi pulsar tako što smo ga krivo klasificirali. Dvije bitne rezultantne vrijednosti za ovaj slučaj su:

$$TPR = \frac{TP}{TP + FN} = \frac{4}{4+1} = 0.8 \qquad FPR = \frac{FP}{FP + TN} = \frac{1}{1+4} = 0.2 \tag{4.10}$$

Kao što vidimo, cilj nam je maksimizirati odziv, odnosno TPR vrijednost, a minimizirati vjerojatnost pogreške, odnosno FPR vrijednost. Vrijednosti odziva smo već dobili, a vrijednosti vjerojatnosti pogreške su prikazani na Slici 4.9. Vidimo da svi algoritmi imaju dobre rezultate, ali nam algoritam potpornih vektora (SVM) daje najbolje vrijednosti u usporedbi s druga dva algoritma te nam je to najbolji odabir za učenje modela kategorizacije pulsara.



Slika 4.9: Vjerojatnosti pogreške za tri algoritma.

5 Zaključak

HTRU-1 baza podataka o kandidatima za pulsare se nakon balansiranja pokazala dobrim skupom podataka s korisnim parametrima za strojno učenje. Većina parametara je pokazivala dobru separabilnost između dvije kategorije kandidata. Pokazalo se da su odziv i vjerojatnost pogreške najbolji indikatori uspješnosti kategorizacije pulsara.

Za algoritam k-najbližih susjeda najbolji rezultati su dobiveni pridavanjem jednake važnosti svim podacima uz korištenje manhattanskog načina izračuna udaljenosti. Također se pokazalo da algoritam daje najveću točnost za male vrijednosti parametra k, te je prikazan zanimljiv uzorak kod parnih i neparnih odabira vrijednosti tog parametra. Izabrana je vrijednost $k = 5$. Dobivene vrijednosti za odziv i vjerojatnost pogreške su $TPR = 97.21\%$ i $FPR = 1.89\%$.

Za algoritam potpornih vektora najbolje vrijednosti ulaznih parametara su odabir linearnog kernela i vrijednosti $C = 1$. Dobivene vrijednosti za odziv i vjerojatnost pogreške su $TPR = 99.07\%$ i $FPR = 0.26\%$. Iako algoritam nema puno parametara koji bi se mogli podešavati, dobiveni rezultati su dobri.

Za algoritam klasifikacijskog stabla odlučivanja najbolje vrijednosti ulaznih parametara su korištenje Gini nečistoće uz opciju najboljeg odabira, a maksimalna dubina stabla je postavljena na $n = 19$ kako bi spriječili prenaučenost modela. Dobivene vrijednosti za odziv i vjerojatnost pogreške su $TPR = 99.38\%$ i $FPR = 0.76\%$.

Zaključak je da su svi algoritmi pokazali dobre rezultate i primjereni su za analizu kandidata za pulsare, ali najboljim odabirom se pokazao algoritam potpornih vektora s linearnom kernel funkcijom.

Dodaci

Dodatak A Pulsari u osnovnoj i srednjoj školi

U dječjim knjigama, slikovnicama i animiranim filmovima često se pojavljuje tematika svemira. Astronauti su uzor djeci, planeti i galaksije su im lijepi i zanimljivi, rakete su im fascinantne igračke, a prvo saznanje da je Sunce zapravo zvijezda im je uvijek iznenađujuće. Zainteresiranost im je uvijek visoka i imaju puno pitanja "Kako?" i "Zašto?". S ovom tematikom se učenici djelomično još u osnovnoj školi susreću kroz predmete poput "Prirode i društva", izbornih predmeta, te možda i u posjetu zvjezdarnici koje mnoge škole organiziraju.

Detaljnija razrada svemirske tematike se nažalost ne pojavljuje u programu fizike za 7. i 8. razred, a smatram da bi se mogla implementirati kao dodatno neobavezno gradivo koje ne bi ulazilo u nikakvu matematičku razradu i formulaciju. Veći fokus bi bio na zanimljivosti gradiva kako bi se učenike zainteresiralo za teme poput nastanka i razvoja svemira, formiranja galaksija i crnih rupa, različitih životnih ciklusa zvijezda, te čak i nastanka našeg planeta. Primjerice, učenike bi se moglo podijeliti u grupe, gdje bi svaka grupa obradila jednu od upravo navedenih tematika. Na ovaj način bi učenici koji obrađuju temu životnog ciklusa zvijezda mogli spomenuti i pulsare kao neutronske zvijezde koje nastaju kao posljedica supernove. Također, zvjezdarnica u Zagrebu nudi širok program predavanja za osnovnu školu i organizirane posjete za školske grupe, te se u slučaju vedrog vremena može gledati i kroz teleskop.

Jedna od najboljih aktivnosti vezanih uz istraživanje svemira, koja je dostupna učenicima osnovnih škola od 5. do 8. razeda kao i učenicima srednje škole, je "Astronomska ljetna škola" koju organizariraju Zvjezdarnica Zagreb i Hrvatsko astronomsko društvo. Ova ljetna škola se održava najčešće u mjesecu srpnju još od 1968. godine, a obično traje 7-8 dana tijekom kojih se održavaju predavanja i noćna promatranja svemira pomoću teleskopa. Ove 2020. godine se ljetna škola održala u mjestu Petehovac kod Delnica. Jedno od predavanja pod nazivom "Eksplozije u svemiru i kozmologija" se fokusiralo upravo na supernove i njihovu evoluciju [70]. Smatram da je ovo odlično organizirana izvannastavna aktivnost koja upoznaje učenike s temom svemira i tehnologijom koja omogućuje ta istraživanja poput teleskopa i radioastronomije.

Što se tiče generalne tematike svemira u srednjoj školi, smatram da se ova tema jako

oskudno obrađuje i to na samom kraju 4. razreda srednje škole kao posljednja cjelina. Lekcija o nastanku svemira i razvoju zvijezda je posljednja u srednjoškolskom školovanju iz fizike, te je učenici često ignoriraju zbog fokusa na pripremu ispita državne mature [71]. Ova tema zajedno s temama iz moderne fizike koje se tiču kvantne fizike su obično obrađene s fokusom na matematičkim formulacijama, te na taj način ignoriraju potencijalne filozofske rasprave koje su učenicima jako zanimljive, koje ih mogu zaintrigirati i o kojima često čitaju na internetu jer su vrlo popularne. Ovo bi se moglo postići organizacijom dodatnih sati iz fizike na kojima bi se obrađivale odabrane teme iz moderne fizike, na kojima bi učenici mogli sudjelovati u raspravama i razviti dublje razumijevanje kvantne fizike i svemira.

Dodatak B Strojno učenje u srednjoj školi

Strojno učenje je tema koja je u posljednje vrijeme vrlo popularna u medijima. Tako se npr. na YouTube platformi u mnoštvu edukativnih sadržaja mogu pronaći i zanimljive primjene strojnog učenja. Neke od njih su modeli koji su naučeni igrati razne logičke igre poput križić kružića, šaha i tetrisa [72]. Navedeni modeli vezani su uz naprednije nadzirano strojno učenje, koje je općenito za razumijevanje i implementaciju na satu možda ipak malo prekomplicirano za srednjoškolsko obrazovanje. Smatram da bi za uvod u strojno učenje bilo dobro primijeniti klasifikacijske algoritme, primarno algoritam k-najbližih susjeda jer ima shvatljiv i jednostavan princip rada. Zanimljivom primjenom ovog algoritma potaknuo bih učenički interes za strojno učenje i samim time zainteresirao učenike za naprednije algoritme.

Blok sat na kojem bi učenicima približio strojno učenje, organizirao bih u četvrtom razredu srednje škole, kada su učenici već upoznati sa strukturama podataka i algoritmima u programskom jeziku Python [73]. U uvodnom dijelu sata učenicima bih pričao o nekim zanimljivim primjenama strojnog učenja te im objasnio što točno rade klasifikacijski algoritmi. Nakon što bih učenicima objasnio koji je cilj tih algoritama, pitao bih imaju li ideju kako bi riješili problem podjele ulaznih podataka na dvije kategorije. Prikupljao bih učeničke prijedloge te bi ih postepeno, uz potpitanja, navodio na ideju da bi trebalo gledati u kakvom se to ”susjedstvu” nalazi neki novi ulazni podatak.

Nakon što učenici dobiju osnovne informacije o principu rada algoritma k-najbližih susjeda, objasnio bih im značenje faktora k. Objasnio bih da je to samo jedan od algoritama koji su već implementirani u *scikit-learn* paketu koji se može vrlo lako primijeniti u nekom novom projektu. Predložio bih učenicima da otvore novu datoteku u Pythonu te da me prate

u pisanju koda. Grupni rad učenika zamislio sam tako da svaka grupa izabere jedan od dostupnih skupova podataka. Neki od zanimljivih podataka su primjerice skup muških i ženskih lica, skup lica koja se smiju i koja se ne smiju, YouTube statistike, magnetske snimke pacijenata s rakom i bez raka i slično [74]. Kada bi grupa odabrala bazu podataka, koja naravno ima kategorizacijske parametre, objasnio bih kako koristiti datoteku u programu te od tih podataka pripremiti obradive skupove, poput skupova za treniranje i testiranje. Nakon ovoga svaka skupina učenika pokreće algoritam te analizira rezultate, odnosno vrijednosti poput konačne točnosti naučenog modela. Ako je to moguće, svaka grupa bi mogla dodati i analizirati nove podatke. Primjerice, ako naučeni model ima veze s prepoznavanjem osmjeha, tada bi učenici mogli dodati svoju sliku na kojoj se smiju i vidjeti hoće li algoritam točno prepoznati osmijeh.

Nakon što grupe obrade svoje podatke, zatražio bih da odaberu predstavnika koji će ostatku razreda prezentirati rezultate i objasniti skup nad kojim su radili učenje modela. Za kraj sata postavio bih učenicima pitanja kako bih dobio uvid u njihov način razmišljanja. Primjerice, što se događa kada je $k = 2$, a podaci im zahtijevaju binarnu klasifikaciju, kako onda algoritam odluči kojoj skupini pripada podatak i kako bi jednostavno riješili tu situaciju. Konačno, spomenuo bih ukratko i neke druge algoritme koji se koriste te bih preporučio zainteresiranim učenicima da ih primjene na neku bazu podataka.

Literatura

[1] McNamara, G.: Clocks in the Sky - The Story of Pulsars. Berlin : Springer, 2008.

[2] Data, Data Everywhere, https://tinyurl.com/y4sg7sms, 30. 8. 2020.

[3] Baade, W.; Zwicky, F.: On Super-Novae // Proc. Natl. Acad. Sci. USA. 20 (1934), str. 254-259.

[4] Baade, W.; Zwicky, F.: Remarks on Super-Novae and Cosmic Rays. // Phys. Rev. 46 (1934), str. 77.

[5] Shields, G. A.: A Brief History of AGN, https://tinyurl.com/y8topuvd, 14. 6. 2020.

[6] Chiu, H. Y.: Gravitational Collapse // Physics Today, vol. 17, issue 5, p. 21 (1964), str. 21.

[7] Hewish, A.: Pulsars and High Density Physics // Science 188 (1975.), str. 180.

[8] Bell Burnell, S. J.: So Few Pulsars, So Few Females // Science, 304 (2004.), str. 1.

[9] Gold, T. : Rotating Neutron Stars as the Origin of the Pulsating Radio Sources // Nature, 218(5143) (1968.), str. 732.

[10] The Nobel Prize in Physics 1974, https://tinyurl.com/y8wfoqvo, 15. 6. 2020.

[11] Bell Burnell, S. J.: Little Green Men, White Dwarfs or Pulsars? https://tinyurl.com/yck9dxzo, 15. 6. 2020.

[12] Cofield, C.: What Are Pulsars?, https://tinyurl.com/y75tp4us, 15. 6. 2020.

[13] Olson, B.: Fusion Regulation in the Sun, https://tinyurl.com/y88qxlv2, 17. 6. 2020.

[14] NASA Goddard Space Flight Center, Imagine the Universe!, Stars, https://tinyurl.com/v5vnook, 15. 6. 2020.

[15] University of Maryland, Introductory Astronomy: Main Sequence Stars, https://tinyurl.com/yaxkwb7g, 15. 6. 2020.

[16] NASA Goddard Space Flight Center, Imagine the Universe!, White Dwarfs, https://tinyurl.com/jatmxep, 15. 6. 2020.

[17] Lisakov, S. : Core-collapse supernovae and their progenitors // Université Côte d'Azur, 2018, str. 13-16, https://tinyurl.com/ya52aarx.

[18] Pogge, R.: Lecture 18, Supernovae, https://tinyurl.com/y7znvz7o, 15. 6. 2020.

[19] NASA Fermi Gamma-ray Space Telescope, Neutron Stars, https://tinyurl.com/y9tvnq64, 15. 6. 2020.

[20] National Radio Astronomy Observatory, Pulsar properties, https://tinyurl.com/yc4lsw4r, 15. 6. 2020.

[21] Lorimer, D. R.; Kramer M. : Handbook of Pulsar Astronomy. Cambridge : Cambridge University Press, 2005.

[22] AAS NOVA, An Infant Pulsar Defies Categorization, Sky and Telescope, https://tinyurl.com/y2kqdw8b, 16. 8. 2020

[23] The Nobel Prize in Physics 1993, https://tinyurl.com/yd7cjdt3, 15. 6. 2020.

[24] Backer, D. C.; Kulkarni, S. R.; Heiles, C.; Davis, M. M.; Goss, W. M. : A millisecond pulsar // Nature 300 (1982.), str. 615.

[25] Manchester, R. N.; Hobbs, G. B.; Teoh, A.; Hobbs, M. : The ATNF Pulsar Catalogue // Astronomical Journal 129, 1993 (2005)

[26] The Updated ATNF Pulsar Catalogue, https://tinyurl.com/y4t4qj5e, 15. 6. 2020.

[27] Scaife, A.: The Pulsar Classification Problem, https://tinyurl.com/y9pyccu7, 17. 6. 2020.

[28] Ransom, S.: Pulsars, https://tinyurl.com/yd8snkt7, 15. 6. 2020.

[29] Thompson, A.: Google's Mission Statement and Vision Statement (An Analysis), https://tinyurl.com/y44nleex, 20. 7. 2020.

[30] Reinsel, D.; Gantz, J; Rydning, J.: The Digitization of the World, From Edge to Core, https://tinyurl.com/y95ogat4, 20. 7. 2020.

[31] Jordan, M. I.; Mitchell, T. M.: Machine learning: Trends, perspectives, and prospects. // Science. Vol. 349 (2015), str. 255-260

[32] Müller, A. C.; Guido, S.: Introduction to Machine Learning with Python. Sebastopol: O'Reilly Media

[33] Kuhn, M.: The Caret Package, https://tinyurl.com/y5st3654, 16. 8. 2020.

[34] Scikit-learn Package, https://tinyurl.com/yxtdzsj6, 16. 8. 2020.

[35] Kozyrkov, C.: When not to use machine learning or AI, https://tinyurl.com/yagotkfe, 16. 8. 2020.

[36] Rosenblatt, F. : The Perceptron - a perceiving and recognizing automaton // Cornell Aeronautical Laboratory (Report No. 85-460-1), 1957.

[37] Loiseau, J. B.: Rosenblatt's perceptron, the first modern neural network, https://tinyurl.com/y2ccfyrc, 16. 8. 2020.

[38] The MathWorks, Perceptron, Limitations and Cautions, https://tinyurl.com/y6hfua4u, 16. 8. 2020.

[39] Martinez, H. I.: Train/Test Split, Cross-Validation, and You, https://tinyurl.com/y5rt3h3s, 16. 8. 2020.

[40] Bronshtein, A.: Train/Test Split and Cross Validation in Python, https://tinyurl.com/yafp4n28, 16. 8. 2020.

[41] Kasturi, S. N.: Underfitting and Overfitting in machine learning and how to deal with it, https://tinyurl.com/y38ca3qz, 16. 8. 2020.

[42] Riggio, C.: What's the deal with Accuracy, Precision, Recall and F1?, https://tinyurl.com/y3ko4z29, 16. 8. 2020.

[43] A Short Introduction to K-Nearest Neighbors Algorithm, https://tinyurl.com/y22kgbsy, 16. 8. 2020.

[44] Scikit-learn documentation: Nearest Neighbors Classification, https://tinyurl.com/y3dp6d5f, 16. 8. 2020.

[45] Scikit-learn documentation: KNeighborsClassifier, https://tinyurl.com/qqwlpbh, 16. 8. 2020.

[46] Chatterjee, M.: A Quick Introduction to KNN Algorithm, https://tinyurl.com/y6qxzfey, 16. 8. 2020.

[47] Gandhi, R.: Support Vector Machine - Introduction to Machine Learning Algorithms, https://tinyurl.com/y43udev6, 16. 8. 2020.

[48] Misra, R.: Support Vector Machines - Soft Margin Formulation and Kernel Trick, https://tinyurl.com/y4ong2jd, 16. 8. 2020.

[49] Leskovec, J.: Stanford University Course CS246: Mining Massive Data Sets, Lecture 70: Soft-Margin SVMs, https://tinyurl.com/yyq4l9ul, 16. 8. 2020.

[50] Packt, What is a support vector machine?, https://tinyurl.com/yye5j47u, 16. 8. 2020.

[51] Deepthi, A. R.: Support Vector Machines and Imbalanced Data, https://tinyurl.com/y6xlwqrx, 16. 8. 2020.

[52] Fouquet, C.; Histace, A.; Duvaut, P.; Automated Classification of Defect Signatures in Pipelines using Ultrasonic Images, https://tinyurl.com/y2argkpc, 16. 8. 2020.

[53] Patel, S.: Chapter 3, Decision Tree Classifier - Theory, https://tinyurl.com/y5tfo2gl, 16. 8. 2020.

[54] Zhou, V.: A Simple Explanation of Gini Impurity, https://tinyurl.com/y5gya7av, 16. 8. 2020.

[55] Raschka, S.: What are the advantages of the different impurity metrics?, https://tinyurl.com/yxnxvzdr, 1. 9. 2020.

[56] Scikit-learn Documentation: Post pruning decision trees with cost complexity pruning, https://tinyurl.com/yyvz5zbs, 16. 8. 2020.

[57] Dhiraj, K.: Top 5 advantages and disadvantages of Decision Tree Algorithm, https://tinyurl.com/y7cqa4zg, 16. 8. 2020.

[58] Python General F.A.Q., https://tinyurl.com/y6xhmyek, 29. 8. 2020.

[59] Jupyter Notebook Main Page, https://tinyurl.com/yypqbost, 29. 8. 2020.

[60] Numpy Package Main Page, https://tinyurl.com/yxz8qfjq, 29. 8. 2020.

[61] Pandas Package About Page, https://tinyurl.com/y4kkephc, 29. 8. 2020.

[62] Scikit-learn Project Main Page, https://tinyurl.com/y5umb372, 29. 8. 2020.

[63] Matplotlib Project Examples Page, https://tinyurl.com/y37cprjj, 29. 8. 2020.

[64] Seaborn: statistical data visualization, https://seaborn.pydata.org, 29. 8. 2020.

[65] Keith M. J., Jameson A. : The High Time Resolution Universe Pulsar Survey – I. System configuration and initial discoveries // Monthly Notices of the Royal Astronomical Society 409 (2010), str. 619-627

[66] Lyon, R. J.; Stappers, B.W.; Cooper, S.; Brooke, J. M.; Knowles, J. D.: Fifty Years of Pulsar Candidate Selection: From simple filters to a new principled real-time classification approach // Monthly Notices of the Royal Astronomical Society 459 (2016), str. 1104-1123

[67] Pulsar Feature Lab - GitHub Repository, https://tinyurl.com/y2ax9bc4, 29. 8. 2020.

[68] Ceballos, F.: Searching for Pulsars with Machine Learning, https://tinyurl.com/yyxm9wgw, 29. 8. 2020.

[69] K-Nearest Neighbors for Machine Learning, https://tinyurl.com/y8fh9fgn, 29. 8. 2020.

[70] Izvješće o ostvarenom programu 51. astronomske ljetne škole, https://tinyurl.com/y297r4kb, 1. 9. 2020.

[71] Kurikulum nastavnog predmeta fizika za osnovne škole i gimnazije, https://tinyurl.com/y48mrjze, 1. 9. 2020.

[72] A.I. Learns to Play Tetris, https://tinyurl.com/y3aojv6j, 2. 9. 2020.

[73] Kurikulum nastavnog predmeta informatika za osnovne škole i gimnazije, https://tinyurl.com/yxs5pjgj, 1. 9. 2020.

[74] Kaggle Datasets, https://tinyurl.com/hotl42a, 2. 9. 2020.