Univerzitet Crne Gore
Elektrotehnički fakultet

Danilo Planinić

DETEKCIJA ZLOUPOTREBA PLATNIH KARTICA UPOTREBOM ALGORITAMA KLASIFIKACIJE

– MASTER RAD –

Podgorica, 2024. godine

PODACI I INFORMACIJE O MAGISTRANDU

Ime i prezime: **Danilo Planinić**

Datum i mjesto rođenja: **8. jun 1999. godine, Podgorica**

Naziv završenog osnovnog studijskog programa i godina završetka studija: **Elektrotehnički fakultet, osnovne akademske studije, studijski program: Elektronika, telekomunikacije i računari, 2021. godine**

INFORMACIJE O MASTER RADU

Naziv master studija: **Master studije, studijski program Računari**

Naslov rada: **Detekcija zloupotreba platnih kartica upotrebom algoritama klasifikacije**

Fakultet na kojem je rad odbranjen: **Elektrotehnički fakultet Podgorica**

UDK, OCJENA I ODBRANA MASTER RADA

Datum prijave master rada: **13. septembar 2023. godine**

Datum sjednice Vijeća na kojoj je prihvaćena tema: **16. novembar 2023. godine**

Mentor: **Prof. dr Vesna Popović-Bugarin**

Komisija za ocjenu/odbranu rada:

1. **Prof. dr Miloš Daković, ETF Podgorica**
2. **Prof. dr Vesna Popović-Bugarin, ETF Podgorica**
3. **Doc. dr Miloš Brajović, ETF Podgorica**

Datum odbrane: **8. novembar 2024. godine**

Ime i prezime autora: Danilo Planinić

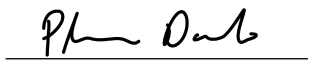
Etička izjava

U skladu sa članom 22 Zakona o akademskom integritetu i članom 18 Pravila studiranja na master studijama, pod krivičnom i materijalnom odgovornošću, izjavljujem da je magistarski rad pod naslovom:

„Detekcija zloupotreba platnih kartica upotrebom algoritama klasifikacije"

moje originalno djelo.

Podnosilac izjave:



Danilo Planinić

Podgorica, 20. septembar 2024. godine

Predgovor

Sa razvojem interneta i digitalizacije većine poslovnih procesa, platne kartice su dobile veoma bitnu ulogu u globalnoj ekonomiji. Jednostavnost i lakoća korišćenja platnih kartica, kao i sve veća trgovina preko internet sajtova su uticali da platna kartica postane dominantan način plaćanja robe i usluga. Međutim, sa porastom upotrebe platnih kartica, raste i težnja ka njihovoj zloupotrebi, što predstavlja veliku opasnost, kako za korisnike, tako i za poslovanje banaka i kompanija u cjelini.

Cilj ovog rada jeste pružanje sveobuhvatnog uvida u cjelokupan proces izgradnje sistema, zasnovanog na metodama mašinskog učenja, čiji je cilj blagovremena detekcija zloupotreba, koja bi pružila mogućnost reagovanja i sprečavanja istih. Brojnim regulativama, banke i države nastoje zaštiti korisnike, međutim počinitelji zloupotreba platnih kartica se prilagođavaju ovim promjenama, stoga je neophodno kreirati efikasan sistem koji će alarmirati banke i zaštiti korisnike.

Ovo istraživanje se bavi analizom performansi različitih algoritama klasifikacije nadgledanog mašinskog učenja, njihovim prednostima i manama, kao i izazovima sa kojima se susrijeću sistemi za detekciju zloupotreba platnih kartica. Rad takođe obuhvata detaljne instrukcije pripreme sirovog seta podataka do forme pogodne za upotrebu različitih algoritama, čime se pružaju teorijske i praktične smjernice za sve one koje interesuje primjena ove tehnologije u finansijskom poslovanju.

Srdačno se zahvaljujem svojoj profesorici Vesni Popović-Bugarin koja me prije svega podstakla da se bavim ovom veoma interesantnom i aktuelnom temom, kao i svojim savjetima i pruženom pomoći usmjeravala ka što efikasnijem i kvalitetnijem pristupu rješavanju ovog problema.

Sažetak

Razvoj interneta i *online* trgovine povećali su upotrebu platnih kartica, ali i rizik od zloupotreba. Ovaj rad koristi algoritme klasifikacije u cilju detekcije zloupotreba platnih kartica. Testirane su performanse osam aktuelnih algoritama nadgledanog mašinskog učenja, sa posebnim akcentom na uticaj primjene tehnika balansiranja seta podataka i optimizacije hiperparametara na njihove performanse. Rad pruža sveobuhvatan pristup pretprocesiranju podataka, obuhvatajući sve ključne tehnike potrebne za pripremu seta podataka u cilju primjene algoritama klasifikacije. Kao metrika za evaluaciju performansi dobijenih modela korišćena je F_1 mjera. Rezultati pokazuju da su algoritmi zasnovani na stablima odlučivanja izuzetno efikasni za dati problem, pri čemu se posebno ističu algoritmi *Random Forest*, XGBoost i Catboost. Najbolji rezultat $F_1 = 0.8694$, uz najkraće vrijeme treniranja modela i predikcije je zabilježio CatBoost model sa optimizovanim hiperparametrima i primjenom tehnike slučajnog preodabiranja. Dodatno su testirani uticaji promjene praga odlučivanja na mogućnost detekcije zloupotreba i preciznost odabranog modela.

Ključne riječi: zloupotreba platnih kartica, algoritmi klasifikacije, CatBoost, tehnike balansiranja, F_1 mjera

Abstract

The development of the internet and online commerce has increased the use of payment cards, but also the risk of fraud. This paper utilizes classification algorithms to detect payment card fraud. The performance of eight current supervised machine learning algorithms was tested, with special emphasis on the impact of applying data balancing techniques and hyperparameter optimization on their performance. The paper provides a comprehensive approach to data preprocessing, covering all the key techniques needed to prepare the dataset for the application of classification algorithms. The F_1 score was used as the metric for evaluating the performance of the models. The results show that decision tree-based algorithms are particularly effective for this problem, with Random Forest, XGBoost, and Catboost standing out. The best result, $F_1 = 0.8694$, along with the shortest training and prediction time, was achieved by the CatBoost model with optimized hyperparameters and the application of the random undersampling technique. Additionally, the effects of changing the decision threshold on the ability to detect fraud and the precision of the selected model were tested.

Keywords: payment card fraud, classification algorithms, CatBoost, balancing techniques, F_1 score

Sadržaj

1 Uvod

Zloupotrebom platnih kartica se smatra svaka upotreba platnih kartica u svrhu sticanja ličnog benefita, bez znanja vlasnika kartice. Istraživanje "UK Finance"rađeno 2023. godine [1] ističe da ovo predstavlja jedan od vodećih uzroka gubitaka u finansijskoj industriji. Sprečavanje zloupotreba stoga predstavlja jedan od prioriteta banaka, koje su različitim regulativama i metodama zaštite uspjeli spriječiti veliki broj neovlašćenih transakcija u 2022. godini koje bi dovele do otuđenja 1.2 milijarde funti. Međutim, u istraživanju se takođe navodi da je samo u Velikoj Britaniji tokom 2022. godine, posredstvom neovlašćene upotrebe platnih kartica, otuđeno 726.9 miliona funti. Ovo predstavlja veliki problem kako za korisnike čiji je novac otuđen, tako i za banke čija se pouzdanost i bezbjednost dovodi u pitanje.

Veliki broj banaka u detekciji zloupotreba koristi sistem zasnovan na pravilima [2]. Ova pravila utvrđuje ekspert temeljnom analizom istorijata transakcija, ispitujući zakonitosti po kojima se određena transakcija može smatrati zloupotrebom. Međutim, analiza velikog broja transakcija opisanih brojnim karakteristikama je vremenski veoma zahtjevan posao. Takođe, sistem baziran na pravilima ne bi bio u stanju da se prilagodi čestim promjenama u ponašanju korisnika i vršioca prevara. Sistem baziran na algoritmima klasifikacije mašinskog učenja, koji mogu analizirati veliku količinu informacija o transakcijama (vrijeme, iznos, lokacija, proizvod, usluga,...) i drugim relevantnim podacima vezanim za korisnika, se stoga nameće kao rješenje u cilju identifikovanja sumnjivih aktivnosti koje odskaču od navika korisnika i ukazuju na moguću zloupotrebu platnih kartica [3][4]. Mogućnost obrade velike količine podataka u veoma kratkom vremenu i donošenje brze odluke čini ove algoritme boljim od tradicionalnih metoda detekcije zloupotreba zasnovanim na unaprijed definisanim pravilima i statističkoj analizi, koje mogu biti ograničene u prepoznavanju kompleksnih aktivnosti počinitelja zloupotreba.

Identifikovanje sumnjivih aktivnosti predstavlja svojevrsni izazov i za algoritme mašinskog učenja zbog promjene distribucije podataka koje treba obraditi usljed novih metoda napada i varijabilnosti korišćenja kartica u zavisnosti od doba godine (praznici, godišnji odmori), kao i promjena navika korisnika. Takođe, veoma mali broj zloupotreba u odnosu na broj regularnih transakcija otežava algoritmima otkrivanje šablona i pravilnosti na osnovu kojih se neka transakcija može proglasiti sumnjivom. Navedeni izazovi, kao i nužnost poboljšanja bezbjednosti u finansijskom sektoru naglašavaju potrebu za modifikacijom i unapređenjem trenutnih metoda i kreiranjem novih algoritama.

U zavisnosti od strukture podataka i samog pristupa detektovanju zloupotreba razlikujemo primjene algoritama klasifikacije nadgledanog (eng. *supervised learning*)

i nenadgledanog (eng. *unsupervised learning*) mašinskog učenja. Kod nadgledanog mašinskog učenja model na osnovu označenog seta podatka uči kako da klasifikuje nove transakcije [5], dok su nenadgledani algoritmi mašinskog učenja u stanju da na osnovu prethodnih, neoznačenih transakcija detektuju one koje na osnovu svojih karakteristika odskaču od ostalih [6], odnosno odskaču od uobičajenih navika korisnika.

Algoritmi klasifikacije nadgledanog učenja poput logističke regresije (eng. *Logistic Regression* - LR), K-najbližih susjeda (eng. *K-Nearest Neighbours* - KNN) i naivnog Bajes klasifikatora (eng. *Naive Bayes* - NB) su korišćeni za detekciju zloupotreba u radu [7]. U ovom radu je jednostavan algoritam poput KNN-a, koji donosi odluku na osnovu sličnosti među transakcijama, postigao izuzetne rezultate kojim je nadmašio ostale algoritme. U radu [8] su upoređivane sposobnost klasifikacije transakcija i predikcije zloupotreba algoritama LR, stabala odlučivanja (eng. *Decision Tree* - DT) i slučajne šume (eng. *Random Forest* - RF), prilikom čega se RF istakao kao najbolja opcija. Takođe, ovaj algoritam je u radu [9] nadmašio performanse NB-a i višeslojnog perceptrona (eng. *Multi-Layer Perceptron* - MLP) koji predstavlja malu neuralnu mrežu.

Kako je broj zloupotreba znatno manji od broja regularnih transakcija (često ispod 0.1%) [10], broj regularnih transakcija znatno dominira, stoga je odnos broja uzoraka koji pripadaju posmatranim klasama neizbalansiran. U radu [11] su opisane brojne tehnike odabiranja koje utiču na broj uzoraka manjinske, većinske klase ili kombinovano. Osim ovih tehnika koje nastoje riješiti problem neizbalansiranosti koja negativno utiče na performanse modela, rad [11] takođe navodi mogućnost rješavanja problema neizbalansiranosti na nivou algoritama. Da balansiranje seta podataka prije korišćenja algoritama vodi njihovim boljim performansama može se vidjeti u radu [12], gdje se posebno ističu tehnike balansiranja na bazi sintetičkog preodabiranja manjinske klase (eng. *Synthetic Minority Over-sampling TEchnique* - SMOTE) [13].

Koja kombinacija tehnike odabiranja i algoritma u kreiranju modela donose optimalne rezultate provjeravaju Alfaiz i Fati u radu [3] kombinujući 19 tehnika balansiranja i 9 različitih klasifikatora. Pokazuje se da CatBoost (*Categorical Boosting*) uz balansiranje seta primjenom tehnike svih najbližih susjeda (eng. *All K-Nearest Neighbours* - AllKNN) postiže najbolje rezultate. Da izjednačavanje broja uzoraka obje klase nije uvijek poželjno, te da je potrebno ispitati i druge odnose broja uzoraka dvije klase nakon primjene tehnika balansiranja, pokazuje se u radovima [4], [7].

Uspješne performanse ansambl metoda (eng. *ensemble methods*) baziranih na

stablima odlučivanja, poput RF-a i CatBoost-a u radovima [8, 9, 13, 14], istaknule su ovu grupu algoritama kao primaran izbor u detekciji zloupotreba. U radu [4] vrši se njihovo poređenje, pri čemu se ističu prednosti RF-a, CatBoost-a i XGBoost-a (*Extreme Gradient Boosting*) u odnosu na LightGBM (*Light Gradient Boosting Machine*), koji takođe i u radu [5] uprkos optimizaciji ne postiže dobre rezultate.

Osim kombinovanja slabih klasifikatora u ansambl metode, rad [2] predlaže sekvencijalnu primjenu dva nezavisna algoritma nadgledanog mašinskog učenja, pri čemu će drugi algoritam učiti na dobro klasifikovanim uzorcima prvog klasifikatora. Carcillo u radu [15] koristi metode nenadgledanog mašinskog učenja kako bi dostupni označeni set podataka učinio informisanijim, što će obezbijediti bolje performanse algoritama nadgledanog učenja. Takođe, u radu [16] se pokazuju mogućnosti rješavanja problema neizbalansiranosti primjenom dubokog učenja (eng. *Deep Learning* - DL) za generisanje uzoraka manjinske klase, dok su Jurgovsky i saradnici tretirali problem detekcije zloupotreba platnih kartica kao klasifikaciju sekvenci u okviru nadgledanog mašinskog učenja. Za klasifikaciju su koristili LSTM (*Long Short-Term Memory*) mreže koje predstavljaju poseban tip rekurentnih neuralnih mreža (eng. *Recurrent Neural Networks*), dizajniran da bolje uči i modelira sekvencijalne podatke.

Da se detektovanje zloupotreba platnih kartica, osim primjene metoda mašinskog učenja, mogu otkriti njihovom analizom u frekvencijskom domenu pokazuju Saia i Carta u radu [17]. Ovdje se predlaže rješavanje problema neizbalansiranosti primjenom modela zasnovanog na diskretnoj Furijeovoj transformaciji (eng. *Discrete Fourier Transform* - DFT), koja se koristi za prepoznavanje frekvencijskih obrazaca. Ovaj pristup koji analizira frekvencijski domen, osim mogućnosti da se bolje suoči sa problemom neizbalansiranosti, manje je podložan heterogenosti podataka. Slični motivi su podstakli Saia da predloži novi metod zasnovan na vejvletima (eng. wavelets) [18], koji je postigao zavidne rezultate.

2 Mašinsko učenje

Arthur Samuel je 1959. godine opisao mašinsko učenje kao polje istraživanja koje pruža mogućnost učenja računarima bez eksplicitnog programiranja. Za razliku od tradicionalnog programiranja, gdje postoje jasno utvrđena pravila na osnovu kojih se donosi odluka, mašinsko učenje je veoma pogodno u situacijama kada je potrebno utvrditi zakonitosti na osnovu dostupnih podataka.

Matematički korektniju definiciju sposobnosti računara da uči dao je Tom Michell 1997. godine. U njoj ističe da se za određeni kompjuterski program može reći da uči iz iskustva (eng. *experience* - E) u odnosu na određeni zadatak (eng. *task* - T) i mjeru performansi (eng. *performance* - P), ukoliko se njegove performanse mjerene sa P na određenom zadatku T popravljaju sa iskustvom E.

Razvoj mašinskog učenja, kao grane vještačke inteligencije, datira još od 1943. godine kada su Walter Pitts i Warren McCulloch u svom radu [19] prezentovali matematički model neuralne mreže sa ciljem oponašanja donošenja odluka od strane ljudi. Par godina kasnije, Alan Turing u svom radu [20] razmatra mogućnost učenja računara na osnovu iskustva i njegovo prilagođavanje novim okolnostima. Njegovo pitanje koje se odnosi na mogućnost računara da misli je dovelo do kreiranja Turingovog testa. Test razmatra mogućnost razlikovanja tekstualnih odgovora od strane računara i čovjeka, i pokazuje koliko je zaista računar inteligentan. Razvoj mašinskog učenja su pratile brojne faze velikog napretka koje su najčešće bile sputavane zbog softverskih i hardverskih ograničenja. Prva veoma značajna primjena mašinskog učenja, koja je imala veliki uticaj na živote svih ljudi, jeste filter neželjene pošte (eng. *spam filter*) 1990. godine. Ovaj program je na osnovu velikog broja mejlova koji čine trening set, gdje svaki mejl predstavlja jedan trening uzorak za koji se zna da li je neželjena pošta ili ne, pokušavao naučiti zakonitosti po kojima bi bio u stanju da sam detektuje pojavu *spam*-a. Ukoliko bi se primijenila definicija Tom Michell-a, T bi predstavljao zadatak detekcije *spam*-a, E bi se odnosilo na informacije značajne za klasifikaciju koje su sadržane u trening setu, dok bi program bio evaluiran metrikom P, koja bi mogla biti procenat dobro detektovanih *spam* mejlova.

Danas se primjeri primjena mašinskog učenja mogu naći svuda oko nas. Pretraživanje na internetu, traženje filmova na Netflix-u, slušanje muzike na YouTube-u i Spotify-u je uveliko olakšano primjenom algoritama koji će na osnovu ličnih potreba i interesovanja korisnika preporučiti što je ono što bi ih moglo interesovati. Brojne kompanije koriste različite algoritme kako bi na osnovu podataka o korisnicima, njihovim željama i navikama, utvrdile ciljnu grupu kojoj treba da plasiraju svoje proizvode.

Mogućnost prikupljanja i skladištenja velikog broja podataka o pacijentima, o njihovom zdravstvenom stanju, istoriji bolesti i dijagnozama omogućili su algoritmima mašinskog učenja izvlačenje medicinskog znanja. Ovaj primjer otkrivanja znanja iz podataka i prepoznavanja korelacija među njima predstavlja rudarenje podataka (eng. *data-mining*). Ono obuhvata čitavu proceduru pripreme i obrade podataka, primjenu algoritama mašinskog učenja i vizuelizaciju rezultata. Izvučeno znanje iz velikog seta podataka, osim što služi za donošenje odluke i potvrđivanje odluka stručnih lica, može ukazati na neke nove zavisnosti i proširiti dotadašnje znanje iz određene oblasti. Ovo posebno dolazi do izražaja pri primjeni algoritama u analiziranju genetskih podataka osoba radi definisanja personalizovane terapije i predikcije potencijalne bolesti.

Razumijevanje govora i teksta, koliko god ljudima prirodno i jednostavno djelovalo, nije moguće opisati tradicionalnim algoritmima. Korišćenjem naprednih algoritama mašinskog učenja, poput neuralnih mreža (eng. *neural networks*) i dubokog učenja, koji su u stanju da otkriju veoma kompleksne obrasce u podacima, ovo je postalo moguće. Danas imamo mogućnost da pričamo sa virtualnim asistentima (Siri, Alexa, Cortana) i dopisujemo se sa *chatbot*-ovima poput ChatGPT-a koji analiziraju i razumiju prirodni jezik. Osim teksta i govora, danas je moguće analizirati slike, što je veoma značajno u medicini gdje se analizom medicinskih slika mogu detektovati različite vrste tumora i bolesti.

U zavisnosti od toga da li se modeli mašinskog učenja razvijaju na osnovu označenih podataka gdje je ciljna vrijednost koju model treba da estimira unaprijed poznata, ili bez njih, razlikujemo:

- Nadgledano mašinsko učenje;
- Nenadgledano mašinsko učenje;
- Polu-nadgledano mašinsko učenje (eng. *Semi-supervised learning*);
- Učenje sa podrškom (eng. *Reinforcement learning*);

Nadgledano mašinsko učenje predstavlja kategoriju mašinskog učenja gdje algoritmi uče na trening setu koji se sastoji od prethodno označenih podataka. Algoritmu se daju ulazni podaci i željene vrijednosti izlaza, pri čemu je njegov glavni cilj izgradnja modela koji će predviđati izlazne vrijednosti za nove, nepoznate podatke. Kako bi postigao željeni cilj algoritam analizira označeni trening set, uči pravila, obrasce i zavisnosti koje povezuju ulazne podatke sa izlaznim vrijednostima. Algoritam naučen na trening setu se zatim primjenjuje na novim, neoznačenim podacima.

Ukoliko ciljna vrijednost predstavlja oznaku klase uzorka radi se o problemu klasifikacije. Primjer klasifikacionog problema je detekcija zloupotrebe platnih kartica koja će se obrađivati u ovom radu. Svaka transakcija je opisana velikim brojem karakteristika koje mogu biti predstavljene numeričkim ili kategorijskim vrijednostima. Kategorijske karakteristike obilježavaju pripadnost određenoj kategoriji ili vrijednost iz diskretnog ograničenog skupa. Jedna od kategorijskih karakteristika je oznaka klase uzorka, koja u slučaju detekcije zloupotrebe platnih kartica ima vrijednost 1 ukoliko se radi o zloupotrebi, ili 0 ukoliko dati uzorak predstavlja regularnu transakciju. Algoritam ima za cilj da predvidi klasu kojoj pripada uzorak, na osnovu obrazaca naučenih iz trening seta.

Osim klasifikacije, veliku upotrebu algoritmi nadgledanog mašinskog učenja pronalaze u predikciji kontinualne numeričke vrijednosti, što spada u domen problema regresije. Primjer regresionog problema je predikcija cijene kuće, gdje algoritam na osnovu označenog trening seta uči zavisnosti stvarnih cijena kuća na tržištu od njihovih karakteristika poput veličine, broja soba, lokacije i ostalog, koje će zatim iskoristiti za estimaciju cijene nove, do tada nepoznate kuće. Glavni predstavnici algoritama nadgledanog mašinskog učenja su KNN, linearna regresija (eng. *Linear Regression*), LR, metoda potpornih vektora (eng. *Support Vector Machine*), DT, RF i neuralne mreže [21].

Za razliku od algoritama nadgledanog, algoritmi nenadgledanog mašinskog učenja uče iz podataka koji nijesu unaprijed označeni. Glavni cilj algoritama iz ove kategorije je otkrivanje skrivenih obrazaca i struktura u podacima. Jedna od brojnih primjena jeste klasterizacija uzoraka na osnovu sličnosti među njima. Ukoliko uzorci predstavljaju osobe, a karakteristike njihova interesovanja, potrebe i želje, algoritmi će klasterizovati osobe u cilju personalizovane preporuke proizvoda. Primjere možemo naći na digitalnim platformama i internet prodavnicama, koje nastoje da predlože sadržaj koji je veoma blizak sadržaju koji se dopada korisniku ili koji su ljudi sličnih interesovanja označili kao kvalitetan.

Osim za klasterizaciju, nenadgledano mašinsko učenje se koristi za vizuelizaciju kompleksnih podataka što može biti veoma značajno u njihovoj analizi i razumijevanju, kao i za smanjenje dimenzionalnosti podataka uz zadržavanje što većeg broja korisnih informacija. Takođe, na osnovu analize strukture podataka i obrazaca u njima, algoritmi nenadgledanog mašinskog učenja su u stanju da detektuju uzorke koji odstupaju od uobičajenih obrazaca, što se primjenjuje u detekciji anomalija.

Polu-nadgledano mašinsko učenje predstavlja kombinaciju prethodne dvije kategorije, gdje je moguće izvršiti predikciju oznake uzorka većinski neoznačenih podataka na osnovu njihove sličnosti sa uzorcima čija je stvarna oznaka klase poznata.

Ovu tehniku primjenjuje servis Google Photos koji klasterizuje slike na osnovu osoba koje se na njima pojavljuju. U ovom procesu koristi nenadgledane algoritme mašinskog učenja, nakon čega je u mogućnosti da na osnovu dobijanja oznake, odnosno imenovanja osobe na jednoj slici, pronađe sve slike na kojima se data osoba pojavljuje.

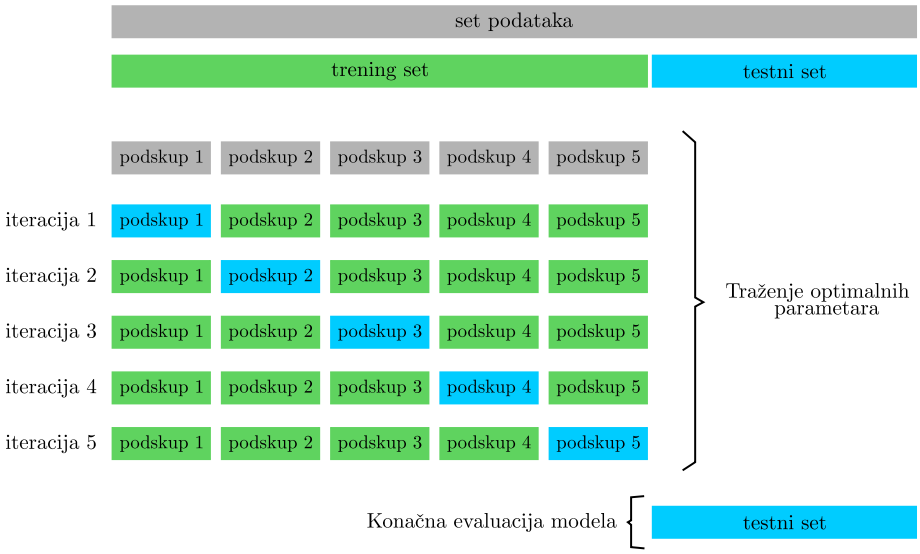
Učenje sa podrškom se razlikuje od prethodno opisanih kategorija mašinskog učenja. Sistem za učenje, koji se u ovom kontekstu zove agent, interaguje sa sredinom i prilikom svojih odluka biva nagrađen ili kažnjen u zavisnosti od toga da li odluka vodi ka zadatom cilju ili ne. Na ovaj način, sistem samostalno uči koja je to strategija, odnosno sekvenca postupaka koji ga vode ka rješenju. Kako je za postupak učenja neophodan veliki broj pokušaja i promašaja, ne čudi što su najveću primjenu algoritmi ove kategorije pronašli u kreiranju agenata koji mogu igrati, čak i pobjeđivati ljude u kompleksnim igrama kao što su Go i šah. Takođe, različiti simulatori vožnje su omogućili treniranje sistema bez stvarnih posljedica u slučaju greške, što je dovelo do optimizacije performansi kontrolnih sistema za upravljanje autonomnim vozilima.

Odabir modela pogodnog za konkretan problem obuhvata odabir algoritma i podešavanje njegovih hiperparametara (broj najbližih susjeda kod KNN-a, dubina stabla kod DT-a i slično). Algoritam se bira u zavisnosti od brojnih faktora poput strukture seta podataka i prirode problema. Svaki od algoritama posjeduje određene hiperparametre koji utiču na proces učenja i strukturu algoritma, kao i kompleksnost njegovog treniranja, stoga je neophodno naći one hiperparametre koji poboljšavaju performanse algoritma.

Odabir vrijednosti hiperparametara se vrši prije treninga algoritma. Naknadnom evaluacijom performansi dobijenog modela biraju se one vrijednosti hiperparametara koje daju najbolje performanse. Ukoliko se učenje algoritma i evaluacija modela vrše na istom setu podataka doći će do njegovog pretjeranog prilagođavanja tome setu podataka. Kao rezultat, model ne bi mogao dobro da klasifikuje nove uzorke, odnosno ne bi imao dobru sposobnost generalizacije. Kako bi se izbjegao ovaj problem, ukupan set podataka se dijeli na tri dijela: trening set, validacioni set i testni set. Trening set služi za treniranje modela mašinskog učenja, stoga je važno da su u njemu prisutni primjeri svih različitih situacija koje se mogu pojaviti. Ako model ne vidi određene obrasce ili slučajeve tokom treniranja, neće moći ni da ih prepozna u budućnosti [21]. Performanse različitih modela, njegovih različitih struktura i hiperparametara se evaluiraju na validacionom setu, pri čemu se u zavisnosti od tipa problema, kao i strukture podataka koriste različite metrike za evaluaciju performansi. Nakon odabira modela optimalne strukture i parametara, vrši se njegovo ponovno treniranje na objedinjenom trening i validacionom setu, a zatim njegova

konačna evaluacija na testnom setu kako bi se procijenila njegova sposobnost generalizacije.

Opisani način kreiranja fiksnog, izdvojenog validacionog seta se naziva validacija sa odvojenim setom (eng. *holdout validation*). Međutim, ukoliko se radi o malom validacionom setu evaluacije neće biti dovoljno pouzdane. Kako bi svako povećanje validacionog seta vodilo smanjivanju trening seta, pristupa se K-slojnoj unakrsnoj validaciji (eng. *K-fold cross-validation*). Kod ove tehnike set ulaznih podataka se dijeli na trening i testni set, nakon čega se trening dio dijeli na K podskupova kao na slici 1. Svaki model se trenira na *K-1* podskupova (označenih zelenom bojom), dok se njegova validacija vrši na preostalom podskupu (plava boja). Konačna greška validacije se dobija kao prosječna greška svih K iteracija. Na ovaj način se dobijaju pouzdanije mjere, pri čemu se smanjuje mogućnost velike varijabilnosti procjena performansi usljed slučajne podjele trening i validacionog seta.



Slika 1: Ilustracija procesa traženja optimalnih hiperparametara modela 5-slojnom unakrsnom validacijom, te njegova evaluacija na testnom setu.

Dva su glavna problema koja se mogu očekivati prilikom razvijanja modela, to su problem velike varijanse i problem velikog *bias*-a. Problem velike varijanse se odnosi na situaciju kada imamo veoma kompleksan model koji je u stanju da se previše prilagodi uzorcima trening seta, čak i onim uzorcima koji predstavljaju šum. Preveliko prilagođavanje uzorcima trening seta vodi do veoma male greške modela na trening setu, ali značajnoj grešci modela na podacima izostavljenim radi njegovog testiranja. Načini rješavanja velike varijanse su pojednostavljivanje modela, smanjivanje broja karakteristika trening seta, povećavanje broja podataka i ograničavanje složenosti modela primjenom regularizacije.

Za razliku od varijanse, problem velikog *bias*-a se događa u situacijama kada se veoma kompleksan problem pokušava riješiti primjenom veoma jednostavnog modela, koji nije u stanju da uhvati kompleksne obrasce u podacima. Osim postizanja loših rezulata na trening setu, model nije u mogućnosti izvršiti dobru generalizaciju, stoga je neophodno povećati dimenzionalnost seta podataka i kompleksnost modela ili smanjiti stepen regularizacije. Na osnovu pomenutih problema jasno je da je cilj čitavog procesa treniranja modela, validacije i testiranja potraga za balansom između modela koji je u stanju da se prilagodi i što više nauči iz trening seta, uz zadržavanje jednostavnosti radi bolje generalizacije.

3 Pretprocesiranje podataka

Podaci se u današnjem vremenu prikupljaju svakodnevno na različite načine. Napredak interneta, kao i same elektronike, doveli su do sveprisutnosti senzora oko nas i omogućili prenos, obradu i čuvanje velikih količina podataka. Sirovi podaci prikupljeni na ovaj način najčešće nijesu u pogodnom obliku, te ne mogu poslužiti algoritmima mašinskog učenja kao dobar materijal za učenje i izvor novog znanja. Kako bi se pomenutim algoritmima obezbijedili najpovoljniji uslovi za postizanje najboljih performansi neophodno je pretprocesirati podatke.

Pretprocesiranje započinje analizom i profilisanjem podataka. Potrebno je obratiti pažnju na njihovu strukturu i kvalitet, kao i njihovu relevantnost za rješavanje datog problema. Nakon izvršenog uvida u podatke i njihovog razumijevanja, javlja se potreba za primjenom određenih tehnika i metoda koje obuhvataju:

- Metode čišćenja podataka;
- Tehnike za kodiranje kategorijskih karakteristika;
- Tehnike skaliranja podataka;
- Tehnike za balansiranje seta podataka;
- Metode za ekstrakciju karakteristika;

Metode čišćenja i tehnike za kodiranje i skaliranje podataka koriste se zbog neadekvatne strukture, tipa i distribucije podataka i njihov glavni cilj je modifikacija podataka. Za razliku od njih, tehnike balansiranja se suočavaju sa problemom neravnomjerne zastupljenosti predstavnika klasa u setu podataka, odnosno njihovom neizbalansiranošću. Neizbalansiranost seta podataka može voditi do pristrasnosti određenih algoritama većinskoj klasi. Stoga se u cilju poboljšanja performansi i pouzdanosti javlja potreba za balansiranjem neizbalansiranog seta. Osim pomenutih tehnika, metode za ekstrakciju karakteristika se koriste u pretprocesiranju podataka u cilju smanjivanja njihove redudantnosti.

3.1 Metode čišćenja podataka

Podaci se sakupljaju u različite svrhe. Osim njihovog sakupljanja u cilju monitoringa, danas se sve više koriste za dodatne analize procesa koji predstavljaju ili čija su posljedica. Iz analize podataka mogu proisteći veoma korisni zaključci i pravilnosti vezani za odgovarajući proces, što dalje može voditi estimaciji vrijednosti željenih veličina u budućnosti, kao i predikciji klase uzorka.

Nepravilno održavanje i čuvanje, nedovoljno kvalitetni senzori, kao i ljudski faktor utiču na pojavu velikog broja neupotrebljivih ili nedostajućih podataka. Nekada, usljed pomenutih razloga, dolazi do pojave podataka čija vrijednost znatno odstupa od vrijednosti ostalih podataka (eng. *outliers*) [22], koji unose šum i negativno utiču na performanse modela ukoliko prije toga ne dođe do adekvatne obrade ovih vrijednosti.

Prvi korak u pretprocesiranju podataka podrazumijeva njihovo čišćenje, odnosno uklanjanje nekvalitetnih podataka i rješavanje problema nedostajućih vrijednosti. U ovu svrhu se koriste:

- Metode rješavanja problema nedostajućih vrijednosti;
- Metode uklanjanja *outlier*-a;

Nedostajuće vrijednosti su veoma česta pojava u setovima podataka. Dok administratori baza podataka mogu zahtijevati unos kompletnih podataka, nekada sama situacija nalaže mogućnost izostavljanja ili nemogućnost sakupljanja istog. Iako za čovjeka ne predstavlja problem, pojava nedostajućih vrijednosti u setu uzrokuje nemogućnost rada određenih algoritama.

U zavisnosti od toga da li se nedostajuće vrijednosti zanemaruju, uklanjaju ili popunjavaju, postoji više pristupa rješavanju ovog problema. Jedan od načina jeste popunjavanje nedostajućih vrijednosti karakteristika. Međutim, vrijednosti kojim će se popuniti nedostajući podaci ne bi trebalo da budu slučajno generisane, već vrijednosti uslovljene od strane distribucije i strukture dostupnih podataka. Podaci predstavljeni u numeričkom obliku, koji mogu imati neku vrijednost iz kontinualnog skupa vrijednosti, se dopunjavaju medijanom ili srednjom vrijednosti dostupnih podataka date karakteristike. Srednja vrijednost, iako statistički najvjerovatnija, često ne predstavlja najbolje rješenje zbog prisutnosti *outlier*-a kao i neizbalansiranosti klasa, zbog čega se najčešće koristi medijan. U sekvencijalnim podacima koji predstavljaju vremensku seriju, neophodno je uzeti u obzir vremensku korelaciju uzoraka. Stoga se, kod sekvencijalnih podataka, za rekonstrukciju nedostajućih vrijednosti najčešće koristi interpolacija ili kopiranje prethodnika ili sljedbenika u vremenu.

Kategorijski podaci za razliku od numeričkih, uzimaju vrijednosti iz ograničenog skupa vrijednosti i reprezentuju pripadnost datog uzorka određenoj klasi ili kategoriji. Podatak koji nedostaje se zamjenjuje najvjerovatnijom vrijednošću, odnosno onom koja se najveći broj puta pojavljivala u setu. U nekim slučajevima dolazi do formiranja dodatne kategorije koja reprezentuje nedostatak vrijednosti.

Veoma je važno naglasiti da svako imputiranje podataka može dovesti do neželjenih efekata pristrasnosti samog modela i pogrešnih predikcija, stoga ovom postupku

treba pristupiti oprezno tek nakon temeljne analize svojstava i karakteristika seta podataka.

Za razliku od popunjavanja nedostajućih vrijednosti, prosto uklanjanje karakteristika sa nedostajućim vrijednostima ili samih nedostajućih uzoraka, ima svrhe upotrebljavati samo u određenim situacijama. Svaka karakteristika opisuje neki aspekt analiziranog problema i može imati značajan udio u procesu donošenja odluka, stoga njihovim uklanjanjem možemo potencijalno izgubiti veoma bitne informacije. Iz navedenog je jasno da se mora pokušati naći kompromis, odnosno izabrati procenat nedostajućih uzoraka određene karakteristike ispod koga se neće vršiti njihovo uklanjanje. Ukoliko preko 60% uzoraka ima nedostatak vrijednosti određene karakteristike, datu karakteristiku bi trebalo ukloniti [2], jer bi popunjavanje odgovarajućim proračunom tako velikog broja podataka, predstavljalo preveliki uticaj inženjera i odabranog metoda proračuna nedostajućih vrijednosti na set podataka, što može voditi promjeni njegove distribucije.

Outlier-i u podacima ne predstavljaju uvijek podatak nastao kao greška mjernih uređaja i propust u njihovom prikupljanju i skladištenju, već mogu reprezentovati prirodnu anomaliju koja može biti glavni indikator određenih procesa važnih za samu klasifikaciju. Veliki uticaj koji ovi podaci imaju na statistiku, kao što je srednja vrijednost, može se negativno odraziti prilikom njihove standardizacije i upotrebe algoritama. Potrebno je pažljivo razmotriti set podataka, njegovu distribuciju i statistiku kako bi se utvrdilo da li su ti uzorci značajni, te da li bi se njihovom modifikacijom i uklanjanjem popravila generalizacija modela.

Vizuelizacijom podataka i njihove distribucije može se utvrditi koji su to uzorci čije vrijednosti odudaraju od ostatka. Drugi pristupi obuhvataju njihovu detekciju primjenom statističkih mjera Z-skora (eng. *Z-score*) i interkvartilnog raspona (eng. *Interquartile Range* - IQR), kao i korišćenjem algoritama nenadgledanog mašinskog učenja [21].

Z-skor je mjera udaljenosti koja opisuje koliko je standardnih devijacija uzorak udaljen od srednje vrijednosti i računa se po formuli:

$$z = \frac{x - \overline{x}}{\sigma}, \tag{1}$$

gdje x predstavlja uzorak, $\overline{x}$ predstavlja srednju vrijednost uzoraka, dok σ predstavlja njihovu standarnu devijaciju. Standardna devijacija opisuje koliko su podaci raspršeni u odnosu na srednju vrijednost. Primjena srednje vrijednosti i standarne devijacije, upućuje na glavni nedostatak Z-skora koji je podložan negativnom uticaju *outlier*-a i podrazumijeva normalnu distribuciju gdje se 99.73% uzoraka nalazi u opsegu vrijednosti $[\overline{x} \pm 3\sigma]$. Uzorci čija vrijednost ne spada u pomenuti opseg,

odnosno čiji se Z-skor ne nalazi u opsegu od $[-3\sigma, 3\sigma]$ se najčešće mogu smatrati *outlier*-ima.

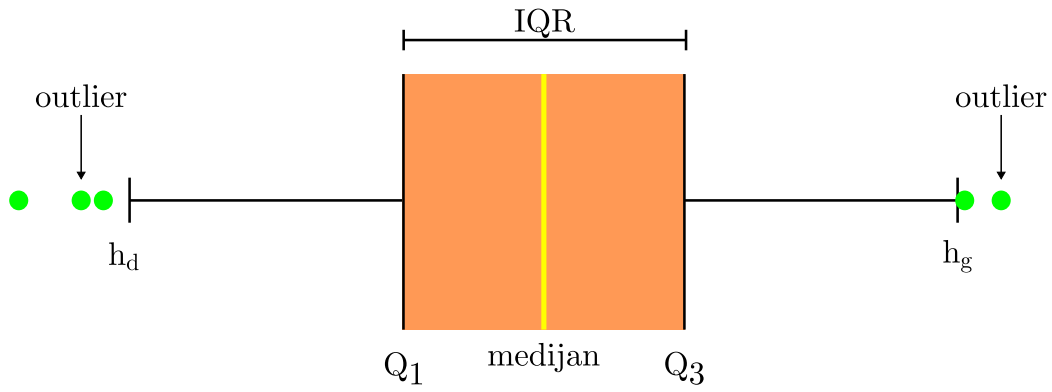
Kako podaci koji nijesu prethodno transformisani najčešće ne podliježu normalnoj distribuciji, nije preporučljivo koristiti srednju vrijednost i standardnu devijaciju. U ovim slučajevima prednost se daje interkvartilnom rasponu koji predstavlja opseg vrijednosti koji obuhvata 50% uzoraka seta podataka i koji je robustniji na prisustvo ekstremnih vrijednosti. U definisanju IQR-a i pragova za detekciju *outlier*-a koriste se kvartili (Q_1, Q_2 (medijan), Q_3) čije vrijednosti dijele set podataka na četiri dijela. Medijan dijeli set podataka na dva seta koji sadrže podjednak broj uzoraka i čiji su medijani Q_1 i Q_3. U svakom dijelu se nalazi po četvrtina ukupnog broja uzoraka.

Donji (h_d) i gornji (h_g) prag, koji se koriste u detekciji *outlier*-a, se računaju po formuli:

$$h_d = Q_1 - C \cdot \text{IQR} \tag{2}$$

$$h_g = Q_3 + C \cdot \text{IQR}, \tag{3}$$

gdje je vrijednost C proizvoljna vrijednost kojom se utiče na širinu opsega, a samim tim i na broj detektovanih *outlier*-a. *Outlier*-ima se smatraju svi oni uzorci koji se ne nalaze u opsegu $[h_d, h_g]$, koji su predstavljeni zelenim kružićima na slici 2.



Slika 2: Ilustracija detektovanja *outlier*-a primjenom IQR-a za računanje donjeg (h_d) i gornjeg (h_g) praga.

Nakon detekcije *outlier*-a postavlja se pitanje šta uraditi sa njima. Ukoliko se radi o malom setu podataka, jednostavnim uklanjanjem uzoraka doći će do gubljenja potencijalno značajnih podataka. U radu [22] nakon detekcije *outlier*-a IQR metodom predlaže se korišćenje *Winsorizing* metode, čime se ove detektovane vrijednosti dovode na vrijednost donjeg i gornjeg praga. Međutim, prije ovakvog uticaja na same podatke neophodno je izvršiti njihovu temeljnu analizu kako bi se spriječio gubitak informacija i pristrasnost modela.

3.2 Tehnike za kodiranje kategorijskih karakteristika

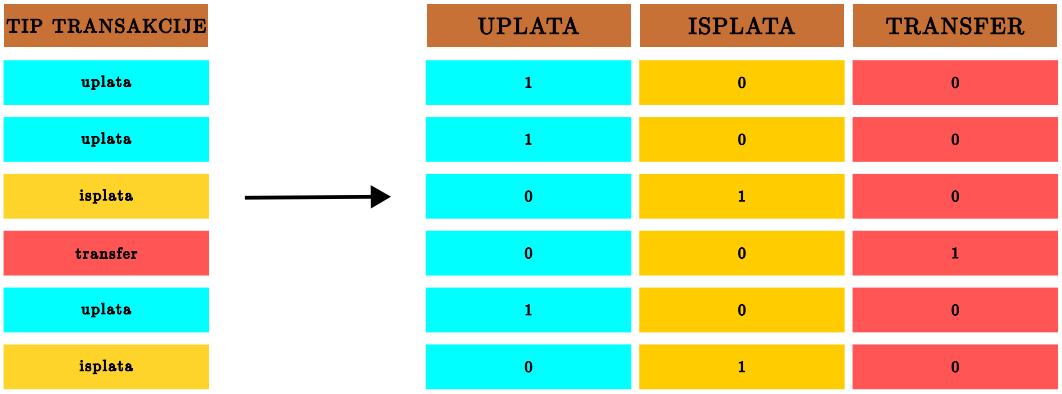
Većina algoritama mašinskog učenja se zasniva na algebarskim operacijama koje zahtijevaju numeričke podatke. Kako bi set podataka prilagodili i omogućili rad ovih algoritama primjenjuju se različite tehnike za kodiranje kategorijskih karakteristika u numeričke vrijednosti.

Koja tehnika kodiranja najbolje odgovara zavisi od vrste kategorijske karakteristike, koja može biti:

- *Nominalna* - karakteristika predstavlja kategorije koje nemaju određenu hijararhiju (redosljed), odnosno kategorije među kojima ne postoji zavisnost. Primjer nominale karakteristike predstavlja tip transakcije čija vrijednost može biti uplata, isplata, kupovina, transfer i ostalo;
- *Ordinalne* - karakteristike koje takođe opisuju pripadnost određenoj kategoriji, sa razlikom postojanja korelacija među njima. Ove korelacije su najčešće uzrokovane postojanjem određene hijararhije među kategorijama. Ovaj tip karakteristike se koristi pri opisivanju stepena aktivnosti korišćenja platne kartice klijenta banke, na osnovu čega razlikujemo nisku, visoku i srednju aktivnost;
- *Binarne* - karakteristike koje mogu imati samo jednu od dvije moguće vrijednosti, odnosno kategorije. U slučaju označavanja regularnosti transakcije to su transakcije koje spadaju u kategoriju regularnih ili u kategoriju zloupotreba;

Kada se radi o nominalnim karakteristikama primjenjuje se tehnika jednoznačnog kodiranja (eng. *one-hot encoding*). Na slici 3 je demonstrirana njena primjena nad karakteristikom koja opisuje tip transakcije, gdje se razlikuju tri kategorije - uplata, isplata i transfer. Za svaku kategoriju tipa transakcije se formira nova karakteristika koja označava pripadnost datog uzorka kategoriji, zbog čega će samo oni uzorci koji predstavljaju uplatu imati jedinicu kao vrijednost karakteristike *uplata*, dok će u ostalim imati nule. Ova tehnika povećava dimenzionalnost seta podataka, stoga je poželjno primjenjivati samo kod nominalnih karakteristika sa malim brojem kategorija. Ovo ograničenje je nametnuto ukoliko je zahtijevana velika brzina treniranja i predikcije modela, kao i u slučajevima kada nema dovoljno memorije na raspolaganju.

Prilikom kodiranja ordinalnih karakteristika neophodno je uzeti u obzir korelaciju među kategorijama, odnosno njihovu hijerarhiju. Korišćenjem kodiranja oznaka (eng. *label encoding*) dodjeljuje se broj svakoj kategoriji. Veća vrijednost označava bolju rangiranost kategorije, čime se čuva početna korelacija među njima. Primjer



Slika 3: Primjer primjene jednoznačnog kodiranja karakteristike koja obilježava tip transakcije. Za svaku od mogućih kategorija formira se zasebna karakteristika koja označava pripadnost uzorka (koja se obilježava sa 1) datoj kategoriji.

upotrebe kodiranja oznaka nad karakteristikom koja opisuje aktivnost korisnika dat je na slici 4. Visoka aktivnost u ovom primjeru opisuje najaktivnijeg korisnika, stoga se njemu dodjeljuje najveća vrijednost 2, dok se ostalim kategorijama, srazmjerno aktivnosti, dodjeljuju niže vrijednosti.

Slika 4: Ilustracija primjene kodiranja oznaka nad ordinalnom karakteristikom koja opisuje aktivnost korisnika, koja može biti: niska, srednja ili visoka.

Binarne karakteristike se sastoje od dvije kategorije i kao takve ne predstavljaju veliki izazov za kodiranje. Kod ovog tipa karakteristike, najčešće se jednoj klasi dodjeljuje vrijednost 1, dok se drugoj dodjeljuje vrijednost 0. Ovim postupkom se ne vrši proširivanje dimenzionalnosti seta podataka.

3.3 Tehnike skaliranja podataka

Karakteristike su zavisno od toga što označavaju, predstavljene različitim mjernim oznakama i imaju vrijednosti iz različitog opsega. Ovo se najbolje može primijetiti u primjeru procjene cijene kuće na osnovu karakteristika različitih mjernih skala. Na primjer, broj soba je najčešće jednocifren broj koji varira od 0 do 4, dok površina kuće, izražena u kvadratnim metrima, najčešće ima mnogo veće vrijednosti. Kod algoritama koji kao mjeru sličnosti koriste udaljenost, poput K-najbližih susjeda, one karakteristike koje imaju znatno veći opseg vrijednosti će biti značajnije prilikom poređenja dva uzorka. Skaliranjem se karakteristike dovode u sličan opseg vrijednosti, čime se obezbjeđuje njihov ravnomjerni značaj u računanju udaljenosti i omogućava njihovo poređenje. Dodatno, dovođenjem karakteristika u sličan opseg vrijednosti ubrzava se konvergencija algoritama koji u svojoj optimizaciji koriste gradijentni spust i njegove modifikacije, kakvi su linearna regresija, logistička regresija i neuralne mreže. Takođe, na ovaj način se ekstremne vrijednosti dovode u unaprijed definisani opseg, što je veoma značajno za stabilnost prilikom vršenja računskih operacija. Za razliku od prethodno pomenutih algoritama, neskalirani podaci ne predstavljaju izazov za stabla odlučivanja koja na nivou čvora vrše grananje seta podataka, na osnovu vrijednosti samo jedne karakteristike.

U zavisnosti od željene distribucije seta podataka, kao i njihove strukture razlikujemo nekoliko tehnika skaliranja koje će biti demonstrirane na setu podataka $D = \{(\mathbf{x_i}, y_i) | \, |D| = n, \, \mathbf{x_i} \in \mathbb{R}^m, \, y_i \in \mathbb{R}\}$, gdje n predstavlja broj uzoraka koji su opisani sa m karakteristika. Najčešće korišćene tehnike skaliranja su:

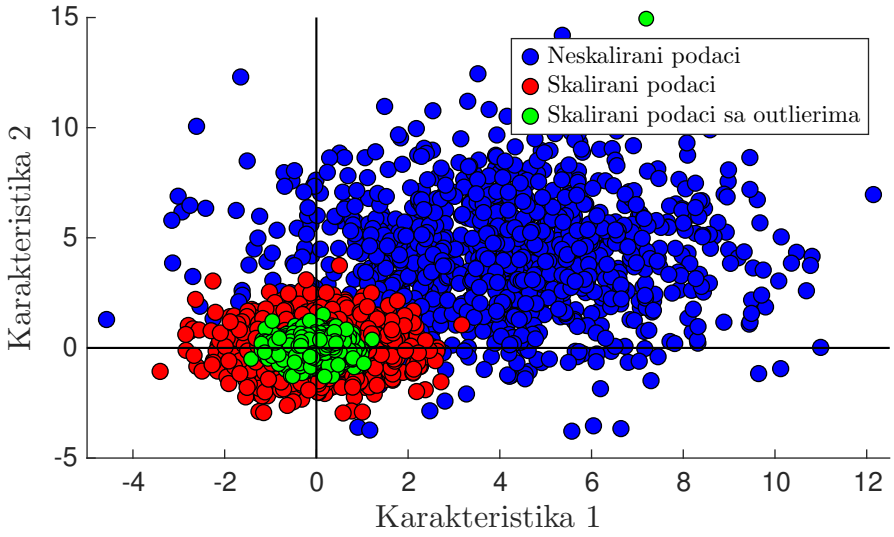
- Standardizacija;
- Robustna standardizacija;
- Skaliranje na bazi minimuma i maksimuma (eng. *min-max scaling*);

Standardizacija vrši transformisanje vrijednosti svih karakteristika u standardizovan oblik sa srednjom vrijednošću 0 i standarnom devijacijom 1, prilikom čega se čuvaju uzajamni odnosi koji su postojali među uzorcima prije skaliranja. Skaliranje se vrši po formuli:

$$\mathbf{x}^{(j)'} = \frac{\mathbf{x}^{(j)} - x^{(j)}_{\text{mean}}}{\sigma^{(j)}}, \quad j = 1, 2, ..., m, \tag{4}$$

gdje $\mathbf{x}^{(j)}$ predstavlja vektor jedne od m mogućih karakteristika uzoraka, dok su $x^{(j)}_{\text{mean}}$ i $\sigma^{(j)}$ njena srednja vrijednost i standardna devijacija respektivno.

Na slici 5 je prikazano standardizovanje originalnog seta podataka opisanog sa dvije karakteristike (predstavljenog plavom bojom), sa i bez prisustva *outlier*-a. Crvenom bojom su prikazani podaci bez *outlier*-a nakon standardizacije. Naknadno su setu podataka dodati uzorci sa ekstremnim vrijednostima karakteristika (veće od 50) kako bi se demonstrirao uticaj ovih uzoraka na proces standardizacije cijelog seta. Sa slike se može primijetiti kako prisustvo *outlier*-a znatno utiče na skaliranje ostalih uzoraka seta (predstavljenih zelenom bojom) tokom standardizacije, sabijajući ih prema centru. Ovo predstavlja glavni nedostatak ove tehnike, čija se upotreba preporučuje ukoliko podaci prate Gausovu raspodjelu.



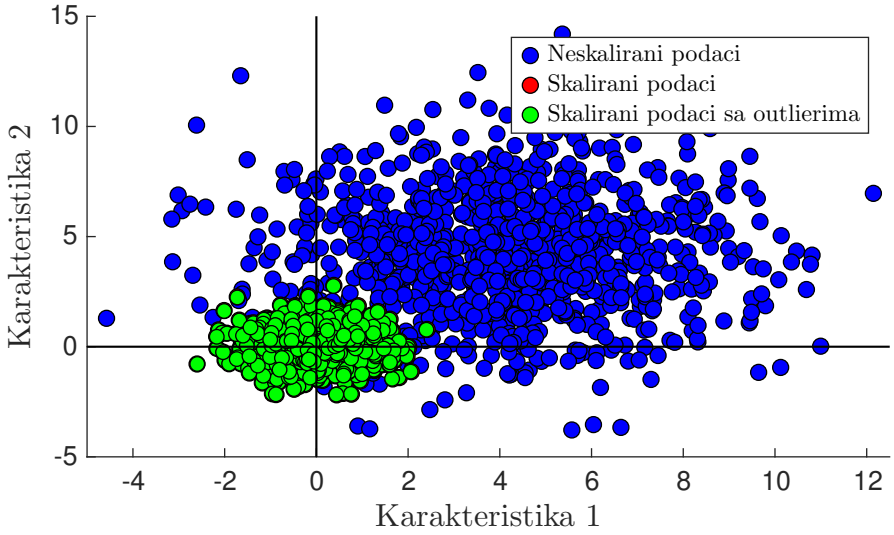
Slika 5: Ilustracija standardizacije na primjeru podatka sa normalnom raspodjelom, sa i bez prisustva *outlier*-a.

Glavni nedostatak standardizacije je korišćenje srednje vrijednosti za skaliranje podataka, koja zbog velikog uticaja *outlier*-a na nivou karakteristike ne predstavlja pouzdanu prosječnu vrijednost. Kako bi prevazišla ovaj nedostatak, robustna standardizacija koristi medijan i IQR, što proces skaliranja čini otpornijim na negativan uticaj ekstremnih vrijednosti. Računa se po formuli:

$$\mathbf{x}^{(j)'} = \frac{\mathbf{x}^{(j)} - x^{(j)}_{median}}{\mathrm{IQR}^{(j)}}, \quad j = 1, 2, ..., m, \tag{5}$$

gdje $\mathrm{IQR}^{(j)}$ predstavlja raspon vrijednosti j-te karakteristike u kome se nalaze 50% uzoraka trening seta. Na slici 6 se može vidjeti da dodavanje *outlier*-a ne utiče na skaliranje seta robustnom standardizacijom, gdje se ogleda robustnost ove tehnike na prisustvo ekstremnih vrijednosti.

Skaliranje na bazi minimuma i maksimuma vrši pomjeranje distribucije karakteristika u opseg od 0 do 1, kao što se može vidjeti na slici 7. Vrijednosti skaliranih

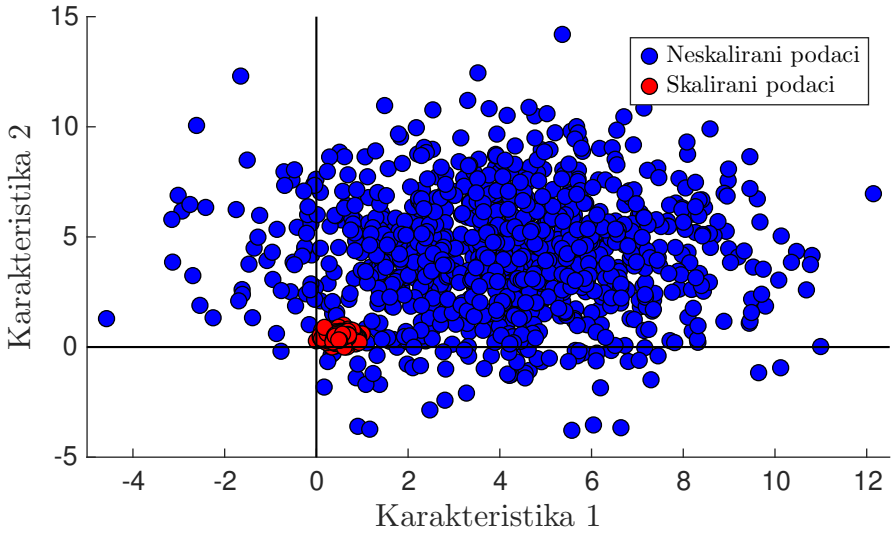


Slika 6: Ilustracija robustne standardizacije na primjeru podatka sa normalnom raspodjelom, sa i bez prisustva *outlier*-a.

uzoraka se računaju po formuli:

$$\mathbf{x}^{(j)'} = \frac{\mathbf{x}^{(j)} - x^{(j)}_{\min}}{x^{(j)}_{\max} - x^{(j)}_{\min}}, \quad j = 1, 2, ..., m, \tag{6}$$

gdje $x^{(j)}_{\min}$ i $x^{(j)}_{\max}$ predstavljaju najmanju i najveću vrijednost j-te karakteristike. Međutim, korišćenje ovih ekstremnih vrijednosti može voditi veoma lošem skaliranju ukoliko one predstavljaju *outlier*-e.



Slika 7: Ilustracija skaliranja originalnog seta podataka, predstavljenog plavom bojom, na bazi minimuma i maksimuma (6). Nakon skaliranja svi podaci se prenose u opseg od [0,1].

3.4 Tehnike za balansiranje seta podataka

Set podataka koji dominantno sadrži uzorke jedne klase u odnosu na drugu, se naziva neizbalansirani set. Rad sa neizbalansiranim setom predstavlja jedan od najčešćih izazova sa kojima se susreću inženjeri mašinskog učenja [11]. U klasičnim algoritmima klasifikacije kao što su logistička regresija, K-najbližih susjeda i stabla odlučivanja, ključna pretpostavka je da se radi o setovima sa sličnim brojem uzoraka obje klase [11]. Ukoliko to nije slučaj, kao u primjeru zloupotrebe platnih kartica, gdje broj regularnih transakcija uveliko premašuje broj njihovih zloupotreba, klasifikator postaje pristrasan klasi koja dominira. Ova pristrasnost, usljed velike razlike u zastupljenosti, vodi do zanemarivanja odbiraka manjinske klase. Jedan od glavnih uzroka nesnalaženja ovih algoritama, pri radu sa neizbalansiranim setom, je njihova težnja ka povećavanjem tačnosti predikcije, koja nije vjerodostojna metrika, o čemu će biti više riječi u Sekciji 5.

Primjenom ansambl metoda, kao i dodjeljivanjem većih težina uzorcima manjinske klase radi njihove bolje klasifikacije pokušava se prevazići ovaj problem na nivou algoritama [11]. Drugi pristup se oslanja na modifikaciju i promjenu odnosa broja uzoraka klasa, tokom faze pretprocesiranja podataka, sa ciljem postizanja željenog odnosa:

$$\alpha = \frac{n_m}{n_v}, \tag{7}$$

gdje n_m predstavlja broj uzoraka manjinske klase, a n_v broj pripadnika većinske klase nakon primjene tehnika balansiranja.

Razlikuju se tri kategorije tehnika balansiranja na nivou podataka:

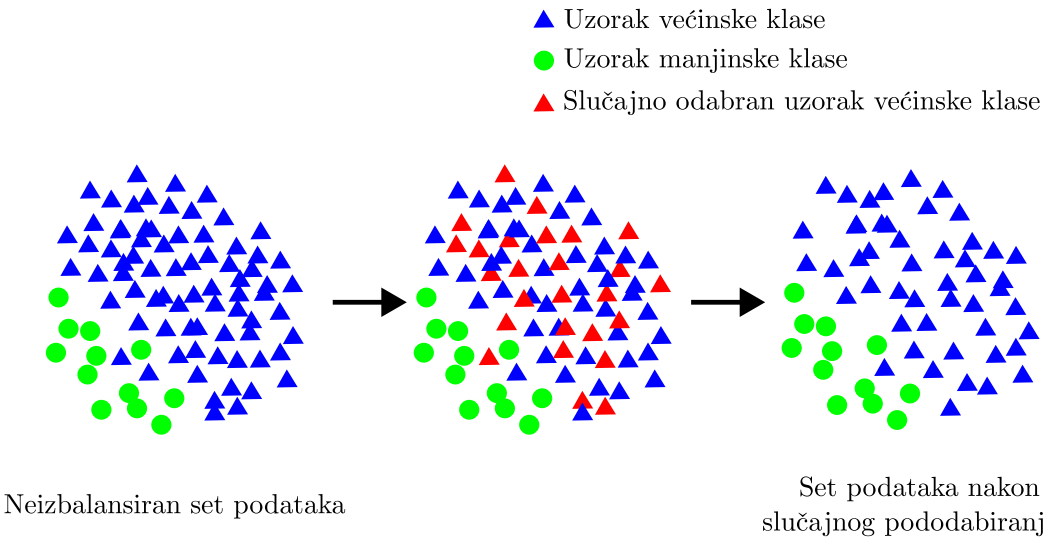
- Tehnike pododabiranja (eng. *Undersampling techniques*);
- Tehnike preodabiranja (eng. *Oversampling techniques*);
- Hibridne tehnike;

3.4.1 Tehnike pododabiranja

Tehnike pododabiranja nastoje smanjiti broj uzoraka većinske klase njihovim uklanjanjem. Na osnovu različitih pristupa balansiranju klasa izdvajaju se:

- Slučajno pododabiranje (eng. *Random UnderSampling* - RUS);
- Tomekove veze (eng. *Tomek links* - Tomek);
- Modifikovani najbliži susjedi (eng. *Edited Nearest Neighbour* - ENN);

RUS slučajno bira i uklanja uzorke većinske klase seta podataka, što je prikazano na slici 8. Radi se o veoma brzoj tehnici, koja odbacuje veliki dio originalnog seta podataka, što vodi do bržeg procesa treniranja modela. Međutim, upravo ovo slučajno odbacivanje velikog dijela potencijalno korisnih informacija predstavlja glavnu manu ove metode, koja može voditi lošijoj generalizaciji modela.



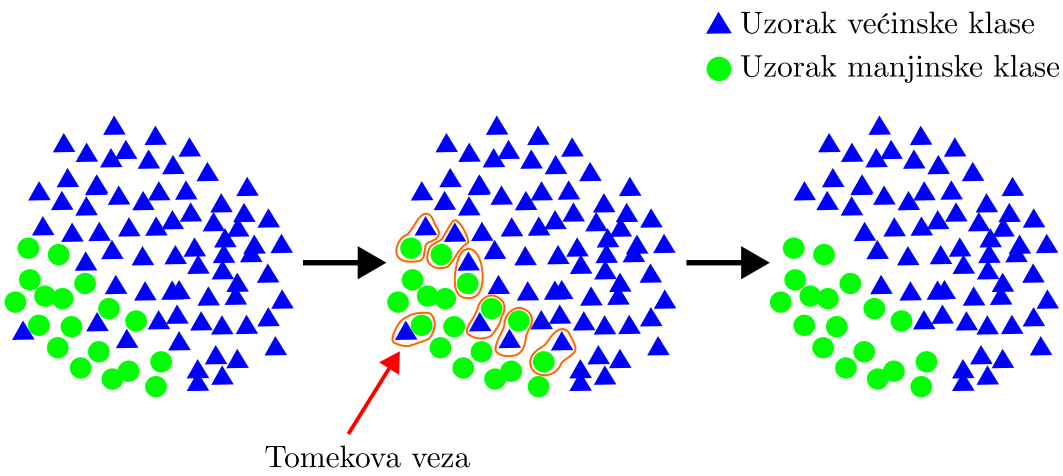
Slika 8: Primjena tehnike slučajnog pododabiranja na neizbalansiranom setu podataka.

Tomekove veze su prvi put predstavljene 1976. godine od strane Ivana Tomeka [23]. Tomekovom vezom, predstavljenom na slici 9, se označava par uzoraka koji pripadaju različitim klasama i koji su međusobno najbliži. Za ovakve parove se kaže da pripadaju graničnom pojasu dvije klase.

Uklanjanjem pripadnika većinske klase u Tomekovoj vezi vrši se jasnije diferenciranje prostora dvije klase, što može doprinijeti boljoj klasifikaciji. Iako, veoma koristan metod, samostalna primjena ne može riješiti problem velike neizbalansiranosti, stoga se ova tehnika najčešće upotrebljava u paru sa nekom od tehnika preodabiranja [11].

Uzorak, koji za svog najbližeg susjeda ima pripadnika suprotne klase, najčešće se tretira kao *outlier*. Stoga ova tehnika može da posluži za njihovu detekciju, nakon čega bi došlo do uklanjanja čitavog para ili odabranog uzorka [11].

Primjenom metoda za grupisanje uzoraka u klastere, a zatim uklanjanje onih uzoraka većinske klase koji među svojim susjedima imaju makar jedan ili većinu uzoraka suprotne klase, postiže se isti efekat kao i kod Tomekovih veza. Posmatrani broj susjeda i potreban broj pripadnika suprotne klase za uklanjanje prosljeđuju se kao argumenti. Ova tehnika modifikovanog najbližeg susjeda najčešće ne vodi



Slika 9: Uspostavljanje Tomekove veze i uklanjanje uzorka većinske klase.

velikoj redukciji broja uzoraka, radi čega je Ivan Tomek u radu [24] predložio dvije modifikacije:

1. Ponovljeni modifikovani najbliži susjed (eng. *Repeated Edited Nearest Neighbour - RENN*) - podrazumijeva iterativno ponavljanje tehnike;
2. AllKNN - podrazumijeva iterativno ponavljanje tehnike, pri čemu se posmatrani broj susjeda (početno 1) inkrementira u svakoj iteraciji;

Iterativno ponavljanje se vrši do unaprijed zadatog odnosa broja uzoraka dvije klase α (7) ili unaprijed zadatog broja iteracija.

3.4.2 Tehnike preodabiranja

Za razliku od tehnika pododabiranja koja nastoje uticati na brojnost većinske klase, tehnike preodabiranja nastoje balansirati set podataka generisanjem novih uzoraka manjinske klase. Neke od najpoznatijih tehnika obuhvataju:

- Slučajno preodabiranje (eng. *Random OverSampling* - ROS)
- SMOTE
- Adaptivno sintetičko odabiranje (eng. *ADAptive SYNthetic sampling approach - ADASYN*)

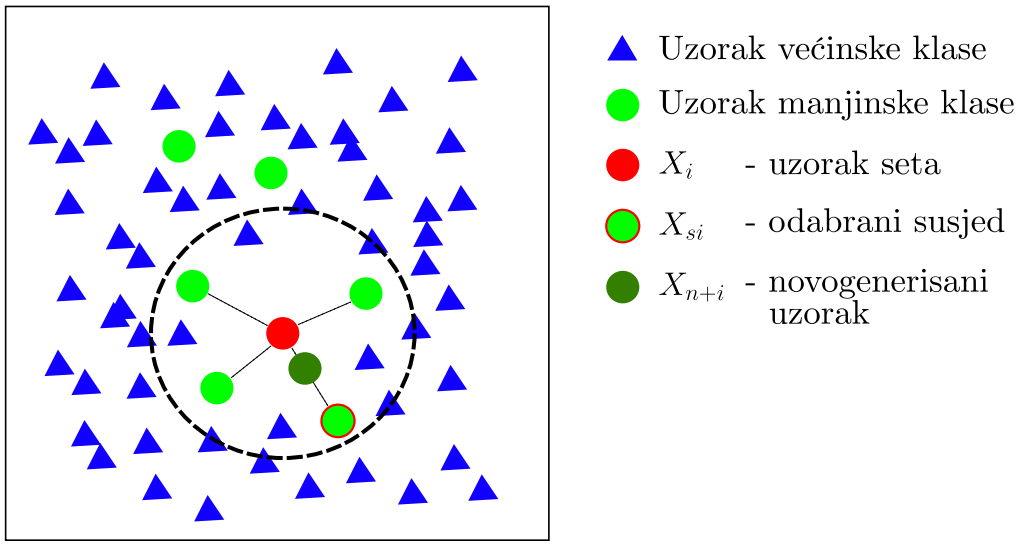
ROS vrši balansiranje seta podataka duplikacijom uzoraka manjinske klase, izabranim primjenom slučajnog odabira sa ponavljanjem. Kod odabiranja sa ponavljanjem u svakoj iteraciji bira se uzorak iz čitavog seta podataka, tako da jedan

uzorak na kraju može biti dupliran više puta. Ova naivna tehnika, ne koristi heuristiku ili statističke informacije seta podataka, što je čini veoma brzom i pogodnom za primjene nad velikim setom podataka. Multiplikovanjem uzoraka manjinske klase povećavaju se šanse da će algoritmi, koji za evaluaciju svojih performansi i optimizovanje parametara koriste tačnost, dobro klasifikovati pomenute uzorke. Iako veoma brza tehnika, njom se ne unose nikakve nove informacije u set podataka, dok veliko preodabiranje može voditi ka preprilagođavanju trening podacima i usporavanju procesa treniranja [25].

SMOTE je tehnika koja vrši generisanje novih uzoraka interpolacijom. Za svaki uzorak $\mathbf{x}_i$ trening seta $D = \{(\mathbf{x_i}, y_i) | \, |D| = n, \, \mathbf{x_i} \in \mathbb{R}^m, \, y_i \in \mathbb{R}\}$ koji pripada manjinskoj klasi ($y_i = 1$), nalazi se k najbližih susjeda ($\mathbf{x}_{s0}$,$\mathbf{x}_{s1}$,... $\mathbf{x}_{sk-1}$), na osnovu Euklidove distance [26]. Novi uzorak $\mathbf{x}_{n+i}$ se generiše negdje duž linije između uzoraka $\mathbf{x_i}$ i slučajno odabranog susjeda $\mathbf{x_s}$ po formuli:

$$\mathbf{x}_{n+i} = \mathbf{x}_i + \lambda(\mathbf{x}_s - \mathbf{x}_i), \quad i = 1, 2, ..., n_m, \tag{8}$$

gdje λ predstavlja slučajno odabranu vrijednost od 0 do 1 kojom se definiše pozicija novogenerisanog uzorka, dok n_m predstavlja broj uzoraka manjinske klase. Broj generisanih uzoraka za svakog predstavnika manjinske klase zavisi od željenog odnosa α definisanog formulom (7).



Slika 10: Generisanje novog uzorka korišćenjem SMOTE tehnike.

Uzorci koji se nalaze u graničnom pojasu dvije klase se mnogo teže klasifikuju u odnosu na one koji se nalaze daleko od pripadnika sebi suprotne klase. Granično sintetičko predodabiranje manjinske klase (eng. *Borderline-SMOTE* - BSMOTE) predstavlja modifikaciju SMOTE-a, gdje se preodabiraju samo oni odbirci čija većina

susjeda pripada suprotnoj klasi. Uzorci koji u svojoj okolini nemaju pripadnika sopstvene klase se proglašavaju šumom i oni ne učestvuju u procesu generisanja novih podataka [11].

ADASYN takođe predstavlja novu verziju SMOTE-a, gdje broj multiplikacija uzorka zavisi od klasa uzoraka iz njegove okoline. Za svaki uzorak trening seta koji pripada manjinskoj klasi, traži se k najbližih susjeda i računa odnos susjeda r_i zavisno od njihove oznake klase prema formuli:

$$r_i = \frac{\omega_i}{k}, \quad i = 1, 2, ..., n_m, \tag{9}$$

gdje ω_i predstavlja broj susjeda uzorka $\mathbf{x}_i$ koji pripadaju većinskoj klasi, dok k predstavlja posmatrani broj najbližih susjeda.

Normalizacijom vrijednosti r_i po formuli:

$$\hat{r}_i = \frac{r_i}{\sum_{i=1}^{n_m} r_i}, \quad i = 1, 2, ..., n_m, \tag{10}$$

gdje n_m predstavlja broj uzoraka manjinske klase, dobijamo gustinu distribucije za svaki uzorak manjinske klase. Broj novih uzoraka g_i za $\mathbf{x}_i$ je proporcionalan dobijenom normalizovanom odnosu susjeda $\hat{r}_i$ i računa se po formuli:

$$g_i = \hat{r}_i \cdot G, \quad i = 1, 2, ..., n_m, \tag{11}$$

gdje G predstavlja potreban broj novogenerisanih uzoraka manjinske klase kako bi se postigao željeni odnos broja uzoraka dvije klase α (7). Novi uzorci se dalje generišu po formuli (8) kao i kod SMOTE tehnike.

Na ovaj način osim što dolazi do balansiranja seta podataka, vrši se fokusiranje algoritma na one uzorke koji su teški za učenje, odnosno čija pripadnost klasi nije neupitna [27].

3.4.3 Hibridne tehnike

Kombinujući metode preodabiranja i pododabiranja moguće je iskoristiti prednosti ovih nezavisnih tehnika, postižući bolje rezultate u odnosu na njihovu nezavisnu primjenu. U radu [11] se ističe upotreba SMOTE-a za preodabiranje manjinske klase, a zatim uklanjanje uzoraka većinske klase korišćenjem ENN-a i Tomekovih veza. U odnosu na primjenu isključivo tehnike za preodabiranje u postizanju željenog odnosa, glavna prednost hibridnih tehnika je u brzini i jednostavnosti.

3.5 Metode izdvajanja karakteristika

Mašinsko učenje svoju glavnu prednost u odnosu na čovjeka pokazuje u problemima gdje je svaki uzorak opisan sa velikim brojem karakteristika. Međutim, veliki broj karakteristika koje opisuju posmatrani proces ne predstavlja uvijek prednost. Povećavanje broja karakteristika vodi ka povećavanju dimenzionalnosti prostora, te određeni uzorci, iako blizu u niskodimenzionom prostoru, bivaju raspršeni. Osim mnogo veće memorijske i vremenske kompleksnosti prilikom treninga modela nad ovakvim podacima, traženje prave metrike koja bi oslikavala sličnost među njima predstavlja izazov. Ovaj problem se često referencira kao prokletstvo dimenzionalnosti (eng. *curse of dimensionality*).

Bez obzira na vremensku i memorijsku složenost, postojanje međusobne zavisnosti pojedinih karakteristika, dovodi do redudantnosti podataka. Metode ekstrakcije imaju za cilj generisanje manjeg broja novih karakteristika na osnovu postojećih, pri čemu se teži očuvati što veća količina informacija i ukloniti redudantnost karakteristika. Jedna od najpopularnijih metoda je analiza glavnih komponenti (eng. *Principal Component Analysis* - PCA).

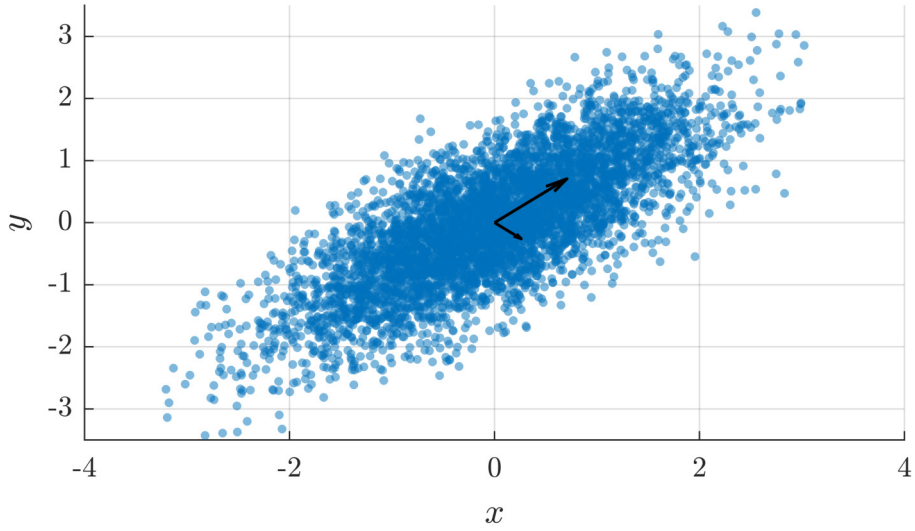
Analiza glavnih komponenti, definisana 1901. godine od strane Karl Pearsona [28], je metoda koje utvrđuje zavisnost među karakteristikama i vrši uklanjanje redudantnih podataka primjenjujući metode linearne algebre. Korišćenjem ortogonalnih transformacija vrši se preslikavanje prostora seta podataka koji čine n uzoraka, predstavljenih sa m karakteristika, $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(n)}\}$, $\mathbf{x}^{(i)} \in \mathbb{R}^m$ u k-dimenzioni prostor, pri čemu važi da je $(k \ll n, k < m)$. Ose novog prostora predstavljaju vektore kojima su definisani pravci glavnih komponenti. Projekcijom podataka na ove ose dobijaju se glavne komponente, odnosno nove karakteristike kojima je opisan novi set podataka.

Prije primjene metode glavnih komponenti neophodno je standardizovati podatke (Sekcija 3.3), nakon čega se traže one glavne komponente koje zadržavaju najveću informativnost podataka, kao što je prikazano na slici 11. Varijansa, koja oslikava razlike u podacima koje nose najveću količinu informacije i imaju najveći značaj pri njihovom upoređivanju, predstavlja mjeru njihove informativnosti. Jedan od načina dolaženja do glavnih komponenti, koji ne zahtijeva eksplicitno računanje matrice kovarijanse je primjena faktorizacione tehnike dekompozicije singularnih vrijednosti (eng. *Singular Value Decomposition* - SVD). Ova tehnika polazi od mogućnosti reprezentacije originalnog seta podataka $\mathbf{X}$ pomoću proizvoda tri matrice:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \tag{12}$$

gdje je matrica $\mathbf{U}$ ortogonalna matrica dimenzija $n \times n$ koja sadrži lijeve singularne

vektore, matrica $\mathbf{V}$ ortogonalna matrica dimenzija $m \times m$ koja sadrži desne singularne vektore koji definišu pravce glavnih komponenti, dok matrica $\mathbf{\Sigma}$ dimenzija $n \times m$ predstavlja dijagonalnu matricu singularnih vrijednosti.



Slika 11: Na slici su prikazani standardizovani odbirci dvodimenzione slučajne promjenljive koja podliježe Gausovoj raspodjeli. Prikazani vektori predstavljaju pravce glavnih komponenti, skalirani proporcionalno količini varijanse podataka sadržane u njima.

$$\mathbf{V} = \begin{bmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_m \\ \downarrow & \downarrow & & \downarrow \end{bmatrix}_{m \times m} \tag{13}$$

Singularne vrijednosti σ koje se nalaze po dijagonali matrice $\mathbf{\Sigma}$ jednake su kvadratnom korijenu sopstvenih vrijednosti λ skalirane matrice kovarijanse $\mathbf{X}^T\mathbf{X}$. Sortirani u rastućem poretku tako da važi $\sigma_1 \geq \sigma_2 \cdots \geq \sigma_m$, oslikavaju koliko je varijanse sadržano u projekcijama na pravce definisane sopstvenim vektorima $\mathbf{v}_1, \mathbf{v}_2 \ldots \mathbf{v}_m$ (13). Projekcijom početnog skaliranog seta podataka $\mathbf{X}$ na prvih k sopstvenih vektora ($\mathbf{V}_k$) dobijaju se glavne komponente koje predstavljaju nove karakteristike seta podatka $\mathbf{Z}$ manje dimenzionalnosti:

$$\mathbf{Z} = \mathbf{X}\mathbf{V}_k = \begin{bmatrix} - & \mathbf{x}_1 & \rightarrow \\ & \vdots & \\ - & \mathbf{x}_n & \rightarrow \end{bmatrix}_{n \times m} \begin{bmatrix} | & & | \\ \mathbf{v}_1 & \cdots & \mathbf{v}_k \\ \downarrow & & \downarrow \end{bmatrix}_{m \times k} . \tag{14}$$

Cilj PCA jeste identifikacija glavnih komponenti koji će zadržati najveći dio varijanse u podacima preslikanim u novi k-dimenzioni prostor. Broj glavnih komponenti

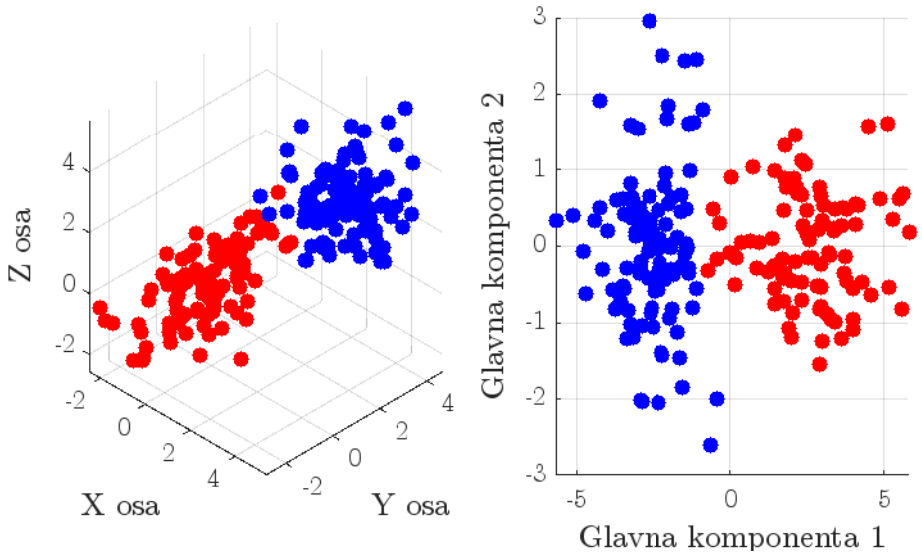
koji će se koristiti u (14) zavisi od zahtjeva i prirode problema. Pragom α se definiše količina zadržanih informacija čime se podešava stepen redukcije. Bira se minimalan broj glavnih komponenti k koji zadovoljava:

$$\sum_{i=1}^{k} \frac{\sigma_i^2}{\sum_{j=1}^{m} \sigma_j^2} > \alpha, \tag{15}$$

pri čemu m predstavlja broj karakteristika kojim su opisani uzorci originalnog seta podataka.

Uklonjeni dio varijanse podataka, koji ne nosi previše informacija, biva nepovratno izgubljen i naziva se greškom rekonstrukcije (eng. *reconstruction error*). Rekonstruisani originalni podaci mogu se jednostavno dobiti reverzibilnim procesom datim formulom:

$$\hat{\mathbf{X}} = \mathbf{Z}\mathbf{V}_k^T \tag{16}$$



Slika 12: Primjer projekcije podataka predstavljenih u 3D prostoru (slika lijevo), sa jasno označenim klasama, na 2D ravan (slika desno) korišćenjem PCA.

Ortogonalnost glavnih komponenti je obezbijedila nekorelisanost karakteristika novog seta podataka **Z**. Osim uklanjanja redudantnih podataka, smanjila se njihova dimenzionalnost što vodi ka njihovom bržem procesiranju i obradi. Dodatno, mogućnost reprezentacije višedimenzionih podataka u dvodimenzionom (2D) ili trodimenzionom (3D) prostoru omogućava njihovu vizuelizaciju, što doprinosi njihovoj analizi i interpretaciji. Na slici 12 je prikazana transformacija podataka koji sadrže uzorke dvije klase iz 3D u 2D prostor. Reprezentacijom podataka u 2D prostoru ne

dolazi do gubitaka velike količine informacija, jer su glavne komponente pažljivo odabrane da sadrže maksimalnu varijansu podataka. Na ovaj način granica odlučivanja između dvije klase postaje jasnija i lakše je uočiti strukturu podataka, što je veoma značajno pri njihovoj klasifikaciji.

Nakon primjene PCA, dobijeni podaci $\mathbf{Z}$ nemaju fizički smisao te se ni na koji način ne mogu povezati sa originalnim karakteristikama po značenju. Upravo ovo svojstvo ove metode nalazi primjenu u zaštiti podataka, što je veoma značajno pri radu sa osjetljivim i privatnim podacima kakve su transakcije.

4 Algoritmi klasifikacije mašinskog učenja

Zavisnost pripadnosti određenoj klasi od karakteristika uzoraka algoritmi klasifikacije mašinskog učenja uče u toku procesa treniranja nad trening setom $D = \{(\mathbf{x_i}, y_i) | \, |D| = n, \, \mathbf{x_i} \in \mathbb{R}^m, \, y_i \in \mathbb{R}\}$.

Problem detekcije zloupotreba platnih kartica predstavlja problem binarne klasifikacije, stoga će se ovdje razmatrati algorimi:

- logistička regresija;
- K-najbližih susjeda;
- stabla odlučivanja;
- ansambl metode;

Osim što će se razmatrati prednosti i mane ovih algoritama kod binarne klasifikacije, važno je istaći da se ovi algoritmi, uz blage modifikacije mogu primijeniti i u problemima višeklasne klasifikacije.

Iako u radu [29] nadmašuje performanse KNN-a (*K-Nearest Neighbours*) i NB-a (*Naive Bayes*), LR (*Logistic Regression*) se u radovima [9, 13, 30] pokazuje kao loš izbor za konkretni problem, stoga će biti korišćen kao referentna vrijednost sa kojom će se upoređivati performanse drugih algoritama. Za razliku od LR, KNN pokazuje veoma dobre rezultate u detekciji zloupotreba platnih kartica u radovima [7, 13, 31].

Da se ansambl metodi poput RF-a, GBDT-a, CatBoost-a i XGBoost-a nameću kao najbolja opcija pokazuje se u radovima [3, 4, 12, 14]. U brojnim radovim [9, 13, 14, 30–33] RF je postigao bolje rezultate u odnosu na ostale algoritme nadgledanog učenja, čime se dokazuje njegova mogućnost rada sa neizbalansiranim podacima i dobra mogućnost generalizacije [32]. LightGBM u radovima [3–5], SVM u radovima [30, 32, 33], kao i neuralna mreža u radovima [9, 12] su pokazali slabije performanse od gore pomenutih algoritama, stoga neće biti razmatrani.

4.1 Logistička regresija

Logistička regresija jedan je od osnovnih algoritama nadgledanog mašinskog učenja. Prilikom binarne klasifikacije uzorka $\mathbf{x}_i$ vjerovatnoća njegove pripadnosti pozitivnoj klasi (klasi sa oznakom 1) se može izraziti formulom:

$$h_{\mathbf{w},b}(\mathbf{x}_i) = g(\mathbf{w}\mathbf{x}_i + b) \quad i = 1, 2, \ldots, n, \tag{17}$$

gdje n predstavlja broj uzoraka seta podataka, $\mathbf{x}_i$ predstavlja i-ti uzorak predstavljen sa m karakteristika, $\mathbf{w}$ predstavlja odgovarajući m-dimenzioni vektor težinskih koeficijenata, koji reprezentuje zavisnost karakteristika i-tog uzorka i poznate oznake klase datog uzorka y_i, b je slobodni član koji predstavlja odstupanje (eng. *bias*), g predstavlja sigmoid funkciju:

$$g(z) = \frac{1}{1 + e^{-z}}. \tag{18}$$

Sigmoid (logistička) funkcija prevodi linearnu kombinaciju ulaznih karakteristika u opseg od 0 do 1, što odgovara vjerovatnoći da uzorak pripada pozitivnoj klasi.

Nakon što se pronađu optimalne vrijednosti težinskih koeficijenata modela, na osnovu ulaznih karakteristika novog uzorka $\mathbf{x}'$ vrši se predikcija klase $\hat{y}'$ po formuli:

$$\hat{y}' = \begin{cases} 0, & h_{\mathbf{w},b}(\mathbf{x}^{'}) < 0.5 \\ 1, & h_{\mathbf{w},b}(\mathbf{x}^{'}) \geq 0.5. \end{cases} \tag{19}$$

Optimalne vrijednosti koeficijenata $\mathbf{w}$ i b kod binarne klasifikacije dobijaju se minimizovanjem negativne logaritamske funkcije gubitaka (eng. *log(loss) function*):

$$\begin{aligned} J(\mathbf{w}, b) = &- \frac{1}{n} \sum_{i=1}^{n} \left[y_i \log(h_{\mathbf{w},b}(\mathbf{x}_i)) + (1 - y_i) \log(1 - h_{\mathbf{w},b}(\mathbf{x}_i)) \right] \\ &+ \frac{\lambda}{2n} \sum_{j=1}^{m} w_j^2, \end{aligned} \tag{20}$$

gdje drugi član predstavlja L2 regularizaciju sa hiperparametrom λ. Regularizacija ima za cilj da smanji preprilagođavanje modela trening podacima nastojeći poboljšati njegovu generalizaciju, održavajući niske vrijednosti težinskih koeficijenata.

Optimizacija težinskih koeficijenata $\mathbf{w}$ i b se vrši njihovim iterativnim ažuriranjem metodom gradijentnog spusta. U svakoj iteraciji računaju se parcijalni izvodi funkcije gubitaka (20) u odnosu na $\mathbf{w}$ i b:

$$\begin{aligned} \frac{\partial J}{\partial w_j} &= \frac{1}{n} \sum_{i=1}^{n} \left(h_{\mathbf{w},b}(\mathbf{x}_i) - y_i \right) x_i^{(j)}, \quad j = 1, 2, \ldots, m, \\ \frac{\partial J}{\partial b} &= \frac{1}{n} \sum_{i=1}^{n} \left(h_{\mathbf{w},b}(\mathbf{x}_i) - y_i \right), \end{aligned} \tag{21}$$

nakon čega dolazi do njihovog ažuriranja:

$$\begin{aligned} w_j &:= w_j - \alpha \frac{\partial J}{\partial w_j} \quad j = 1, 2, \ldots, m, \\ b &:= b - \alpha \frac{\partial J}{\partial b}, \end{aligned} \tag{22}$$

gdje α predstavlja korak učenja (eng. *learning rate*) kojim se utiče na stepen promjene koeficijenata u svakoj iteraciji, odnosno brzinu konvergencije.

Veći korak učenja vodi ka bržoj konvergenciji, međutim njegova vrijednost mora biti pažljivo odabrana kako bi algoritam uopšte imao mogućnost konvergencije. Uzimajući u obzir da je funkcija gubitaka konveksna funkcija koja garantuje konvergenciju, njen rast tokom iteracija ukazuje na preveliki korak učenja koji je potrebno smanjiti. LR postiže bržu konvergenciju kada se podaci nalaze u sličnom opsegu vrijednosti [21], stoga se preporučuje skaliranje podataka prije primjene samog algoritma (Sekcija 3.3).

Ažuriranje parametara (22) se ponavlja unaprijed definisan maksimalan broj iteracija (eng. *maximum iterations*), ili dok se na osnovu praćenja krive učenja koja prikazuje promjenu vrijednosti funkcije gubitaka tokom iteracija ne zaključi da je došlo do zasićenja, te da nema smisla nastavljati proces treniranja.

Kako u radu sa velikim setovima podataka računanje gradijenta (22) nezavisno za svaki parametar i uzorak predstavlja veoma računski zahtjevan posao, jedan od mogućih pojednostavljenja predstavlja stohaistički gradijentni spust (eng. *stochaistic gradient descent*). Stohaistički gradijentni spust u svakoj iteraciji računa gradijente samo na osnovu jednog, slučajno odabranog, uzorka iz trening seta. Kompromis između standardnog i stohaističkog gradijentnog spusta je gradijentni spust nad manjim setom podataka (eng. *mini-batch gradient descent*) gdje se prilikom treniranja u svakoj iteraciji koriste slučajni podsetovi trening seta [21].

Danas se mogu naći naprednije implementacije LR koje koriste optimizovane metode za efikasniju izgradnju modela. Jedna od takvih implementacija je *LogisticRegression* u *scikit-learn* biblioteci, koja nadograđuje osnovne principe LR modernim tehnikama optimizacije, koje omogućavaju brže i preciznije podešavanje koeficijenata i hiperparametara modela. Doprinos ovih optimizacija se posebno ističe u radu sa velikim skupovima podataka i višeklasnoj klasifikaciji.

4.2 K-najbližih susjeda

Algoritam K-najbližih susjeda jedan je od najjednostavnijih predstavnika nadgledanih algoritama. Bazira se na predikciji klase uzorka na osnovu klase njemu najsličnijih uzoraka, pri čemu se njihova međusobna udaljenost u prostoru koristi kao mjera sličnosti. Kako su podaci često opisani karakteristikama predstavljenim različitim fizičkim veličinama, različitih opsega vrijednosti, standardizacija podataka (Sekcija 3.3) prije korišćenja algoritma K-najbližih susjeda može značajno poboljšati njegove performanse [34].

Cilj KNN-a je predikcija klase neoznačenog uzorka $\mathbf{x}'$ predstavljenog sa m karakteristika $(x'^{(1)}, x'^{(2)}, x'^{(3)}, ..., x'^{(m)})$. Algoritam vrši predikciju klase $\hat{y}'$ na osnovu označenog trening seta $D = \{(\mathbf{x_i}, y_i) | \, |D| = n, \, \mathbf{x_i} \in \mathbb{R}^m, \, y_i \in \mathbb{R}\}$, gdje n predstavlja broj uzoraka koji su opisani sa m karakteristika. Udaljenost uzorka $\mathbf{x}'$ od svakog uzorka trening seta $\mathbf{x}_i$ se računa Minkowskom mjerom (p-normom) datom formulom:

$$\|\mathbf{x}' - \mathbf{x}_i\|^p = \left(\sum_{j=1}^{m} \left| x'^{(j)} - x_i^{(j)} \right|^p \right)^{1/p}, \quad i = 1, 2, \ldots, n, \tag{23}$$

gdje za specijalan slučaj kada je $p = 2$ govorimo Euklidskoj udaljenosti, koja je podrazumijevana.

Dobijene udaljenosti se sortiraju, nakon čega se bira K najbližih susjeda, odnosno K uzoraka sa najmanjom udaljenošću od $\mathbf{x}'$. Hiperparametrom K se utiče na veličinu posmatrane okoline, odnosno broj susjeda koji su uključeni u proces predikcije. Klasa novog uzorka $\mathbf{x}'$ je klasa kojoj pripada većina susjeda, stoga se predlaže da K bude neparan broj [34], kako ne bi dolazilo do situacija u kojima su klase podjednako zastupljene. Opisani algoritam je dat pseudokodom 1.

Algoritam 1: K-najbližih susjeda

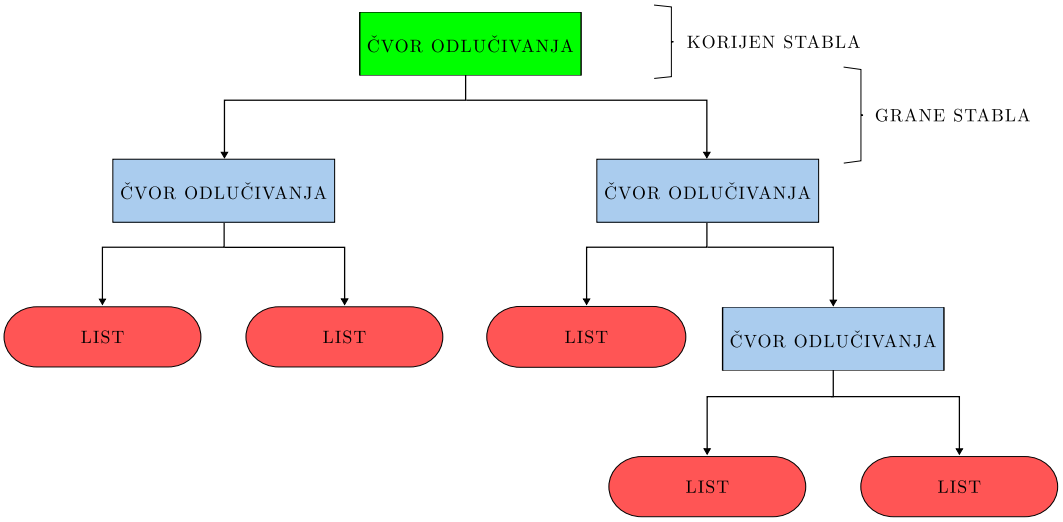
Data: Trening set $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, novi uzorak čija se klasa treba utvrditi $\mathbf{x}'$
Result: $\hat{y}' \leftarrow$ predikcija klase uzorka $\mathbf{x}'$
$K \leftarrow$ broj najbližih susjeda;
$p \leftarrow$ željena mjera udaljenosti;
for $i \leftarrow 1$ *to* n **do**
  $d_i \leftarrow \|\mathbf{x}' - \mathbf{x}_i\|^p$
end
Sortirati vektor udaljenosti $\mathbf{d}$ od najmanje ka najvećoj i odabrati prvih K najbližih susjeda;
$\hat{y}' \leftarrow$ klasa kojoj pripada većina odabranih susjeda;
return $\hat{y}'$

Optimalan broj najbližih susjeda se određuje tokom procesa treninga, postupkom unakrsne validacije (Sekcija 2). Veći broj susjeda povećava posmatranu okolinu, odnosno broj susjeda koji utiču na proces klasifikacije. Međutim, u radu [35] se ističe da proces predikcije na osnovu klase kojoj pripada većina susjeda nije uvijek dobro rješenje. Naime, neizbalansiranost klasa usljed znatno većeg broja pripadnika većinske klase, dovodi do njihovog čestog pojavljivanja među susjedima, te dominantnog uticaja pri predikciji. Osim primjene tehnika balansiranja za rješavanje problema neizbalansiranosti (Sekcija 3.4), rad [36] uvodi dodjeljivanje težina susje-

dima obrnuto srazmjerno njihovoj udaljenosti, čime se povećava uticaj uzoraka koji se nalaze bliže u prostoru pri procesu odlučivanja.

4.3 Stablo odlučivanja

Stabla odlučivanja su algoritmi nadgledanog mašinskog učenja koji zauzimaju značajno mjesto u rješavanju problema klasifikacije. Binarna stabla odlučivanja se sastoje od čvora, grana i listova, kao što je ilustrovano na slici 13. Čvorovi predstavljaju uslov podjele u odnosu na određenu karakteristiku uzoraka seta podataka, grane moguće vrijednosti karakteristike, dok list stabla odlučivanja predstavlja klasu u koju je uzorak svrstan. Za kontinualne karakteristike vrši se podjela seta u dva podseta poređenjem vrijednosti karakteristike uzorka sa pragom izabranim na određeni način [37]. Takođe, kod binarnih karakteristika ili za kategorijske karakteristike kodirane na odgovarajući način, opisane u Sekciji 3.2, vrši se podjela na dva podseta u zavisnosti od kategorije uzorka. Opisanim postupkom, na nivou svakog čvora vrši se podjela seta uzoraka R na set uzoraka desnog sina (R_D) i set uzoraka lijevog sina (R_L), pri čemu važi $R = R_D \cup R_L$. Stablo se formira tako što se proces podjele na podsetove rekurzivno ponavlja, počevši od novodobijenih čvorova koji se proglašavaju korijenima, do lista.



Slika 13: Ilustracija strukture stabla odlučivanja.

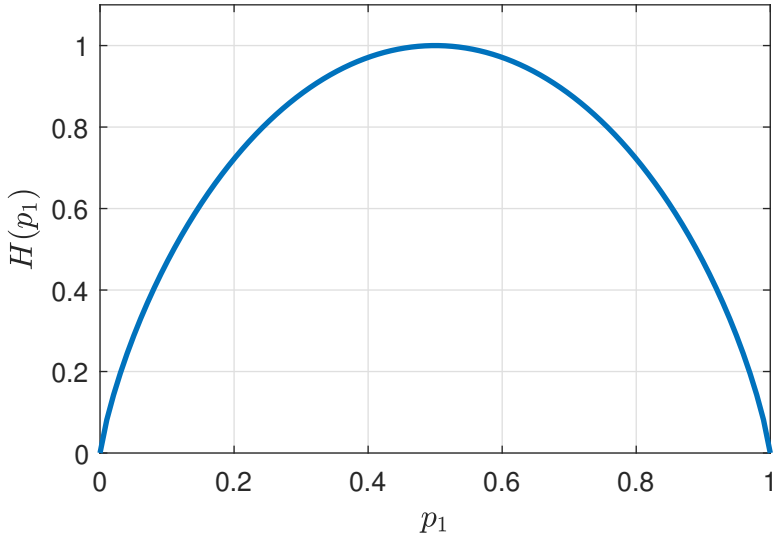
Za jedan set uzoraka može se kreirati veliki broj različitih stabala odlučivanja, stoga je u toku procesa treniranja potrebno utvrditi najbolju strukturu. Algoritmi treniranja rekurzivno vrše odabir karakteristike za podjelu na nivou čvora koji će izvršiti najčistiju podjelu seta podataka R, koji je doveden u čvor, na dva najčistija podseta R_L i R_D. Podset je utoliko čistiji ukoliko u sebi većinski sadrži pripadnike

jedne klase. U tu svrhu prilikom procesa treniranja stabla odlučivanja upotrebljava se CART (*Classification And Regression Trees*) algoritam [21].

Čistoća seta podataka mjeri se entropijom [8] i u slučaju binarne klasifikacije definisana je:

$$H(p_1) = -p_1 \log_2(p_1) - (1 - p_1)\log_2(1 - p_1), \tag{24}$$

gdje $H(p_1)$ predstavlja entropiju čvora, dok p_1 predstavlja broj pripadnika pozitivne klase u odnosu na ukupan broj uzoraka dospjelih u čvor na osnovu vrijednosti karakteristike koja predstavlja granu. Ukoliko se radi o setu podataka koji sadrži samo predstavnike jedne klase entropija je 0, što govori da se radi o apsolutno čistom setu. Jednak ili približno jednak broj uzoraka obje klase ($p_1 \approx 0.5$) vodi maksimalnoj vrijednosti entropije, kao što je ilustrovano na slici 14. Stoga manja vrijednost entropije ukazuje na čistiji set.



Slika 14: Funkcija entropije $H(p_1)$ definisana jednačinom (24).

U kojoj mjeri će podjela na osnovu razmatrane karakteristike voditi čistijim podsetovima, odnosno smanjenju entropije, pokazuje dobitak informacija (eng. *Information Gain* - IG), koji se računa po formuli:

$$\text{IG} = H(p_1) - (\frac{|R_L|}{|R|}H(p_1)^L + \frac{|R_D|}{|R|}H(p_1)^D), \tag{25}$$

gdje $H(p_1)$ predstavlja entropiju razmatranog čvora, dok $H(p_1)^L$ i $H(p_1)^D$ predstavljaju entropiju lijevog i desnog podseta, koji bi se dobili odabirom konkretne karakteristike za dalje grananje. Osim IG-a stabla odlučivanja mogu koristiti i Džini koeficijent (eng. *Gini index*) kao metriku za evaluaciju kvaliteta podjele seta uzoraka na nivou čvora [8].

U slučaju binarne klasifikacije CART algoritam se sastoji od sljedećih koraka:

Korak 1: **Početak u korijenu:** Svi uzorci trening seta nalaze se u korijenom čvoru;

Korak 2: **Računanje IG-a:** Za svaku karakteristiku i sve moguće vrijednosti praga računa se IG;

Korak 3: **Podjela seta na osnovu karakteristike:** Set podataka se dijeli na dva podseta na osnovu one karakteristike i praga koji su dali najveću vrijednost IG-a;

Korak 4: **Rekurzivni proces 4:** Proces se rekurzivno ponavlja za svaki podset, slijedeći korake 2 i 3, sve dok se ne ispune neki od unaprijed definisanih kriterijuma zaustavljanja. Česti kriterijumi zaustavljanja su:

- ograničavanje dubine stabla (eng. *maximum depth*) - postavljanje maksimalne dubine stabla koja predstavlja broj čvorova od korijena do najudaljenijeg lista;
- minimalni broj uzoraka za razdvajanje (eng. *minimum samples split*) - minimalan broj uzoraka u čvoru potreban za njegovo dalje grananje;
- minimalan broj uzoraka u listu (eng. *minimum samples leaf*);
- maksimalan broj listova (eng. *maximum leaf nodes*);
- postavljanje praga IG-a neophodnog za dalje grananje;

Svaki list će predstavljati klasu kojoj pripada najveći broj uzoraka trening seta, dospjelih u taj list prethodno opisanom procedurom. Novi uzorak se sprovodi granama stabla na osnovu vrijednosti svojih karakteristika, do jednog od listova, koji reprezentuje predikciju njegove klase.

Stabla odlučivanja su veoma slična načinu na koji čovjek donosi odluke što ih čini veoma interpretabilnim i lakim za razumijevanje. Za razliku od većine drugih algoritama, može se jasno ispratiti klasifikacioni proces, kao i značaj i uticaj određenih karakteristika pri procesu donošenja odluka [2]. Karakteriše ih robusnost na *outlier*-e i mogućnost dobrog rada sa neskaliranim podacima i nedostajućim vrijednostima [8, 38]. Upravo ih ovo čini pogodnim i izuzetno korišćenim algoritmom u problemima klasifikacije i regresije.

Glavni nedostatak ovog algoritma predstavlja njegova sklonost ka preprilagođavanju trening podacima [2]. Jedan od razloga preprilagođavanja jeste činjenica da sa povećavanjem kompleksnosti stabla odlučivanja, odnosno njegove dubine i broja listova rastu njegove performanse na trening setu. Međutim, na ovaj način dobijene dobre performanse na trening setu ne garantuju njegovu dobru predikciju

klase novih uzoraka, stoga se posebna pažnja mora posvetiti utvrđivanju optimalne kompleksnosti stabla. U tu svrhu, osim definisanja kriterijuma zaustavljanja koji će kontrolisati razgranavanje stabla tokom treninga, koriste se metode obrezivanja (eng. *pruning methods*). Metode obrezivanja uklanjaju čvorove i grane stabla, ukoliko se njihovim uklanjanjem popravlja tačnost klasifikacije na validacionom setu u procesu unakrsne validacije [37].

Drugi razlog za preprilagođavanje trening podacima jeste činjenica da promjena malog broja uzoraka trening seta može dovesti do generisanja potpuno različite strukture stabla. Najbolji način za prevazilaženje problema velike osjetljivosti stabla odlučivanja na male promjene uzoraka seta za treniranje jeste korišćenje ansambla stabala, o čemu će biti više riječi u nastavku.

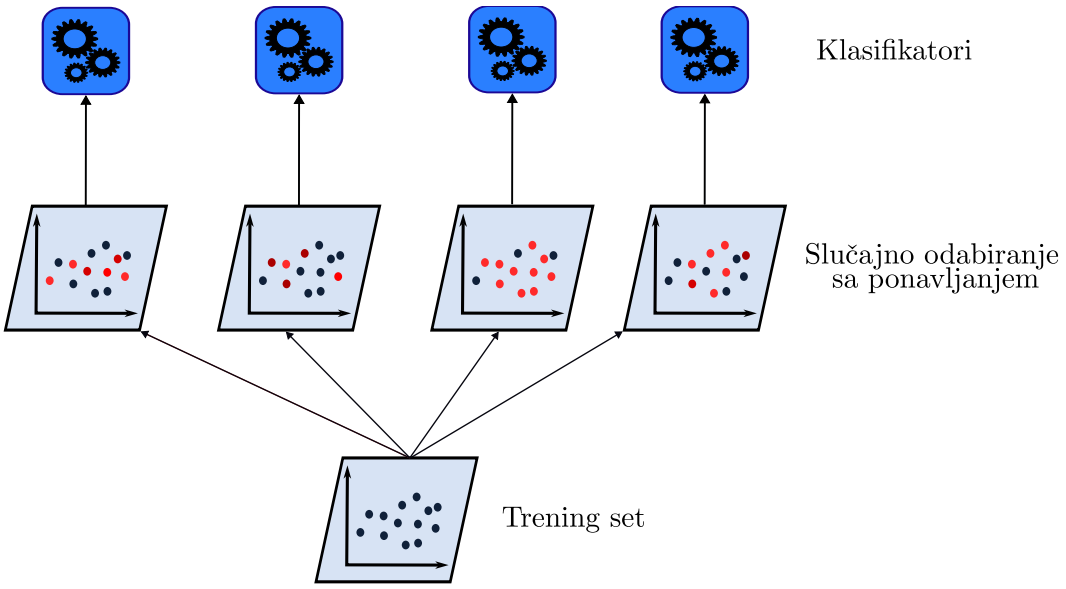
4.4 Ansambl metodi bazirani na stablu odlučivanja

Prilikom rješavanja određenog problema pokazuje se da skupina nezavisnih ljudi najčešće može donijeti odluke bolje od pojedinca koji posjeduje stručnost iz date problematike. Na istom principu, ansambl metode donose konačnu odluku o klasi uzorka uzimajući u obzir predikcije pojedinačnih klasifikatora. Kako bi ovaj pristup bio efikasan neophodno je obezbijediti nezavisnost klasifikatora, što se postiže primjenom različitih algoritama ili istih algoritama treniranih na različitim setovima podataka. Ukoliko se ansambl sastoji od istih klasifikatora primjenjuju se tehnike usrednjavanja i tehnike pojačavanja (eng. *boosting*) prilikom izgradnje ansambla.

Kod tehnika usrednjavanja više nezavisnih klasifikatora, generisanih za različite podsetove trening seta, donose odluku glasovima većine. U zavisnosti od načina kreiranja podseta razlikuju se:

- *bagging* (*bootstrap aggregating*) metoda - podset se generiše primjenom *bootstrap* odabiranja koje podrazumijeva slučajni odabir uzoraka trening seta sa ponavljanjem, objašnjen u Sekciji 3.4.2. Primjena *bagging*-a u kreiranju podsetova originalnog trening seta je ilustrovana na slici 15;
- *pasting* metoda - slučajan odabir uzoraka bez ponavljanja, gdje svaki uzorak trening seta može biti odabran samo jednom pri kreiranju određenog podseta;

Oba pristupa su preporučljiva zbog svoje mogućnosti paralelizacije i pozitivnog uticaja na redukciju varijanse modela. Korišćenje odabiranja sa ponavljanjem kod *bagging* metoda doprinosi većoj raznolikosti podsetova, što često vodi ka kreiranju boljih modela [21]. Najpoznatiji predstavnik ove grupe algoritama jeste RF.



Slika 15: Primjer *bagging* metode za pripremu seta podataka.

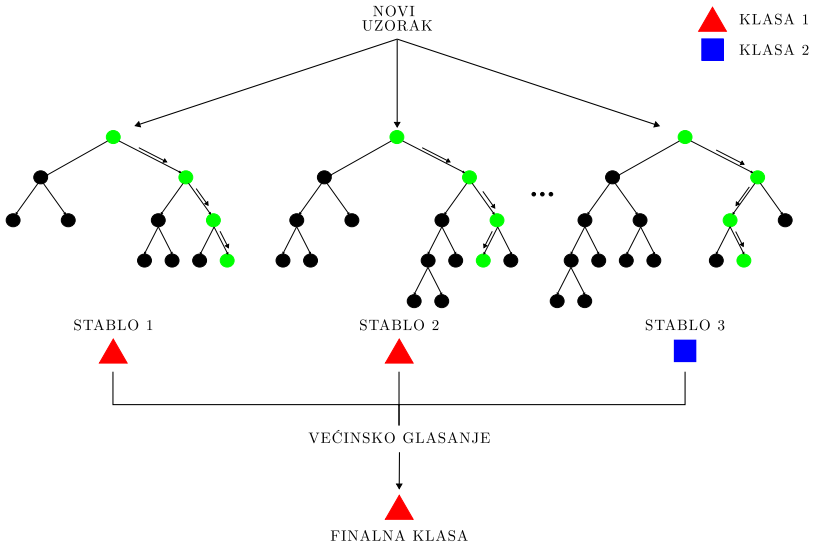
Kreiranje ansambla tehnikom pojačavanja predstavlja sekvencijalno generisanje tzv. slabih klasifikatora koji nastoje ispraviti greške prethodnika. Termin slabi klasifikator se odnosi na jednostavna stabla odlučivanja sa značajno manjom dubinom od moguće za analizirani set podataka. Ukoliko bi se koristili nezavisno jedan od drugog, slabi klasifikatori bi se odlikovali velikim *bias*-om i predikcijom malo boljom od slučajne. Adekvatnim kombinovanjem izlaza slabih klasifikatora, koji uče na osnovu grešaka prethodnika, popravljaju se performanse modela. Sa aspekta brzine izvršavanja ovaj pristup, zbog nemogućnosti paralelizacije, traje duže od generisanja ansambla tehnikama usrednjavanja. Glavni predstavnici *boosting* algoritama su AdaBoost (*Adaptive Boosting*), GBDT (*Gradient Boosted Decision Tree*), XGBoost i CatBoost.

4.4.1 Random Forest

RF predstavlja kolekciju stabala odlučivanja gdje se svako stablo nezavisno generiše koristeći različite podsetove originalnog trening seta podataka dobijenih primjenom *bagging* metode. Dodatno se može vršiti i slučajan odabir karakteristika koje će se koristiti za treniranje stabla odlučivanja u svakoj iteraciji [39]. Ovaj pristup uvodi slučajnost u kreiranje nezavisnih klasifikatora, koji se i zbog značajno manjeg broja čvorova (karakteristika) mogu smatrati slabim klasifikatorima. Međutim, zajedničkim donošenjem odluke o pripadnosti klasi izbjegava se problem preprilagođavanja stabla odlučivanja trening setu [39], što RF čini jakim klasifikatorom.

Konačna predikcija na ovaj način generisanog ansambla donosi se većinom gla-

sova kao što je ilustrovano na slici 16 (klase su predstavljene crvenim trouglom i plavim kvadratom). Broj klasifikatora (eng. *number of estimators*) koji čini ansambl predstavlja jedan od hiperparametara ovog algoritma. Povećavanjem broja osnovnih klasifikatora povećava se kompleksnost modela i vrijeme predikcije. Međutim, model sa većim brojem klasifikatora najčešće vodi boljim performansama modela [40] stoga se njihov optimalan broj, sa aspekta performansi, kompleksnosti i brzine određuje u procesu unakrsne validacije.



Slika 16: Ilustracija procesa klasifikacije kod RF-a. Uzorak se klasifikuje u klasu u koju ga je klasifikovala većina klasifikatora.

RF zadržava sve prednosti koje imaju stabla odlučivanja. U slučaju neizbalansiranih i velikih setova podataka sa hiljadama karakteristika, RF postiže visoke performanse uz veliku brzinu [41].

4.4.2 AdaBoost

AdaBoost predstavlja još jedan primjer ansambl metode. Poput RF-a koristi slabe klasifikatore (manja stabla odlučivanja), sa tom razlikom da je kod AdaBoost-a dubina pojedinačnog stabla odlučivanja obično 1 (stabla sa korijenom i dva lista - *stump*) [42]. Za razliku od RF-a koji za generisanje svakog novog stabla odlučivanja koristi različite podsetove podataka i karakteristika, AdaBoost pokušava nad istim setom postići bolje rezultate tako što će naredni klasifikator pokušati ispraviti greške prethodnika. Dodatno, slabi klasifikator se generiše korišćenjem jedne ili malog broja karakteristika. Odabrani trening set služi za utvrđivanje greške klasifikacije dobijenog slabog klasifikatora i davanje veće težine pogrešno klasifikovanim uzorcima. Dodjela različite težine uzorcima će voditi ka tome da za svaki naredni klasifikator

prethodno pogrešno klasifikovani uzorci imaju veću vjerovatnoću da budu dio trening seta [42, 43]. Drugi način povećanja uticaja prethodno loše klasifikovanih uzoraka jeste korišćenje ponderisanog IG-a [44]. Na oba načina svaki sljedeći klasifikator se fokusira na poboljšavanje klasifikacije prethodno loše klasifikovanih uzoraka. Dodatno, ukupna greška klasifikacije slabog klasifikatora se koristi za utvrđivanje težine njegovog glasa prilikom donošenja konačne odluke o klasi uzorka. Na ovaj način klasifikatori koji su bolje klasifikovali prethodno loše klasifikovane uzorke imaju veći značaj prilikom donošenja konačne odluke.

Težinski koeficijenti uzoraka se ažuriraju nakon generisanja svakog slabog klasifikatora. Prije primjene početnog klasifikatora, svi uzorci imaju jednaku težinu $w_0 = \frac{1}{n}$, gdje n predstavlja broj uzoraka trening seta.

Greška j-tog slabog klasifikatora nad trening setom $D = \{(\mathbf{x_i}, y_i) | \, |D| = n, \, \mathbf{x_i} \in \mathbb{R}^m, \, y_i \in \mathbb{R}\}$, gdje n predstavlja broj uzoraka koji su opisani sa m karakteristika je:

$$r_j = \frac{\sum_{i=1}^{n} w_i \cdot \left| \hat{y}_i^{(j)} - y_i \right|}{\sum_{i=1}^{n} w_i}, \tag{26}$$

gdje w_i predstavlja težinski koeficijent i-tog uzorka, $\hat{y}_i^{(j)}$ predikciju klase j-tog klasifikatora za i-ti uzorak (0 ili 1), dok y_i predstavlja oznaku stvarne klase uzorka. Na osnovu greške (26) svakom klasifikatoru se nezavisno dodjeljuje težina glasa α_j:

$$\alpha_j = \eta \log \left(\frac{1 - r_j}{r_j} \right), \tag{27}$$

gdje η predstavlja korak učenja (podrazumijevana vrijednost $\eta = 1$) [21].

Ažuriranje težina klasifikovanih uzoraka j-tim klasifikatorom vrši se po formuli:

$$w_i \leftarrow \begin{cases} w_i, & \hat{y}_i = y_i \\ w_i \exp(\alpha_j), & \hat{y}_i \neq y_i \end{cases}, \quad i = 1, 2, \ldots, n, \tag{28}$$

čime se za naredni klasifikator daje veća težina loše klasifikovanom uzorku, nakon čega se vrši normalizacija težinskih koeficijenata dijeljenjem sa $\sum_{i=1}^{n} w_i$. Rad [42] ističe da se sagledavanjem težina uzoraka mogu utvrditi *outlier*-i, čija će težina biti znatno veća u odnosu na ostale zbog njihove često pogrešne klasifikacije.

AdaBoost donosi odluku o klasi novog uzorka $\mathbf{x}'$ na osnovu formule:

$$\hat{y}' \leftarrow \begin{cases} 0, & \sum_{j=1}^{T} \alpha_j \cdot \mathbf{1}(\hat{y}'^{(j)} = 0) \geq \sum_{j=1}^{T} \alpha_j \cdot \mathbf{1}(\hat{y}'^{(j)} = 1) \\ 1, & \sum_{j=1}^{T} \alpha_j \cdot \mathbf{1}(\hat{y}'^{(j)} = 0) < \sum_{j=1}^{T} \alpha_j \cdot \mathbf{1}(\hat{y}'^{(j)} = 1) \end{cases}, \tag{29}$$

gdje T predstavlja finalni broj klasifikatora, dok $\mathbf{1}(\cdot)$ predstavlja indikatorsku funkciju koja ima vrijednost 1 ukoliko je uslov unutar zagrade ispunjen, a 0 u suprotnom. Na ovaj način težina glasa α_j svakog klasifikatora utiče na njegovu važnost

pri donošenju odluke. Oni klasifikatori koji vrše slučajnu predikciju, poput bacanja novčića, će biti zanemareni, dok će klasifikatori loših performansi ($\alpha_j < 0$) negativno uticati na ukupnu sumu, smanjujući njenu vrijednost.

4.4.3 Gradient Boosted Decision Tree

GBDT spada u grupu *boosting* algoritama koji se zasnivaju na sekvencijalnom dodavanju klasifikatora ansamblu u cilju smanjivanja greške predikcije. Ukoliko se ansambl F_0 sastoji od jednog klasifikatora čiji je zadatak predikcija zavisne promjenljive Y na osnovu nezavisnih promjenljivih X, matematička reprezentacija ovog modela može se zapisati kao:

$$y = h_{F_0}(\mathbf{x}) + \varepsilon_0, \tag{30}$$

gdje kod klasifikacije $h_{F_0}(\mathbf{x})$ predstavlja procijenjenu vrijednost hipoteze prvog klasifikatora - vjerovatnoću pripadnosti pozitivnoj klasi, dok ε_0 predstavlja rezidualnu grešku ansambla, a y oznaku klase.

Kod GBDT-a, novi klasifikator G pokušava poboljšati prethodnu klasifikaciju estimiranjem rezidualne greške do tada kreiranog ansambla za svaki list i sve uzorke, njenim prevođenjem u vjerovatnoću pripadnosti određenoj klasi.

Klasifikator G_t kreiran na ovaj način se dodaje ansamblu, čime se teži da se ukupna greška modela smanjuje, pri čemu je:

$$y = h_{F_t}(\mathbf{x}) + \varepsilon_t, \quad t = 1, 2, \ldots, T, \tag{31}$$

gdje je $h_{F_t}(\mathbf{x})$ vrijednost hipoteze modela u t-toj iteraciji, odnosno vjerovatnoća pripadnosti pozitivnoj klasi estimirana novodobijenim ansamblom F_t kreiranim dodavanjem novog klasifikatora G_t prethodnom ansamblu F_{t-1}. Opisani postupak se iterativno ponavlja unaprijed definisani broj puta T.

Inicijalni ansambl F_0, sastoji se od jednog stabla odlučivanja G_0 koji sadrži samo korijeni čvor koji je ujedno i list. Vrijednost predikcije klasifikatora G_t se označava sa $\boldsymbol{\gamma}_{G_t}$, čija dimenzionalnost zavisi od broja listova klasifikatora G_t. Kod inicijalnog modela γ_{F_0} je skalar i estimira se kao logaritam šansi, za sve uzorke trening seta. Logaritam šansi (*log(odds)*) predstavlja logaritam odnosa vjerovatnoće da uzorak pripada pozitivnoj klasi i vjerovatnoće pripadnosti negativnoj klasi:

$$\gamma_{F_0} = \log(\text{odds}) = \log\left(\frac{p}{1-p}\right) = \log\left(\frac{BP}{BN}\right), \tag{32}$$

gdje p predstavlja vjerovatnoću pojavljivanja pozitivne klase u trening setu, odnosno $p = \frac{BP}{BP+BN}$, pri čemu BP i BN predstavljaju broj pozitivnih i negativnih uzoraka trening seta respektivno.

Za svaku narednu iteraciju dodaje se prethodnom ansamblu novo regresiono stablo sa J terminalnih regiona. Regresiona stabla su tip stabala odlučivanja, gdje listovi predstavljaju estimaciju kontinualne numeričke vrijednosti. Kod GBDT-a regresiona stabla se konstruišu težeći da estimiraju rezidualne greške prethodnog ansambla, a zatim za tako konstruisano stablo, računaju se vrijednosti listova $\boldsymbol{\gamma}_{G_t}$ predstavljene u obliku logaritma šansi. Kombinovanjem izlaza klasifikatora prethodnog ansambla i dodatog klasifikatora, te prevođenjem tako dobijenog izlaza u vjerovatnoću pripadnosti pozitivnoj klasi ansambl vrši klasifikaciju uzoraka.

Rezidualna greška ε za inicijalni ansambl F_0, koji se sastoji samo od jednog klasifikatora G_0 kojeg zapravo čini jedan list, računa se za svaki uzorak $\mathbf{x}_i$:

$$\varepsilon_{F_0,i} = y_i - h_{F_0} = y_i - \frac{e^{\gamma_{F_0}}}{1 + e^{\gamma_{F_0}}}, \quad i = 1, 2, \ldots, n, \tag{33}$$

gdje je h_{F_0} skalar, jednak za sve uzorke i dobijen pretvaranjem izlaza γ_{F_0} inicijalnog modela u vjerovatnoću pripadnosti pozitivnoj klasi.

Za novi klasifikator G se bira ono stablo čija je estimacija rezidualne greške za svaki uzorak dobijena usmjeravanjem konačne klasifikacije ka smanjenju funkcije gubitaka prethodnog ansambla [45]. Kako bi se automatizovao proces minimizacije za funkciju gubitka se bira diferencijabilna funkcija $L(y_i, h_{F_t}(\mathbf{x}_i))$, jer njen parcijalni izvod daje informaciju o uticaju mogućih estimacija rezidualne greške na njenu vrijednost. Kako se u problemu koji se ovdje razmatra radi o binarnoj klasifikaciji, funkcija gubitaka koja se minimizuje je negativi logaritamski gubitak (eng. *log(loss) function*) i za jedan uzorak $\mathbf{x}_i$ je data formulom:

$$L(y_i, h_{F_t}(\mathbf{x}_i)) = -\left[y_i \log(h_{F_t}(\mathbf{x}_i)) + (1 - y_i) \log(1 - h_{F_t}(\mathbf{x}_i))\right], \tag{34}$$

za svako $i = 1, 2, ..., n$, gdje y_i predstavlja stvarnu oznaku klase uzorka $\mathbf{x}_i$, dok $h_{F_t}(\mathbf{x}_i)$ predstavlja procjenu vrijednosti hipoteze ansambla F_t, odnosno vjerovatnoću njegove pripadnosti pozitivnoj klasi.

Za svaki od uzoraka $\{\mathbf{x}_i\}_{i=1}^n$ klasifikatora G_t, vrijednost reziduala $\varepsilon_{F_{t-1},i}$ se estimira kao:

$$\varepsilon_{F_{t-1},i} = -\left[\frac{\partial L(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i))}{\partial \gamma_{F_{t-1}}(\mathbf{x_i})}\right], \quad i = 1, 2, \ldots, n, \tag{35}$$

gdje je L dato formulom (34), $h_{F_{t-1}}(\mathbf{x}_i)$ je hipoteza prethodnog ansambla i jednaka je:

$$h_{F_{t-1}}(\mathbf{x}_i) = \frac{e^{\gamma_{F_{t-1}}(\mathbf{x}_i)}}{1 + e^{\gamma_{F_{t-1}}(\mathbf{x}_i)}}, \quad i = 1, 2, \ldots, n. \tag{36}$$

$\gamma_{F_t}(\mathbf{x}_i)$ je jednaka težinskoj sumi izlaza svih klasifikatora ansambla F_{t-1} i dodatog klasifikatora G_t:

$$\gamma_{F_t}(\mathbf{x}_i) = \gamma_{F_{t-1}}(\mathbf{x}_i) + \eta \sum_{j=1}^{J} \gamma_{G_t,j} \cdot \mathbf{1}(\mathbf{x}_i \in R_{G_t,j}), \quad t = 1, 2, \ldots, T, \tag{37}$$

gdje hiperparametar redukcije η predstavlja težinu dodijeljenu izlazu svakog klasifikatora, čime se određuje uticaj svakoga od njih na donošenje konačne odluke ansambla. U prethodnoj formuli u sumi će biti samo jedan element različit od nule i to je element koji odgovara izlazu j-tog lista klasifikatora G_t u koji je sproveden uzorak $\mathbf{x}_i$, što je obilježeno indikatorskom funkcijom $\mathbf{1}(\mathbf{x}_i \in R_{G_t,j})$. Naime, svaki od uzoraka trening seta prolazi kroz grane stabla odlučivanja, na način opisan u Sekciji 4.3, sve dok ne završi u jednom od listova $R_{G_t,j}$. Optimalna vrijednost jednog lista $\gamma_{G_t,j}$ je ona čijim dodavanjem prethodno kreiranom ansamblu F_{t-1}, dolazi do minimizovanja funkcije gubitaka L:

$$\gamma_{G_t,j} = \underset{\gamma}{\operatorname{argmin}} \sum_{\mathbf{x}_i \in R_{G_t,j}} L(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i) + \gamma), \quad j = 1, 2, \ldots, J, \tag{38}$$

za svaki uzorak $\mathbf{x}_i$ iz trening seta $\{\mathbf{x}_i\}_{i=1}^n$.

Funkcija gubitaka je na osnovu (34) i (36), za i-ti uzorak data formulom:

$$L(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i) + \gamma) = -y_i \cdot \left(\gamma_{F_{t-1}}(\mathbf{x}_i) + \gamma\right) + \log(1 + e^{\left(\gamma_{F_{t-1}}(\mathbf{x}_i) + \gamma\right)}), \tag{39}$$

primjenom Tejlorove ekspanzije drugog reda funkcija gubitaka se može aproksimirati [46] kao:

$$L(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i) + \gamma) \approx L(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i)) + G_{i,L} \cdot \gamma + \frac{1}{2} H_{i,L} \cdot \gamma^2, \quad i = 1, 2, ..., n, \tag{40}$$

$$G_{i,L} = \frac{\partial L(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i))}{\partial \gamma_{F_{t-1}}(\mathbf{x}_i)}, \quad i = 1, 2, ..., n, \tag{41}$$

$$H_{i,L} = \frac{\partial^2 L(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i))}{\partial^2 \gamma_{F_{t-1}}(\mathbf{x}_i)}, \quad i = 1, 2, ..., n, \tag{42}$$

čijim diferenciranjem dolazimo do optimalnih vrijednosti listova novog klasifikatora:

$$\gamma_{G_t,j} = \frac{\sum_{i=1}^n -G_{i,L} \cdot \mathbf{1}(\mathbf{x}_i \in R_{G_t,j})}{\sum_{i=1}^n H_{i,L} \cdot \mathbf{1}(\mathbf{x}_i \in R_{G_t,j})}, \quad j = 1, 2, \ldots, J, \tag{43}$$

gdje se svaka vrijednost terminalnih regiona računa samo na osnovu onih uzoraka koji se u njima nalaze, što je označeno indikatorskom funkcijom $\mathbf{1}(\mathbf{x}_i \in R_{G_t,j})$.

Primjenom prvog (35) i drugog izvoda funkcije gubitaka formula se svodi na:

$$\gamma_{G_t,j} = \frac{\sum_i^n \varepsilon_{F_{t-1},i} \cdot \mathbf{1}(\mathbf{x}_i \in R_{G_t,j})}{\sum_i^n h_{F_{t-1}}(\mathbf{x}_i) \cdot (1 - h_{F_{t-1}}(\mathbf{x}_i)) \cdot \mathbf{1}(\mathbf{x}_i \in R_{G_t,j})}, \quad j = 1, 2, \ldots, J. \tag{44}$$

Postupak od formule (35)-(44) se iterativno ponavlja za unaprijed definisani broj iteracija T. Predikcija klase $\hat{y}'$ novog uzorka $\mathbf{x}'$ računa se na osnovu vrijednosti

hipoteze krajnjeg ansambla F_T i unaprijed definisanog praga odlučivanja τ (najčešće 0.5):

$$\hat{y}' \leftarrow \begin{cases} 0, & h_{F_T}(\mathbf{x}') < \tau \\ 1, & h_{F_T}(\mathbf{x}') \geq \tau. \end{cases} \tag{45}$$

Preveliki broj iteracija može da dovede do preprilagođavanja modela podacima iz trening seta, stoga je neophodno pratiti grešku u iteracijama i dodavati nove klasifikatore sve dok se njihovim dodavanjem smanjuje greška na testnom setu. Dodatno, bolja generalizacija modela može se postići pažljivim odabirom parametra redukcije (eng. *shrinkage parameter*), koji predstavlja korak učenja čija vrijednost se kreće $0 < \eta \leq 1$. Friedman je u radu [45] empirijski pokazao da male vrijednosti ($\eta \leq 0.1$) vode znatno boljoj generalizaciji modela. Osim pažljivog odabira veličine modela i već pomenutog koraka učenja, efikasna metoda u borbi protiv preprilagođavanja trening setu je učenje stabala na različitim slučajnim podsetovima trening seta [47].

4.4.4 XGBoost

XGBoost algoritam predstavlja optimizaciju GBDT-a razvijenu od strane Tianqi Chena kao dio grupe Distributed (Deep) Machine Learning Community - DMLC [46]. Veliku pažnju i značaj ova biblioteka otvorenog koda je doživjela nakon brojnih takmičenja u rješavanju problema iz različitih oblasti primjenom mašinskog učenja, gdje je upravo ovaj algoritam pokazao najbolje rezultate [48].

XGBoost takođe predstavlja ansambl metodu zasnovanu na tehnici pojačavanja. U svakoj iteraciji novi klasifikator G_t se dodaje ansamblu F_{t-1} radi smanjivanja regularizovane funkcije gubitaka, koja je kod XGBoost-a data formulom:

$$L(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i) + \gamma) = \sum_{i=1}^{n} \left[\ell\left(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i) + \gamma\right)\right] + \xi J + \frac{1}{2}\lambda\gamma^2, \quad i = 1, 2, ..., n, \tag{46}$$

gdje ℓ predstavlja diferencijabilnu funkciju gubitaka kao kod GBDT-a (39), dok ostatak predstavlja regularizaciju uvedenu radi sprečavanja preprilagođavanja modela koja se sastoji od hiperparametra kompleksnosti stabla ξ (eng. *minimal split loss*) i regularizacionog hiperparametra λ.

Inicijalni anasambl F_0 sastoji se od jednog klasifikatora G_0 sa jednim korijenim čvorom koji ujedno predstavlja i list. Podrazumijevana vrijednost ovog lista, koja ujedno predstavlja izlaz klasifikatora i označava se sa γ_{F_0}, je 0 i data je u obliku logaritma šansi. Svaki od uzoraka će stoga imati istu vjerovatnoću pripadnosti pozitivnoj klasi koja iznosi 0.5, dobijenu primjenom formule (36), kao i rezidualnu grešku $\varepsilon_{F_0,i}$, računatu formulom (33).

U svakoj narednoj iteraciji $t = 1, 2, ..., T$ ansamblu se dodaje novi klasifikator G_t, koji predstavlja modifikovano regresiono stablo odlučivanja koje vrši estimaciju rezudualne greške, po formuli (37). Kako nije izvodljivo ispitati sve moguće strukture novog klasifikatora i izabrati onu optimalnu, koristi se pohlepni algoritam koji počinje u korijenu i iterativno dodaje grane stabla. Inicijalno se stablo odlučivanja G_t sastoji samo od korijenog čvora koji predstavlja list čija se optimalna vrijednost γ_{G_t} bira tako da se njenim dodavanjem prethodnom modelu F_{t-1} vrši minimizacija regularizovane funkcije gubitaka (46) aproksimiranom Tejlorovom ekspanzijom drugog reda kao kod GBDT-a (40), pri čemu se zanemaruje konstantna vrijednost ξJ u formuli (46) jer ne utiče na minimizaciju. Optimalna vrijednost ovog terminalnog regiona je data:

$$\gamma_{G_t} = -\frac{\sum_{\mathbf{x}_i \in R_{G_t}} G_{i,\ell}}{\sum_{\mathbf{x}_i \in R_{G_t}} H_{i,\ell} + \lambda}, \quad i = 1, 2, ..., n, \tag{47}$$

gdje $G_{i,\ell}$ i $H_{i,\ell}$ predstavljaju prvi i drugi izvod funkcije gubitaka $\ell(y_i, \gamma_{F_{t-1}}(\mathbf{x}_i))$ dat formulama (41)(42), dok R_{G_t} predstavlja set uzoraka dospjelih u terminalni region koji u slučaju korijenog čvora obuhvata čitav trening set.

Krećući se od korijena, na nivou svakog čvora, dodaju se dvije grane stabla i vrši se podjela seta uzoraka R na set uzoraka lijevog sina (R_L) i set uzoraka desnog sina (R_D), pri čemu važi $R = R_L \cup R_D$. Odabir praga i karakteristike za podjelu seta na nivou čvora pohlepno se bira tako da maksimizuje dobitak (eng. *gain* - g), odnosno vrijednost redukcije gubitaka nakon podjele predstavljenu formulom:

$$g = \frac{\left(\sum_{\mathbf{x}_i \in R_L} G_{i,\ell}\right)^2}{\sum_{\mathbf{x}_i \in R_L} H_{i,\ell} + \lambda} + \frac{\left(\sum_{\mathbf{x}_i \in R_D} G_{i,\ell}\right)^2}{\sum_{\mathbf{x}_i \in R_D} H_{i,\ell} + \lambda} - \frac{\left(\sum_{\mathbf{x}_i \in R} G_{i,\ell}\right)^2}{\sum_{\mathbf{x}_i \in R} H_{i,\ell} + \lambda}, \quad i = 1, 2, ..., n. \tag{48}$$

Opisanim postupkom vrši se grananje stabla dok se ne ispuni neki od unaprijed zadatih kriterijuma zaustavljanja (Sekcija 4.3), nakon čega se vrši njegovo obrezivanje koje je ključno u sprečavanju preprilagođavanja trening setu. Krećući se od listova ka čvoru stabla uklanjaju se svi oni čvorovi čiji dobitak nije veći od vrijednosti ξ, pod uslovom da neki od njegovih potomaka nije zadržan. Ovdje se ogleda i uloga regularizacionog hiperparametra λ, čijim povećavanjem dolazi do smanjenja vrijednosti dobitka na nivou čvora, odnosno do većeg obrezivanja stabla. Takođe, dodatno obrezivanje se vrši na nivou lista $\{R_{G_t,j}\}_{j=1}^{J}$ ukoliko suma Hesijana funkcije gubitaka ($H_{i,\ell}$) uzoraka koji pripadaju datom listu, nije veća od praga zadatog hiperparametrom minimalne težine djeteta (eng. *minimum child weight*).

U radu [46] kreatori ovog algoritma, osim pomenute regularizacije, predstavljaju brojne prednosti na nivou algoritma i sistema:

- Aproksimativni pohlepni algoritam;

- Aproksimacija težišnih kvantila;
- Mogućnost treninga i predikcije nad podacima sa nedostajućim vrijednostima;
- Povećavanje brzine računskih operacija upotrebom paralelizacije i distribuiranosti;
- Računske operacije van glavne memorije;

Kako bi osposobili algoritam za primjenu nad velikim setovima podataka, koristi se aproksimativni pohlepni algoritam. Ovaj algoritam na nivou svakog čvora pohlepno vrši grananje koje u tom trenutku vodi maksimizaciji dobitka (48). Međutim, ukoliko bi se na nivou svakog čvora vršilo ispitivanje svih mogućih pragova seta podataka sa velikih brojem uzoraka, treniranje bi trajalo predugo. Stoga se kao pragovi za testiranje koriste težišni kvantili, čime se smanjuje broj pragova za ispitivanje. Naime, svaki uzorak dobija određenu težinu koja je jednaka njegovom Hesijanu ($H_{i,\ell}$) funkcije gubitaka, odnosno težine uzorka su veće sa većim stepenom njihove pogrešne klasifikacije u prethodnom modelu. Kvantili se biraju tako da dijele set na podsetove sa jednakom sumom težina uzoraka, čime se osim povećavanja brzine usljed smanjenja broja ispitanih pragova dodatna pažnja posvećuje pogrešno klasifikovanim uzorcima. Ovaj, takođe kompleksan, proces je dodatno ubrzan primjenom algoritama za skiciranje (eng. *sketches algorithms*) koji su u stanju da brzo aproksimiraju rješenje, kao i redukcijom broja posmatranih karakteristika i uzoraka u treningu.

Traženje optimalnog drveta u svakoj iteraciji koje zahtijeva stalno računanje gradijenata i Hesijana funkcije gubitaka predstavlja izazov zbog velike vremenske i memorijske kompleksnosti. Upravo u ovu svrhu, XGBoost upotrebljava prednosti keš memorije za smještanje izvoda funkcija, čime se smanjuje vrijeme pristupa memoriji. Takođe, algoritam upotrebljava resurse hard diska za skladištenje komprimovanih podataka.

Podaci usljed greške prilikom sakupljanja, nedostatka vrijednosti, kao i primjene tehnika poput jednoznačnog kodiranja koje je svojstveno radu sa kategorijskim podacima, kakvi su podaci o transakcijama, imaju veliki broj nedostajućih ili nultih vrijednosti. Ovaj algoritam interno rješava ovaj problem tako što ovi uzorci bivaju izostavljeni iz seta prilikom odabira potencijalnih pragova i računanja statistike. Međutim, prilikom odabira optimalnog praga, ovi uzorci se sprovode u lijevi ili desni potomak zavisno od toga koji vodi najvećoj redukciji gubitaka. Osim što omogućava klasifikaciju podataka sa nedostajućim vrijednostima, ovaj algoritam dobija na brzini i do pedeset puta u odnosu na naivne varijante koje ne uzimaju nedostatak vrijednosti u razmatranje [46].

Deset puta veća brzina izvršavanja na jednoj mašini u odnosu na ostala popularna rješenja, mogućnost brzog rada sa velikim setovima podataka [46], kao i mogućnost rada sa neskaliranim i nenormalizovanim podacima svrstavaju ovaj algoritam u prvi red za probleme detektovanja zloupotreba platnih kartica u realnom vremenu. Međutim, XGBoost iako veoma učinkovit u radu sa velikom količinom podataka, ne snalazi se u radu sa kategorijskim podacima, stoga je prije njegove upotrebe za detekciju zloupotreba potrebno izvršiti određeno pretprocesiranje seta podataka, koje dodatno povećava dimenzionalnost problema.

4.4.5 CatBoost

CatBoost je ansambl metoda otvorenog koda zasnovana na *boosting*-u, kreiran od strane istraživača iz kompanije Yandex. CatBoost koristi nesvjesna simetrična binarna stabla odlučivanja (eng. *oblivious symmetric trees*), koja koriste iste uslove podjele za sve čvorove jedne dubine. Upravo ovakva balansirana struktura je manje podložna preprilagođavanju podacima trening seta i predstavlja slabi klasifikator. Međutim, osim bolje generalizacije, glavna prednost ove strukture stabla jeste u brzini predikcije i maloj memorijskoj zahtjevnosti, čime se ubrzava predikcija klase novog uzorka [49].

Kreatori ovog algoritma ističu prednosti korišćenja ciljne statistike u konvertovanju kategorijskih karakteristika (sa više od dvije kategorije) u numeričke, u odnosu na korišćenje jednoznačnog kodiranja [49]. Ciljna statistika u slučaju binarne klasifikacije procjenjuje očekivanu vrijednost k-te karakteristike uzorka $\mathbf{x}_i$ u zavisnosti od frekvencije pojavljivanja kategorije $x_i^{(k)}$ u pozitivnoj klasi uzoraka, po formuli:

$$\hat{x}_i^{(k)} = \frac{\sum_{j=1}^{n} \mathbf{1}(x_j^{(k)} = x_i^{(k)}) \cdot y_j + ap}{\sum_{j=1}^{n} \mathbf{1}(x_j^{(k)} = x_i^{(k)}) + a} \quad i = 1, 2, \ldots, n, \tag{49}$$

gdje n predstavlja broj uzoraka trening seta, y_j ciljnu klasu j-tog uzorka, p predstavlja prior, a predstavlja njegovu težinu, dok $\mathbf{1}(x_j^{(k)} = x_i^{(k)})$ predstavlja indikatorsku funkciju koja ima vrijednost 1 ukoliko se radi o uzorcima iste kategorije k-te karakteristike, ili 0 u suprotnom. Prior ima ključnu ulogu kod kategorija sa malom frekvencijom pojavljivanja i u slučaju binarne klasifikacije najčešće predstavlja učestalost pojavljivanja pozitivne klase [49].

Računanje ciljne statistike na ovaj način predstavlja direktan uticaj ciljne klase trening uzorka na kodiranje njegovih kategorijskih vrijednosti. Ovo se smatra kršenjem dobre prakse po kojoj podaci van trening seta, koji nijesu dostupni tokom predikcije (kao što je klasa uzorka čija predikcija je cilj), ni na koji način ne bi trebalo da ulaze u model tokom treniranja jer to može dovesti do njegovih preoptimističnih

performansi na trening setu. Jedna od solucija koja se nameće jeste podjela seta na set podataka za računanje ciljne statistike i set uzoraka za treniranje modela. Međutim, u radu [50] se ističe da bi se na ovaj način izgubile korisne informacije koje izostavljeni uzorci (za računanje ciljne statistike) posjeduju.

Rješenje CatBoost pronalazi u primjeni posebnog principa uređenja (eng. *ordered principle*) po kome se trening podaci tretiraju kao da dolaze sekvencijalno jedan za drugim, po redosljedu slučajne permutacije σ. Sada se formula (49) svodi na:

$$\hat{x}_i^{(k)} = \frac{\sum_{x_j \in \mathcal{D}_k} \mathbf{1}(x_j^{(k)} = x_i^{(k)}) \cdot y_j + ap}{\sum_{x_j \in \mathcal{D}_k} \mathbf{1}(x_j^{(k)} = x_i^{(k)}) + a} \quad i = 1, 2, \ldots, n, \tag{50}$$

gdje $\mathcal{D}_k = \{\mathbf{x}_j : \sigma(j) < \sigma(i)\}$ predstavlja dio seta uzoraka koji prethodi trenutnom uzorku po pomenutoj permutaciji.

Ovaj princip uređenja, osim pri računanju ciljne statistike, koristi se i prilikom treniranja algoritma. Naime, prilikom treniranja kod GBDT-a i XGBoost-a se računaju reziduali nakon svake iteracije, koristeći vrijednosti klasa uzoraka koji su učestvovali u njegovom kreiranju, što takođe može dovesti do nepoželjne pristrasnosti modela trening setu. Stoga CatBoost, primjenjujući princip uređenosti, vrši estimaciju reziduala uzorka klasifikatorom koji je prilagođen rezidualima uzoraka koji su mu prethodili po permutaciji.

Nakon početne permutacije σ i kodiranja kategorijskih karakteristika, u svakoj iteraciji t kreira se n klasifikatora, za svaki uzorak po jedan. Izlazne vrijednosti klasifikatora $G_{t,i}$ koji će se koristiti za estimaciju reziduala uzorka $\mathbf{x}_i$ u iteraciji t se računaju na način opisan u Sekciji 4.4.3, pri čemu se u računanju vrijednosti listova koriste uzorci $\mathbf{x}_k$ koji prethode uzorku $\mathbf{x}_i$ u permutovanom setu. Na ovaj način se osigurava da se prilikom predikcije klase uzorka ne koristi ansambl kreiran na bazi toga uzorka, čime se teži izbjeći pristrasnost modela trening setu. Radi smanjivanja kompleksnosti ovog postupka često se ograničava broj klasifikatora na $\log_2 n$, čime se kvadratni problem prevodi u linearni [50]. Takođe, moguće je trenirati i praviti ansamble nad različitim podsetovima trening seta, kao što je to bio slučaj i kod ostalih algoritama koji se baziraju na stablu odlučivanja. Predikcija klase $\hat{y}'$ novog uzorka $\mathbf{x}'$ računa se na osnovu vrijednosti hipoteze krajnjeg ansambla $F_{T,n}$ po formuli (45).

Za razliku od XGBoost-a koji kao mjeru pogodnosti određenih karakteristika i pragova za grananje stabla koristi formulu (48), CatBoost koristi kosinusnu sličnost vektora reziduala $\boldsymbol{\varepsilon}_{F_{t-1}}$ prethodnog ansambla i vektora izlaznih vrijednosti $\boldsymbol{\gamma}$, koji sadrži hipoteze klasifikatora $G_{t,i}$ u iteraciji t za svaki uzorak $\mathbf{x}_i$ trening seta. Kosinusna sličnost koristi ugao između dva vektora kao mjeru sličnosti između njih i

Algoritam 2: Catboost - građenje modela

Input: $\{(x_i, y_i)\}_{i=1}^n \leftarrow$ trening set uređen na osnovu slučajne permutacije σ, $\eta \leftarrow$ hiperparametar redukcije, $T \leftarrow$ broj klasifikatora, $J \leftarrow$ broj čvorova regresionog stabla

$\{(x_i, y_i)\}_{i=1}^n \leftarrow$ kodiranje kategorijskih karakteristika;

$\gamma_{F_{0,i}} \leftarrow 0$ za $i = 1, 2, ..., n$;

for $t \leftarrow 1$ *to* T **do**

$h_{F_{t-1,i}}(\mathbf{x}_i) = \frac{e^{\gamma_{F_{t-1,i}}(\mathbf{x}_i)}}{1+e^{\gamma_{F_{t-1,i}}(\mathbf{x}_i)}}$ za $i = 1, 2, ..., n$;

$\varepsilon_{F_{t-1,i}} = y_i - h_{F_{t-1,i}}(\mathbf{x}_i)$ za $i = 1, 2, ..., n$;

$G_{t,i} \leftarrow$ prazno regresiono stablo za $i = 1, 2, ..., n$;

for *za svaki nivo stabla do ispunjenja kriterijuma zaustavljanja* **do**

$S = [\,] \leftarrow$ vektor kosinusnih sličnosti različitih struktura stabla $G_t^{(c)}$;

for *svaki potencijalni uslov grananja* c **do**

$G_{t,i}^{(c)} \leftarrow$ razgranaj stablo uslovom c za $i = 1, 2, ..., n$;

$\boldsymbol{\gamma} = [\,] \leftarrow$ vektor izlaznih vrijednosti klasifikatora za svaki uzorak;

for $i \leftarrow 1$ *to* n **do**

for $j \leftarrow 1$ *to* J **do**

$\gamma_{G_{t,i}^{(c)},j} = \frac{\sum_{k=1}^{i-1} \varepsilon_{F_{t-1,k}} \cdot \mathbf{1}(\mathbf{x}_k \in R_{G_{t,i}^{(c)},j})}{\sum_{k=1}^{i-1} h_{F_{t-1}}(\mathbf{x}_k) \cdot (1 - h_{F_{t-1}}(\mathbf{x}_k)) \cdot \mathbf{1}(\mathbf{x}_k \in R_{G_{t,i}^{(c)},j})}$;

if $\mathbf{x}_i \in R_{G_{t,i}^{(c)},j}$ **then**

$\gamma_i = \gamma_{G_{t,i}^{(c)},j}$

end

end

end

$S(c) \leftarrow cos(\boldsymbol{\varepsilon}_{F_{t-1}}, \boldsymbol{\gamma})$;

end

$c' \leftarrow \underset{c}{\text{argmax}}(S(c)) \leftarrow$ najbolji uslov grananja;

$G_{t,i} \leftarrow$ razgranaj stablo uslovom c' za $i = 1, 2, ..., n$;

end

$\gamma_{F_{t,i}}(\mathbf{x}_i) = \gamma_{F_{t-1,i}}(\mathbf{x}_i) + \eta \sum_{j=1}^{J} \gamma_{G_{t,i},j} \cdot \mathbf{1}(\mathbf{x}_i \in R_{G_{t,i},j}), \quad i = 1, 2, \ldots, n,$

end

ocjenjuje koliko je podjela na osnovu odabranog uslova doprinijela estimaciji reziduala i poboljšanju predikcije modela:

$$\cos(\angle(\boldsymbol{\varepsilon}_{F_{t-1}}, \boldsymbol{\gamma})) = \frac{\sum_{i=1}^{n} \varepsilon_{F_{t-1},i} \gamma_i}{\sqrt{\sum_{i=1}^{n} \varepsilon_{F_{t-1},i}^2} \sqrt{\sum_{i=1}^{n} \gamma_i^2}}, \tag{51}$$

gdje n predstavlja broj uzoraka trening seta.

Na ovaj način se pohlepno, za svaku dubina nesvjesnog simetričnog stabla odlučivanja, bira uslov podjele na nivou oba čvora te dubine koji daje najveću vrijednost kosinusne sličnosti. Ovo vodi kreiranju klasifikatora iste strukture, a različitih vrijednosti listova, što je uzrokovano različitim djelovima trening seta koje se koriste za njihovo računanje. Prilikom računanja kosinusne sličnosti često se izostavlja prvih nekoliko redova zbog velike varijanse koje mogu imati vrijednosti gradijenata kao i ciljna statistika [50]. Opisani postupak kreiranja ansambla kod CatBoost-a je dat pseudokodom 2.

Važna novina koju uvodi ovaj algoritam jeste kombinovanje kategorijskih karakteristika tokom procesa treniranja, čime se omogućava učenje kompleksnijih zavisnosti i šablona [50]. Dodatno, osnovni model bez podešavanja hiperparametara postiže bolje rezultate i veću brzinu pri izgradnji ansambala istih veličina u odnosu na XGBoost [49], što ga čini najbržom i najpoželjnijom ansambl metodom zasnovanom na stablu odlučivanja.

5 Metrike

Za ocjenu performansi predloženih metoda koristiće se standarne metrike binarne klasifikacije izvedene iz matrice konfuzije: tačnost, preciznost, odziv i F_1 mjera. Matrica konfuzije predstavlja tabelarni prikaz broja ispravno i pogrešno klasifikovanih transakcija u zavisnosti od klase kojoj pripadaju. U slučaju binarne klasifikacije klase se nazivaju pozitivnom i negativnom klasom. Kod detekcije zloupotrebe platnih kartica, pozitivnom klasom se označavaju zloupotrebe, dok regularne transakcije reprezentuju većinsku klasu označenu nulom.

Vrijednosti matrice date u Tabeli 1 predstavljaju:

- Stvarno pozitivni (eng. *true positive* – TP) - broj dobro klasifikovanih zloupotreba;
- Lažno negativni (eng. *false negative* - FN) - broj zloupotreba klasifikovanih kao regularne transakcije;
- Lažno pozitivni (eng. *false positive* - FP) - broj regularnih transakcija pogrešno klasifikovanih kao zloupotrebe;
- Stvarno negativni (eng. *true negative* - TN) - broj ispravno klasifikovanih regularnih transakcija;

Tabela 1: Matrica konfuzije

	Predikcija	
Stvarna klasa	Pozitivna klasa	Negativna klasa
Pozitivna	TP	FN
Negativna	FP	TN

Tačnost (t) predstavlja odnos ispravno klasifikovanih transakcija u odnosu na ukupan broj transakcija i data je formulom:

$$t = \frac{TP + TN}{TP + TN + FP + FN}. \tag{52}$$

U slučaju velike neizbalansiranosti seta podataka, gdje 0.1% ukupnog broja transakcija predstavljaju zloupotrebe, model koji bi sve transakcije klasifikovao kao regularne bi postizao tačnost od 99.9%. Ovo je razlog zbog koga tačnost za konkretan slučaj detekcije platnih kartica ne bi predstavljala dobru metriku za mjerenje performansi.

Preciznost (eng. *precision* - p) predstavlja broj dobro klasifikovanih zloupotreba u odnosu na ukupan broj transakcija klasifikovanih kao zloupotreba i data je formulom:

$$p = \frac{TP}{TP + FP}. \tag{53}$$

Odziv (eng. *recall/sensitivity* - r) predstavlja odnos broja dobro klasifikovanih zloupotreba i ukupnog broja zloupotreba i dat je formulom:

$$r = \frac{TP}{TP + FN}. \tag{54}$$

Povećavanje broja detektovanih zloupotreba obično negativno utiče na preciznost modela, jer dolazi do porasta broja lažnih alarma - odnosno pogrešno klasifikovanih regularnih transakcija. Teško je postići zadovoljavajuće vrijednosti i za preciznost i odziv, jer poboljšanje jednog parametra obično vodi do smanjenja drugog. Stoga je potrebna metrika koja će predstavljati harmonijsku sredinu preciznosti i odziva, odnosno F_1 mjera, data formulom:

$$F_1 = 2 \cdot \frac{p \cdot r}{p + r} \tag{55}$$

6 Eksperimentalna postavka i rezultati

Za detekciju zloupotreba biće testirano 8 algoritama klasifikacije nadgledanog mašinskog učenja: LR, KNN, DT, RF, AdaBoost, GBDT, XGBoost i CatBoost. U svrhu implementacije algoritama koristiće se gotove klase iz aktuelnih javno dostupnih biblioteka, datih u Tabeli 2.

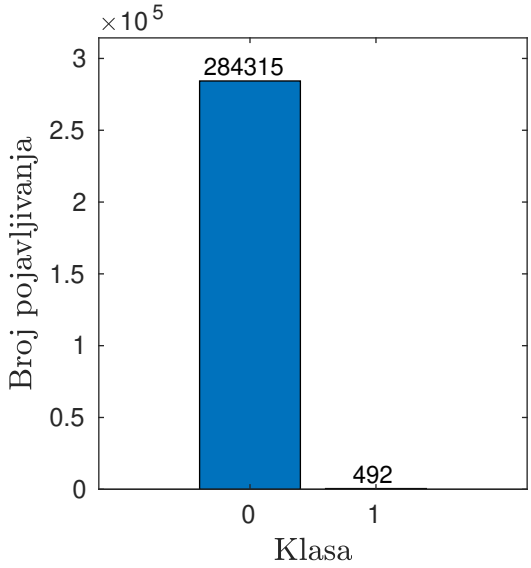
Algoritam	**Biblioteka**	**Klasa**
LR	sklearn.linear_model	LogisticRegression
KNN	sklearn.neighbors	KNeighborsClassifier
DT	sklearn.tree	DecisionTreeClassifier
RF	sklearn.ensemble	RandomForestClassifier
AdaBoost	sklearn.ensemble	AdaBoostClassifier
GBDT	sklearn.ensemble	GradientBoostingClassifier
XGBoost	xgboost	XGBClassifier
CatBoost	catboost	CatBoostClassifier

Tabela 2: Klase iz odgovarajućih biblioteka koje su korišćene za implementaciju korišćenih algoritama klasifikacije.

Pomenuti algoritmi će se trenirati i testirati na setu podataka [10], koji sadrži transakcije od strane Evropskih korisnika u septembru 2013. godine. Ovaj označeni set podataka sadrži 284, 807 uzoraka koje predstavljaju transakcije opisane sa 30 karakteristika i binarnom karakteristikom *Class*. Karakteristika *Class* predstavlja oznaku da li je transakcija bila primjer zloupotrebe platne kartice (vrijednost 1) ili regularna transakcija (vrijednost 0). Radi zaštite povjerljivosti podataka nijesu dostupni sirovi podaci. Naime, od 30 karakteristika, njih 28 ($V_1, V_2, ..., V_{28}$) predstavljaju glavne komponente dobijene primjenom PCA transformacije, stoga su ovi podaci dati u numeričkom obliku i nemaju fizički smisao. Jedine dvije karakteristike nad kojima nije primijenjena PCA su *Time* i *Amount*. *Time* predstavlja broj sekundi koje su prošle od prve transakcije do date transakcije, dok *Amount* predstavlja njen iznos. Ovaj set podataka je odabran zbog njegove velike primjene u brojnim radovima koji su se bavili detekcijom zloupotrebe platnih kartica [3–5, 7, 9, 12–14, 29, 30, 32, 33, 51], kao i činjenicom da predstavlja rijetko dostupan set realnih, a ne simuliranih transakcija.

Odnos broja uzoraka obje klase, koji je predstavljen na slici 17 jasno svjedoči da se radi o jako neizbalansiranom setu podataka. Broj uzoraka većinske klase iznosi 284, 315, dok je svega 492 primjera zloupotrebe platnih kartica, što čini svega 0.172%

cjelokupnog seta podataka. Stoga će se koristiti različite tehnike za balansiranje seta u cilju popravljanja performansi algoritama.



Slika 17: Odnos broja uzoraka dvije klase koje predstavljaju regularne transakcije (klasa 0) i zloupotrebe platnih kartica (klasa 1).

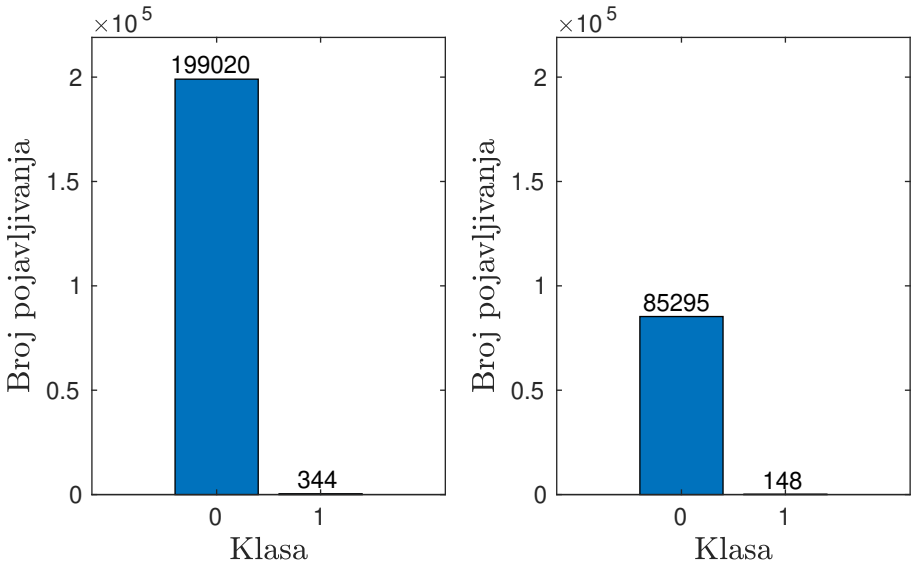
Uvidom u strukturu seta podataka, pokazuje se da u njemu nema nedostajućih vrijednosti, niti kategorijskih karakteristika, što uveliko olakšava proces pretprocesiranja i potvrđuje činjenicu da je već pretprocesiran. Sa obzirom da 28 karakteristika predstavljaju glavne komponente dobijene PCA transformacijom, ove karakteristike se nalaze u standardizovanoj formi stoga nije potrebno njihovo dodatno skaliranje.

Dvije preostale karakteristike *Time* i *Amount* nijesu standardizovane i potencijalno sadrže *outlier*-e, stoga je neophodno istražiti najbolji metod za njihovo uklanjanje ili eventualno skaliranje karakteristika. Kako bi se provjerilo da li karakteristike slijede normalnu distribuciju, koristi se Kolmogorov-Smirnov (KS) test [52]. KS test je neparametarski test koji ispituje hipotezu da uzorci karakteristike podliježu referentnoj distribuciji. Testiranjem svih karakteristika, izuzev ciljne karakteristike *Class*, KS test pokazuje da karakteristike ne podliježu normalnoj distribuciji, što upućuje na korišćenje IQR-a u detekciji *outlier*-a (Sekcija 3.1).

U radu [22] predlaže se upotreba IQR-a sa vrijednošću $C = 1.5$ za detekciju *outlier*-a i Winsorizing metodu za modifikovanje njihove vrijednosti (Sekcija 3.1). Međutim, ovaj pristup bi doveo do modifikovanja preko 13% uzoraka manjinske klase na osnovu vrijednosti karakteristike *Amount*. Kako set podataka sadrži svega 492 uzorka koji predstavljaju zloupotrebu veoma je važno da se sačuva njihova specifičnost, koja potencijalno nosi veoma bitnu informaciju. Sa obzirom da ekstremne vrijednosti iznosa transakcije (*Amount*) mogu upućivati na zloupotrebu i samim tim

biti glavni indikator u otkrivanju prevara, neće se vršiti modifikacija ovih vrijednosti, već će se obaviti njihova robustna standardizacija primjenom klase *RobustScaler* iz biblioteke *sklearn.preprocessing* koji je otporan na negativan uticaj *outlier*-a (Sekcija 3.3).

Prije robustne standardizacije neophodno je podijeliti set podataka na trening set i test set, u procentualnom odnosu 70%-30% [12, 29] u odnosu na broj uzoraka u cjelokupnom setu podataka. Kako je u originalnom setu podataka prisutno svega 0.172% uzoraka zloupotrebe, podjela seta na osnovu slučajne permutacije bi vrlo vjerovatno vodila testnom setu u kome uopšte nijesu prisutni uzorci manjinske klase. Evaluacija modela na ovakvom testnom setu ne bi dala uvid u njegovu stvarnu mogućnost detekcije zloupotreba, stoga je neophodno u trening i testnom setu zadržati odnos klasa iz originalnog seta podataka. Ovo se postiže korišćenjem stratifikacije (eng. *stratify*), čime se osigurava testiranje modela na uzorcima koji su reprezenti stvarne distribucije podataka. Broj uzoraka obje klase u setu za treniranje i testiranje je prikazan na slici 18, pri čemu se može vidjeti da zloupotrebe čine 0.172% oba seta. Radi reproduktivnosti rezultata, podjela na trening i test set je obavljena fiksiranjem nasumične inicijalizacije (eng. *random state*) na vrijednost 42.



Slika 18: Odnos broja uzoraka dvije klase koje predstavljaju regularne transakcije (klasa 0) i zloupotrebe platnih kartica (klasa 1) u trening setu (slika lijevo) i testnom setu (slika desno) nakon podjele.

Svaki od pomenutih algoritama ima hiperparametre koji kontrolišu strukturu modela i proces njihovog treniranja, čija vrijednost se bira u procesu pretraživanja po mreži sa unakrsnom validacijom (eng. *GridSearchCV* - GSCV). Naime, za svaki od algoritama unaprijed se zadaju različite vrijednosti izabranih hiperparametara

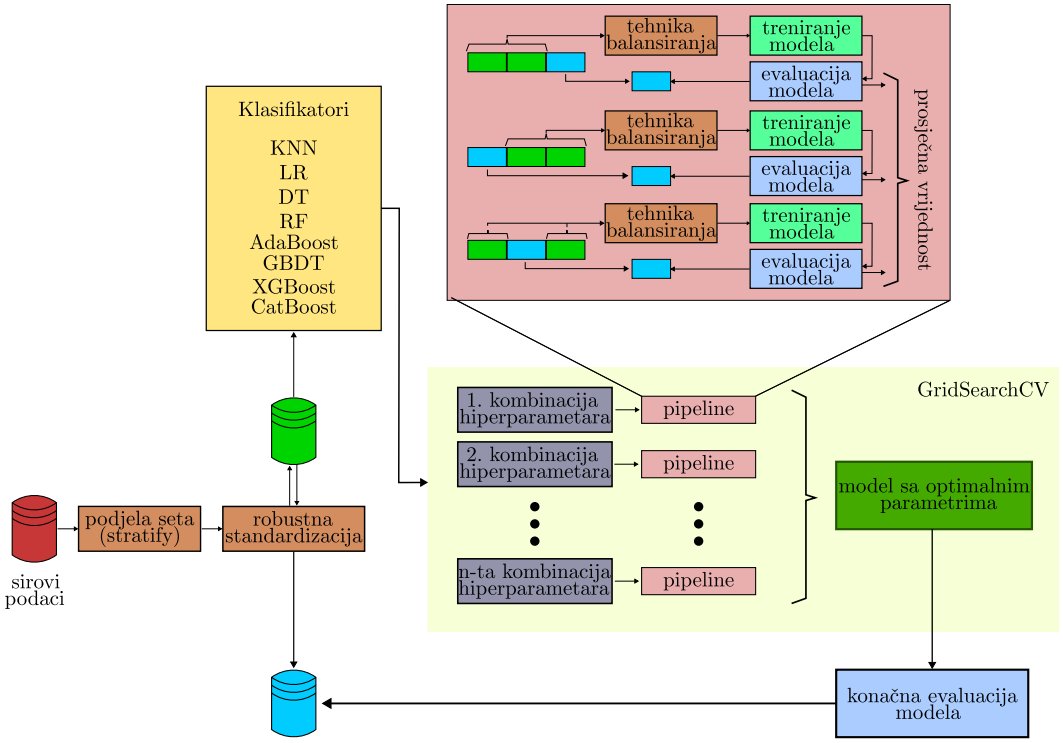
nakon čega se u procesu K-slojne unakrsne validacije (Sekcija 2), uz mogućnost definisanja željene metrike za evaluaciju performansi (u ovom slučaju F_1 mjere), bira njihova najbolja kombinacija. U cilju ubrzavanja procesa treniranja koristi se manji broj mogućih vrijednosti hiperparametara, koje su birane na osnovu dostupne literature ili eksperimentalno.

Osim različitih hiperparametara, biće testirani i uticaji različitih tehnika balansiranja na performanse klasifikatora. Tehnike balansiranja koje će biti korišćene su: ROS, BSMOTE, Tomek, RUS, kao i hibridna tehnika kombinovanja SMOTE-a i ENN-a (SMENN) koji su opisani u Sekciji 3.4. Osim različitih tehnika balansiranja u radovima [4],[7] se ističe važnost stepena balansiranja, odnosno krajnjeg odnosa broja uzoraka manjinske i većinske klase nakon skaliranja, stoga će se testirati odnosi 50-50 i 10-90 respektivno.

Balansiranje trening seta prije procesa K-slojne unakrsne validacije, bi u slučaju primjene tehnika preodabiranja (ROS, BSMOTE), kao i hibridne tehnike SMENN, vodilo ka postojanju uzoraka u validacionom setu koji su generisani na osnovu uzoraka iz preostalog dijela trening seta koji služi za treniranje modela. Na ovaj način evaluacija na validacionom setu ne bi pokazala pravu sposobnost generalizacije modela usljed curenja podataka iz trening seta, stoga je neophodno balansirati samo dio seta koji se koristi za trening u procesu unakrsne validacije. Ovo se postiže korišćenjem sekvenci koraka (eng. *pipeline*) koji podrazumijevaju transformaciju podataka i treniranje modela, izvršavajući se kao jedan proces, što je značajno u ekperimentisanju jer omogućava dosljednu primjenu tehnika i metoda nad svih podacima.

Sekvence koraka su prikazane u okviru ilustracije toka procesa obuke i evaluacije modela na slici 19, gdje je istaknuto kako u procesu K-slojne unakrsne validacije dolazi do primjene tehnika balansiranja nad podsetovima namijenjenim za trening algoritama mašinskog učenja sa odabranim hiperparametrima. U ovom postupku nije došlo do transformacije podseta koji će služiti za validaciju modela, čime su se stekli uslovi za njegovu nepristrasnu evaluaciju. Kako je opisano u Sekciji 2, konačna evaluacija modela dobija se kao prosjek dobijenih evaluacija svake podjele. U procesu unakrsne validacije odabrano je da broj podjela bude 3 kako bi zadržali veći broj zloupotreba u validacionom setu i na taj način obezbijedili pouzdaniju ocjenu performansi modela.

Za svaku tehniku balansiranja i odnos broja uzoraka dvije klase nakon skaliranja, dobija se model sa onim hiperparametrima koji su u procesu unakrsne validacije dali najbolje rezultate. Testiranjem ovih modela na prethodno izdvojenom testnom setu će pokazati koji model postiže najveću vrijednost F_1 mjere, te kako struktura trening seta utiče na performanse algoritama.



Slika 19: Tok procesa obuke i evaluacije modela.

6.0.1 LR rezultati

Osnovni model LR koristi algoritam zasnovan na kvazi-Newtonovoj metodi koji omogućava efikasno prilagođavanje modela velikim skupovima podataka, balansirajući brzinu i tačnost prilikom optimizacije hiperparametara modela. Maksimalan broj iteracija ovog modela je 100, dok regularizacioni hiperparametar ima vrijednost $\lambda = 1$. Algoritam se zaustavlja prije dostizanja maksimalnog broja iteracija ukoliko ne dođe do promjene funkcije gubitaka veće od podrazumijevane vrijednosti tolerancije (0.0001), što signalizira da je došlo do konvergencije te da dalje iteracije neće doprinijeti značajno performansama modela.

Eksperimentalno je pokazano da promjena maksimalnog broja iteracija ne vodi značajnoj promjeni performansi modela, već da one najviše zavise od regularizacionog hiperparametra. Ovo je primarno uslovljeno pomenutim kriterijumom zaustavljanja, stoga su korišćenjem GSCV-a testirane performanse ovog algoritma sa kombinacijom različitih vrijednosti regularizacionog hiperparametara $\lambda \in \{0.01, 0.1, 1, 10, 100, 1000, 10000\}$.

Na osnovu rezultata iz Tabele 3 najbolje rezultate postiže model sa $\lambda = 1000$ uz primjenu tehnike RUS 10-90. Ovaj rezultat $F_1 = 0.7645$ nadmašuje rezultate logističke regresije u radovima [9, 14], čime se ističe važnost optimizacije hiperpara-

metara i pažljivog odabira tehnike balansiranja seta podataka.

Zanimljivo je da model sa optimizovanim hiperparametrom na neizbalansiranom setu postiže nešto lošiju vrijednost F_1 mjere u odnosu na osnovni model. Uvidom u prosječne vrijednosti F_1 mjere na testnim setovima u procesu troslojne unakrsne validacije, model sa $\lambda = 10$ je za 0.006 postigao bolji prosječni rezultat od modela sa podrazumijevanim hiperparametrima i odabran je kao optimalan. Međutim, konačnom evaluacijom na testnom setu se pokazalo da odabrani model postiže za nijansu lošiji rezultat. Dodatnim ispitivanjem u 100 iteracija pri čemu su korišćeni različiti slučajno odabrani setovi podataka za treniranje i testiranje, pokazuje se da nema velikih razlika u performansama ova dva modela, što upućuje da je ovo uzrokovano od strane podjele podataka tokom procesa GSCV-a.

RUS i Tomek tehnika pokazuju da uklanjanje uzoraka većinske klase doprinosi jasnijem diferenciranju dvije klase. Iako se primjenom tehnika preodabiranja takođe vrši balansiranje seta podataka, njihovom primjenom se ne postižu zadovoljavajući rezultati. Dodatno, optimalna vrijednost regularizacionog hiperparametra mnogo zavisi od primijenjene tehnike balansiranja, što je razlog razmatranja ovako širokog opsega vrijednosti ovog hiperparametra. Takođe, važno je istaći da konačni odnos broja uzoraka obje klase utiče na perfomanse, te da odnos 10-90 pokazuje bolje rezultate poredeći sa 50-50.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
osnovni model	0.9992	0.8611	0.6284	0.7266
optimizovani model	0.9991	0.8571	0.6081	0.7115
ROS 50-50	0.9841	0.0874	0.8649	0.1588
ROS 10-90	0.9988	0.6277	0.7973	0.7024
BSMOTE 50-50	0.9921	0.1599	0.8311	0.2683
BSMOTE 10-90	0.9958	0.2607	0.7838	0.3912
RUS 50-50	0.9991	0.7355	0.7703	0.7525
RUS 10-90	0.9992	0.7724	0.7568	**0.7645**
Tomek	0.9992	0.8624	0.6351	0.7315
SMENN 50-50	0.9842	0.0870	0.8581	0.1581
SMENN 10-90	0.9989	0.6413	0.7973	0.7108

Tabela 3: Rezultati evaluacije modela LR-a sa različitim tehnikama balansiranja podataka.

6.0.2 KNN rezultati

Osnovni model koristi $K = 5$ najbližih susjeda koji se određuju korišćenjem Euklidske distance od uzorka čija klasifikacija je cilj. Na osnovu rezultata osnovnog modela prikazanog u Tabeli 4 može se zaključiti da osnovni model postiže zadovoljavajuću vrijednost F_1 mjere koja iznosi 0.8075.

Kako bi se optimizovale performanse algoritma, u procesu GSCV-a testirale su se različite neparne vrijednosti broja najbližih susjeda $K \in \{1, 3, 5, 7\}$, kao i različiti uticaj najbližih susjeda na proces klasifikacije zavisno od njihove udaljenosti od posmatranog uzorka [36]. Na osnovu dobijenih rezultata prikazanih u Tabeli 4 najbolje se performanse postižu za $K = 3$ na neizbalansiranom setu podataka, pri čemu oni uzorci koji su bliže u prostoru imaju veći uticaj na donošenje odluke o pripadnosti klasi. Maksimalna vrijednost F_1 mjere iznosi 0.8401, pri čemu se ističe veoma visoka vrijednost preciznosti, odnosno veoma je mali broj regularnih transakcija koje model pogrešno klasifikuje kao zloupotrebe, kao u radu [7]. Ovakav rezultat gdje dolazi do detektovanja preko 76% zloupotreba, uz mali broj posmatranih susjeda koji su važniji za donošenje odluke ukoliko su bliži u prostoru, sugeriše da su većina uzoraka koji predstavljaju zloupotrebe međusobno slični.

Zavidni rezultati odziva se postižu primjenom tehnika preodabiranja ROS i BSMOTE, koji nadmašuju optimalni model na neizbalansiranom setu podataka. Međutim, bolja stopa detekcije zloupotreba dolazi na račun preciznosti detekcije i veće pojave lažnih alarma. Za razliku od pomenutih tehnika, od ostalih tehnika balansiranja jedino su Tomekove veze uspjele poboljšati performanse osnovnog modela na neizbalansiranom setu. Kada je u trening setu prisutan izuzetno mali broj zloupotreba algoritam teži sužavanju posmatrane okoline, jer bi njeno širenje dovelo do preovladavanja uzoraka većinske klase zbog njihove brojnosti. Na osnovu rezultata unakrsne validacije može se zaključiti da se uklanjanjem uzoraka većinske klase (RUS) širi posmatrana okolina, odnosno povećava optimalan broj susjeda koji učestvuju u procesu klasifikacije. Međutim, ovi rezultati iako postižu bolju detekciju zloupotreba izazivaju pojavu velikog broja lažnih alarma. Ovo ukazuje da uklanjanje odbiraka većinske klase neće doprinijeti poboljšanju perfomansi ovog algoritma sa aspekta F_1 mjere.

6.0.3 DT rezultati

Osnovni model stabla odlučivanja povećava dubinu stabla sve dok se u svakom listu ne nađu minimalno 2 uzorka. Ovako slab kriterijum zaustavljanja pravi kompleksna stabla odlučivanja dubine 20 za konkretan trening set.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
osnovni model	0.9994	0.9145	0.7230	0.8075
optimizovani model	0.9995	0.9339	0.7635	**0.8401**
ROS 50-50	0.9995	0.8992	0.7838	0.8375
ROS 10-90	0.9995	0.8992	0.7838	0.8375
BSMOTE 50-50	0.9994	0.8380	0.8041	0.8207
BSMOTE 10-90	0.9994	0.8380	0.8041	0.8207
RUS 50-50	0.9824	0.0793	0.8649	0.1452
RUS 10-90	0.9979	0.4377	0.8311	0.5734
Tomek	0.9995	0.9113	0.7635	0.8309
SMENN 50-50	0.9987	0.5913	0.8311	0.6910
SMENN 10-90	0.9987	0.6000	0.8311	0.6969

Tabela 4: Rezultati evaluacije modela KNN-a sa različitim tehnikama balansiranja podataka.

U cilju popravljanja performansi osnovnog algoritma, u procesu unakrsne validacije primjenom GSCV-a testirane su različite vrijednosti maksimalne dubine drveta $\{1, 3, 5, 7, 10, 13, 16, 20, 40, 50\}$. Cilj je ograničiti dubinu stabla radi izbjegavanja preprilagođenosti algoritma trening podacima, pri čemu vrijednosti obuhvataju širok spektar uz njihovu veću koncetraciju na kreiranje modela manje kompleksnosti od osnovnog. Velike maksimalne dubine stabla odlučivanja 40 i 50, iako nedostižne osnovnom modelu na neizbalansiranom setu podataka, imaju značaj pri korišćenju određenih tehnika balansiranja, stoga će biti razmatrane.

Rezultati testiranja modela su dati u Tabeli 5, gdje najbolje rezultate $F_1 = 0.8045$ postiže stablo odlučivanja sa maksimalnom dubinom 5 uz primjenu tehnike pododabiranja Tomekove veze. Osim tehnike Tomekovih veza, ostale tehnike balansiranja seta podataka kvare vrijednost F_1 mjere, čime se ističe mogućnost rada ovog algoritma sa neizbalansiranim podacima. Kao i u slučaju KNN-a, primjenom slučajnog pododabiranja iako raste stopa detekcije zloupotreba to je postignuto uz veliki broj pogrešno klasifikovanih regularnih transakcija. Poređenjem rezultata osnovnog i optimizovanog modela zaključuje se da je osnovni model bio previše kompleksan, te da je smanjivanje dubine zaista dovelo do bolje generalizacije na testnom setu.

Optimalna dubina varira i zavisi od odabrane tehnike balansiranja. Nakon odabrane tehnike na osnovu trening i testne greške u procesu unakrsne validacije, može se vršiti dodatna optimizacija pažljivijim odabirom maksimalne dubine iz užeg opsega vrijednosti čime se potencijalno mogu dodatno poboljšati performanse.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
osnovni model	0.9991	0.7707	0.7027	0.7350
optimizovani model	0.9994	0.8852	0.7297	0.8000
ROS 50-50	0.9992	0.7923	0.6959	0.7410
ROS 10-90	0.9988	0.6263	0.8041	0.7041
BSMOTE 50-50	0.9990	0.7230	0.7230	0.7230
BSMOTE 10-90	0.9988	0.6303	0.7027	0.6645
RUS 50-50	0.9816	0.0725	0.8176	0.1332
RUS 10-90	0.9900	0.1267	0.8108	0.2192
Tomek	0.9994	0.9068	0.7230	**0.8045**
SMENN 50-50	0.9976	0.4014	0.7838	0.5309
SMENN 10-90	0.9983	0.4979	0.7973	0.6130

Tabela 5: Rezultati evaluacije modela DT-a sa različitim tehnikama balansiranja podataka.

6.0.4 RF rezultati

Rezultati osnovnog modela bez podešavanja hiperparametara, dati u Tabeli 6, prevazilaze do sada prikazane rezultate algoritama sa optimizovanim hiperparametrima. Model koristi 100 stabala odlučivanja kao slabe klasifikatore, pri čemu za kreiranje svakog od njih koristi set podataka dobijen *bootstrap* metodom. Dodatno, na nivou svakog čvora traži se najbolji uslov podjele na osnovu 5 slučajno odabranih karakteristika, ukoliko u čvoru postoji minimalno dva uzorka.

Uvidom u performanse osnovnog modela na testnom setu, zaključuje se da model ima problem velike varijanse. Model se previše prilagođava podacima stoga je u cilju bolje generalizacije neophodno ograničiti grananje drveta osnovnog klasifikatora (stabla odlučivanja) i testirati različiti broj karakteristika koje se uzimaju u obzir na nivou čvora. Prilikom izgradnje svakog stabla sa obzirom da se u trening setu nalazi svega 0.172% zloupotreba, korišćenjem samo dijela seta za treniranje modela rizikuje se njihovo izostavljanje. Ovo je razlog što se zadržava korišćenje trening seta iste veličine, pri čemu će varijacije među stablima, osim korišćenja različitog broja karakteristika, biti uzrokovane varijacima u trening setovima usljed primjene *bootstrap* odabiranja. Ovaj slučajni odabir uzoraka sa ponavljanjem, kao i slučajnost pri razmatranju karakteristika u određivanju uslova podjele [39] će obezbijediti varijacije u strukturama stabala odlučivanja.

U ovu svrhu tokom procesa GSCV-a testirane su kombinacije minimalnog broja uzoraka za razdvajanje na nivou čvora $\{2, 3, 4, 5\}$, čime će se smanjiti kompleksnost

nezavisnih klasifikatora. Dodatno, na nivou svakog čvora razmatraju se ili sve karakteristike, ili drugi korijen njihovog broja (odnosno vrijednost 5 za konkretan set podataka).

Iako se uz korišćenje svih tehnika balansiranja osim RUS-a postižu dobri rezultati, najveću vrijednost F_1 mjere model postiže na neizbalansiranom setu podataka. Ovaj model razmatra sve karakteristike za odabir podjele na nivou čvora, ukoliko se u njemu nalaze minimalno 4 uzorka. Kao i u radu [41], ovdje se potvrđuje sposobnost postizanja dobrih rezultata RF-a na neizbalansiranom setu podataka. Važno je napomenuti da se primjenom ROS-a takođe popravljaju rezultati, dok se primjenom hibridne tehnike SMENN 10-90 postiže veoma visoka vrijednost odziva.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
osnovni model	0.9995	0.9656	0.7568	0.8485
optimizovani model	0.9996	0.9435	0.7905	**0.8603**
ROS 50-50	0.9995	0.9573	0.7568	0.8453
ROS 10-90	0.9996	0.9583	0.7770	0.8582
BSMOTE 50-50	0.9995	0.9344	0.7703	0.8444
BSMOTE 10-90	0.9995	0.9268	0.7703	0.8413
RUS 50-50	0.9851	0.0924	0.8581	0.1668
RUS 10-90	0.9985	0.5391	0.8378	0.6561
Tomek	0.9995	0.9492	0.7568	0.8421
SMENN 50-50	0.9993	0.8026	0.8243	0.8133
SMENN 10-90	0.9994	0.8105	0.8378	0.8239

Tabela 6: Rezultati evaluacije modela RF-a sa različitim tehnikama balansiranja podataka.

Model sa optimalnim hiperparametrima bez primjene tehnika balansiranja postiže bolje rezultate u odnosu na osnovni model u radu [3]. Iako u ovom radu RF postiže bolje rezultate nakon primjene tehnika balansiranja, autor primjenjuje tehnike nad čitavim setom podataka prije primjene algoritama, nakon čega evaluira samo na osnovu K-slojne unakrsne validacije, bez odvojenog testnog seta. Na ovaj način tokom korišćenja tehnika balansiranja seta, pogotovo tehnika predodabiranja, dolazi do curenja podataka iz dijela seta za trening u set za evaluaciju. Upravo se ovdje ističe prednost predloženog modela, koji korišćenjem *pipeline*-a ne primjenjuje tehnike balansiranja nad testnim setom, kao ni na evaluacionom setu u procesu unakrsne validacije.

Rezultat dobijen ovim modelom prevazilazi rezultate RF u radu [14]. U ovom

radu RF postiže znatno bolje rezultate u odnosu na neuralnu mrežu, GBDT i ansambla modela složenih u slojeve (eng. *stacked ensemble*), pri čemu se neuralna mreža pokazala kao najlošiji izbor klasifikatora.

6.0.5 Adaboost rezultati

Osnovni model koristi maksimalno $T = 50$ stabala odlučivanja maksimalne dubine 1 kao osnovnih klasifikatora, sa korakom učenja $\eta = 1$. Međutim, u klasi je implementirana mogućnost zaustavljanja dodavanja klasifikatora ukoliko se model u potpunosti prilagodio trening setu, što se nije desilo nad korišćenim setom podataka. Ovo ukazuje na potrebu povećavanja kompleksnosti modela u cilju smanjivanja problema velikog *bias*-a.

Uz primjenu tehnika za balansiranje podataka, metodom GSCV-a razmatrane su vrijednosti broja iteracija $T \in \{50, 100\}$, koraka učenja $\eta \in \{0.5, 1\}$ i stabala odlučivanja maksimalne dubine $\{1, 2, 3\}$. Pokazuje se da nezavisno od povećanja broja iteracija, stablo odlučivanja dubine 1 nije u stanju da se prilagodi podacima. Najveća vrijednost $\mathrm{F}_1 = 0.8222$ je postignuta uz primjenu stabla odlučivanja maksimalne dubine 3, $\eta = 0.5$ i $T = 100$, uz primjenu Tomekovih veza. Ovdje se takođe ogleda prednost AdaBoost-a u odnosu na stablo odlučivanja, pri čemu se uz primjenu iste tehnike balansiranja primjenom AdaBoost-a popravljaju rezultati.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
osnovni model	0.9992	0.7868	0.7230	0.7535
optimizovani model	0.9994	0.9364	0.6959	0.7984
ROS 50-50	0.9993	0.8560	0.7230	0.7839
ROS 10-90	0.9994	0.9469	0.7230	0.8199
BSMOTE 50-50	0.9994	0.8626	0.7635	0.8100
BSMOTE 10-90	0.9994	0.8983	0.7162	0.7970
RUS 50-50	0.9770	0.0618	0.8649	0.1154
RUS 10-90	0.9975	0.3994	0.8446	0.5423
Tomek	0.9994	0.9098	0.7500	**0.8222**
SMENN 50-50	0.9988	0.6304	0.7838	0.6988
SMENN 10-90	0.9991	0.7091	0.7905	0.7476

Tabela 7: Rezultati evaluacije modela AdaBoost-a sa različitim tehnikama balansiranja podataka.

6.0.6 GBDT rezultati

Uvidom u rezultate osnovnog modela koji koristi 100 regresionih stabala maksimalne dubine 3, predstavljene u Tabeli 8, može se primijetiti veoma niska vrijednost F_1 mjere. Ovo je primarno uzrokovano veoma slabom detekcijom uzoraka koji predstavljaju zloupotrebe (veoma niska vrijednost odziva). Odabir funkcije gubitaka koja nastoji da popravi tačnost modela, u slučaju velike neizbalansiranosti ne omogućava algoritmu da na pravi način detektuje zloupotrebe koje predstavljaju zanemarljiv procenat seta podataka.

Loše performanse osnovnog modela i nemogućnost preprilagođavanja trening podacima su bili indikator da je neophodno koristiti kompleksnije klasifikatore. Stoga su u procesu GSCV-a korišćene dubine $\{6, 7, 8\}$, pri čemu se težilo popravljanju generalizacije modela zaustavljanjem grananja stabla ukoliko se u čvoru ne nalazi minimalno 3 ili 4 uzorka. Ove vrijednosti su odabrane eksperimentalno, sužavanjem opsega njihovih vrijednosti. Dodatno, u radu [47] se predlaže učenje modela na različitim podsetovima podataka, stoga se u unakrsnoj validaciji razmatralo 80% i 100% karakteristika za podjelu na nivou čvora. Korak učenja je po Friedmanu odabran da bude 0.1 [45], što je njegova podrazumijevana vrijednost za korišćenu klasu.

Najbolje performanse, na osnovu rezultata predstavljenih u Tabeli 8, se postižu sa modelom koji u svakoj iteraciji koristi regresiona stabla dubine 8, pri čemu se na nivou čvora razmatra 80% karakteristika za podjelu ukoliko se u njemu nalazi minimalno 3 uzorka. Ovaj model je treniran na setu podataka nad kojim je primijenjena tehnika ROS 10-90. Osim primjene RUS tehnike, rezultati ne variraju previše u zavisnosti od izabrane tehnike balansiranja. Međutim, povećavanje kompleksnosti modela uz dobar odabir hiperparametara u procesu unakrsne validacije je doprinio znatnom poboljšanju performansi u odnosu na osnovni model, iako ne predstavlja znatno poboljšanje u poređenju sa DT-om.

6.0.7 XGBoost rezultati

Osnovni model, čiji su rezultati predstavljeni u Tabeli 9 ima veoma visoku preciznost, međutim optimizacijom hiperparametara težiće se popravljanju procenta detektovanih zloupotreba kako bi se finalna vrijednost F_1 mjere popravila. Podrazumijevani broj osnovnih klasifikatora je 100, korak učenja $\eta = 0.3$, dok je vrijednost regularizacionog hiperparametra $\lambda = 1$. U svakoj iteraciji kreira se stablo odlučivanja maksimalne dubine 6 koje koristi sve karakteristike prilikom svoje konstrukcije. Minimalna težina djeteta koja utiče na stepen obrezivanja stabla ima podrazumijevanu vrijednost 1.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
osnovni model	0.9984	0.7273	0.1622	0.2652
optimizovani model	0.9992	0.7956	0.7365	0.7650
ROS 50-50	0.9992	0.8074	0.7365	0.7703
ROS 10-90	0.9994	0.9146	0.7230	**0.8075**
BSMOTE 50-50	0.9988	0.6141	0.7635	0.6807
BSMOTE 10-90	0.9992	0.7647	0.7905	0.7774
RUS 50-50	0.9636	0.0406	0.8851	0.0777
RUS 10-90	0.9961	0.2847	0.8176	0.4223
Tomek	0.9992	0.7910	0.7162	0.7518
SMENN 50-50	0.9988	0.6142	0.8176	0.7014
SMENN 10-90	0.9991	0.7041	0.8041	0.7508

Tabela 8: Rezultati evaluacije modela GBDT-a sa različitim tehnikama balansiranja podataka.

U procesu GSCV-a koristiće se model sa korakom učenja $\eta = 0.1$ kao i kod GBDT-a. Smanjivanjem koraka učenja teži se manjem uticaju pojedinačnih klasifikatora na konačnu odluku, stoga će se razmatrati veći broj iteracija $T \in \{100, 150, 200, 250, 300\}$. Porastom broja klasifikatora raste vjerovatnoća prevelikog prilagođavanja trening podacima, stoga će se testirati različite vrijednosti hiperparametara: minimalne težine djeteta $\{1, 2, 3, 4\}$, vrijednosti regularizacionog hiperparametra $\lambda \in \{0, 0.25, 0.5, 0.75, 1, 1.25\}$ i procenta broja karakteristika koje će se koristiti za izgradnju svakog stabla $\{90\%, 100\%\}$.

Najbolje rezultate postiže model koji koristi 200 osnovnih klasifikatora, koji na nivou svakog stabla koristi 90% karakteristika sa hiperparametrima $\lambda = 1.25$ i minimalne težine djeteta 2. Ovaj model se trenira na preodabranom setu primjenom tehnike ROS 10-90. Primjenom tehnika preodabiranja BSMOTE i ROS postižu se bolji rezultati u odnosu na tehnike koje se primjenjuju u radu [4]. U ovom radu se takođe pokazuje da primjena tehnike pododabiranja RUS daje veoma loše rezultate i nije preporučljiva za konkretan problem. Kao i kod RF-a, ukoliko se teži većoj detekciji zloupotreba na račun preciznosti, dobre performanse pokazuje model uz primjenu hibridne tehnike SMENN 50-50. Ovaj model detektuje preko 82% ukupnog broja zloupotreba, pri čemu u skoro 30% slučajeva pogrešno proglasi transakciju zloupotrebom.

Važno je istaći da prosječna greška nad testnim setom u procesu unakrsne validacije ne varira previše u zavisnosti od odabranih hiperparametara, čime se ističe uticaj

brojnih ugrađenih optimizacija XGBoosta-a. Za razliku od GBDT-a i AdaBoost-a koji su zahtijevali mnogo vremena za treniranje modela, XGBoost nudi mogućnost izvršavanja na grafičkoj kartici. Ova mogućnost kao i optimizovano korišćenje memorijskih resursa su znatno ubrzali proces treninga.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
osnovni model	0.9995	0.9483	0.7432	0.8333
optimizovani model	0.9995	0.9496	0.7635	0.8464
ROS 50-50	0.9995	0.9274	0.7770	0.8456
ROS 10-90	0.9995	0.9280	0.7838	**0.8498**
BSMOTE 50-50	0.9994	0.8722	0.7838	0.8256
BSMOTE 10-90	0.9995	0.9063	0.7838	0.8406
RUS 50-50	0.9670	0.0447	0.8851	0.0851
RUS 10-90	0.9973	0.3713	0.8378	0.5145
Tomek	0.9995	0.9250	0.7500	0.8284
SMENN 50-50	0.9991	0.7011	0.8243	0.7578
SMENN 10-90	0.9992	0.7469	0.8176	0.7806

Tabela 9: Rezultati evaluacije modela XGBoost-a sa različitim tehnikama balansiranja podataka.

6.0.8 CatBoost rezultati

Osnovni model CatBoost-a kreira 1000 stabala odlučivanja maksimalne dubine 6, uz regularizacioni hiperparametar sa vrijednošću 3. Algoritam određuje korak učenja, ukoliko nije zadata vrijednost hiperparametra regularizacije, automatski na osnovu maksimalnog broja iteracija. Uvidom u rezultate modela sa navedenim podrazumijevanim hiperparametrima, prikazanih u Tabeli 10, može se primijetiti velika preciznost detekcije.

U procesu GSCV-a koristiće se model kome će broj iteracija biti limitiran na 500 pri čemu će na nivou svakog čvora biti testirano 254 moguće vrijednosti praga svake numeričke karakteristike, što je preporuka ukoliko se žele postići maksimalne performanse. Broj iteracija je smanjen usljed praćenja greške na trening i testnom setu, gdje se primijetilo da se model u 1000 iteracija previše prilagođava podacima, te da je 500 iteracija dovoljno. Vrijednosti hiperparametara koje će se testirati obuhvataju: korak učenja $\eta \in \{0.01, 0.04, 0.05, 0.1, 0.15, 0.2\}$, dubinu osnovnog klasifikatora $\{6, 8, 10\}$, kao i regularizacioni hiperparametar $\lambda \in \{1, 5, 7\}$. Vrijednosti su odabrane eksperimentalnim putem u cilju povećavanja kompleksnosti osnovnih klasifikatora

povećavanjem njegove dubine, pri čemu će se povećavanjem regularizacije smanjiti mogućnost preprilagođavanja modela trening setu. Za korak učenja, koji se više ne bira automatski, su odabrane različite male vrijednosti od 0 do 0.2.

Na osnovu rezultata prikazanih u Tabeli 10 pokazuje se da model sa vrijednostima hiperparametara $\eta = 0.15$, $\lambda = 1$ koji koristi stabla odlučivanja maksimalne dubine 8, uz primjenu tehnike preodabiranja ROS 10-90 pokazuje najbolje performanse. Ovaj model je nadmašio vrijednosti svih prethodnih modela što ga čini najboljim izborom za detektovanje zloupotreba platnih kartica. Važno je istaći da kao i kod XGBoost-a, vrijednosti F_1 mjere u procesu unakrsne validacije su za većinu kombinacija hiperparametara izuzetno visoke, što se također može prepisati brojnim optimizacijama ovog algoritma. Dodatno, mogućnost izvršavanja ovog algoritma na grafičkoj kartici ubrzava treniranje i testiranje modela više od tri puta.

Primjenom tehnika preodabiranja i tehnike Tomekove veze, uz odabir optimalne kombinacije hiperparametara koji sprečavaju preprilagođavanje modela trening setu popravljaju se performanse F_1 mjere u odnosu na prikazane rezultate osnovnog modela. Za razliku od ovih tehnika RUS i SMENN ne postižu zadovoljavajuće rezultate zbog čega se ne preporučuju. Zanimljivo je istaći da primjenom tehnike RUS 50-50, kao i kod ostalih algoritama, model postiže najveću vrijednost odziva. Međutim, veoma niska preciznost, gdje je svega 6% transakcija detektovanih kao zloupotrebe dobro klasifikovano čini ovaj model neupotrebljivim u praksi.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
osnovni model	0.9995	0.9558	0.7298	0.8276
optimizovani model	0.9996	0.9741	0.7635	0.8561
ROS 50-50	0.9995	0.9091	0.8108	0.8571
ROS 10-90	0.9996	0.9516	0.7973	**0.8676**
BSMOTE 50-50	0.9995	0.8750	0.8041	0.8380
BSMOTE 10-90	0.9995	0.9141	0.7905	0.8478
RUS 50-50	0.9781	0.0650	0.8716	0.1210
RUS 10-90	0.9984	0.5299	0.8378	0.6492
Tomek	0.9995	0.9417	0.7635	0.8433
SMENN 50-50	0.9992	0.7423	0.8176	0.7781
SMENN 10-90	0.9993	0.7727	0.8041	0.7881

Tabela 10: Rezultati evaluacije modela CatBoost-a sa različitim tehnikama balansiranja podataka.

6.1 Testiranje modela sa optimalnim hiperparametrima u 100 iteracija

Kako bi se izbjeglo da performanse modela zavise od fiksirane vrijednosti nasumične inicijalizacije prilikom podjele seta podataka na trening i testni set, sprovedeno je 100 iteracija bez fiksiranja te vrijednosti. Pritom je podjela također vršena korišćenjem stratifikacije, odnosno zadržavajući početni odnos broja uzoraka dvije klase u trening i testnom setu. Na taj način pružena je realnija slika i uvid u performanse modela, smanjujući mogućnost pristrasnosti uzrokovane specifičnom podjelom podataka.

Za svaki algoritam, modeli sa optimalnim hiperparametrima i odgovarajuće tehnike balansiranja, koje su u prethodnim analizama dali najbolje rezultate, testirani su 100 puta. U Tabeli 11 su prikazane prosječne vrijednosti tačnosti, preciznosti, odziva i F_1 mjere koji su pouzdaniji i daju realniju sliku o mogućnostima generalizacije modela. Skoro svi modeli, izuzev RF-a koji postiže na nijansu lošije prosječne rezultate, su u 100 iteracija postigli prosječno veće vrijednosti F_1 mjere. Posebno se ističu AdaBoost i GBDT koji su u prosjeku značajno unaprijedili svoje performanse. Na ovaj način se pokazuje da dobre performanse modela nijesu rezultat slučajne podjele podataka, već da se radi o modelima koji dosljedno dobro generalizuju nezavisno od sastava trening i testnog seta.

	Tačnost	**Preciznost**	**Odziv**	**F1 mjera**
LR	0.9992	0.7885	0.7736	0.7796
KNN	0.9995	0.9307	0.7839	0.8505
DT	0.9994	0.8862	0.7673	0.8215
RF	0.9995	0.9367	0.7928	0.8583
AdaBoost	0.9995	0.9312	0.7686	0.8416
GBDT	0.9995	0.9249	0.7997	0.8574
XGBoost	0.9996	0.9236	0.8208	0.8688
CatBoost	0.9996	0.9290	0.8176	0.8694

Tabela 11: Prosječne vrijednosti metrika modela sa odabranim hiperparametrima i tehnikama balansiranja u 100 iteracija, sa slučajnom podjelom na trening i test set primjenom stratifikacije.

6.2 Brzina treninga i predikcije

Kako se navike, rashodi korisnika i cijene proizvoda stalno mijenjanju i zavise od brojnih faktora, neophodno je s vremena na vrijeme vršiti ponovno treniranje modela. Vrijeme treniranja modela sa odabranim hiperparametrima na trening setu i njegove predikcije na testnom setu dati su u Tabeli 12. Računar koji je korišćen za treniranje i testiranje modela je desktop model sa Z590 GAMING X matičnom pločom. Opremljen je Intel Core i7-11700 procesorom koji radi na taktu od 2.5 GHz. Računar posjeduje 16 GB RAM memorije i NVIDIA GeForce RTX 3070 grafičku karticu.

KNN algoritam zbog načina funkcionisanja, gdje je pri svakoj predikciji neophodno proći kroz čitav trening set i naći najbliže susjede, ima veoma dugo vrijeme predikcije od 8.93 sekunde. U realnim slučajevima gdje je neophodno odobriti transakciju u veoma kratkom vremenu, njegova primjena ne bi bila efikasna.

Ansambl metodi bazirani na stablu odlučivanja poput AdaBoost-a, GBDT-a i RF-a usljed generisanja mnogo stabala odlučivanja zahtijevaju veoma dugo vrijeme treninga, pri čemu vrijeme predikcije takođe nije zadovoljavajuće.

Brojnim optimizacijama i mogućnošću izvršavanja na grafičkoj kartici, XGBoost i CatBoost se pokazuju kao mnogo efikasniji. Model CatBoost-a koji je postigao najbolje rezultate F_1 mjere zahtijeva svega 6.25 sekundi za treniranje i 0.02 sekunde za predikciju. Važno je naglasiti da su ovo rezultati predikcije nad čitavim testnim setom, koji čine 148 uzoraka.

Model	**Vrijeme treninga**	**Vrijeme predikcije**
LR	0.01 sek	0 sek
KNN	0.01 sek	8.93 sek
DT	29.76 sek	0.04 sek
RF	1 min 22.21 sek	0.16 sek
AdaBoost	8 min 2.72 sek	1.32 sek
GBDT	15 min 49.82 sek	0.14 sek
XGBoost	12.26 sek	0.04 sek
CatBoost	6.25 sek	0.02 sek

Tabela 12: Vrijeme treninga i predikcije za različite modele.

6.3 Uticaj promjene praga odlučivanja

Kako bi testirali kako promjena praga odlučivanja utiče na vrijednost preciznosti i odziva razmatraće se model CatBoost-a sa podešenim hiperparametrima, koji je pokazao najbolje rezultate, nakon primjene tehnike preodabiranja ROS 10-90. Algoritam vraća vjerovatnoću pripadnosti pozitivnoj klasi na osnovu koje se sve one vrijednosti veće od podrazumijevane vrijednosti praga odlučivanja (0.5) tretiraju kao pozitivne, odnosno primjer zloupotrebe. Matrica konfuzije je predstavljena Tabelom 13. Od 148 zloupotreba model je ispravno klasifikovao 118 uzoraka, pri čemu je 30 zloupotreba pogrešno klasifikovao kao regularne, a za 6 regularnih je podigao lažni alarm.

	Klasifikovana kao regularna	**Klasifikovana kao zloupotreba**
Regularna transakcija	85289	6
Zloupotreba	30	118

Tabela 13: Matrica konfuzije za vrijednost praga odlučivanja 0.5.

Vrijednosti preciznosti i odziva za različite pragove su date u Tabeli 14. Na osnovu prikazanih vrijednosti može se zaključiti da promjena praga odlučivanja za konkretan model ne može poboljšati odziv, odnosno da model ne sumnja u regularnost onih zloupotreba koje je klasifikovao kao regularne. Naime, uvidom u procijenjene vjerovatnoće zloupotreba klasifikovanih kao regularne transakcije, zaključuje se da im je vjerovatnoća pripadnosti pozitivnoj klasi jednaka 0, te da promjena praga ne može uticati na poboljšanje njihove detekcije. Uvidom u ove uzorke od strane stručne osobe uz poznavanje značenja karakteristika moglo bi se dodatno istražiti šta je uzrok, da li ove transakcije ne odudaraju od navika korisnika po karakteristikama koje su analizirane ili je problem do prenaučenosti modela. Međutim, za dati set podataka ovaj model je pokazao najbolju generalizaciju uzimajući u obzir F_1 mjeru, stoga bi se eventualno poboljšavanje odziva moglo tražiti u dodatnim podacima.

Analizom promjena praga ostalih modela pokazuje se da iako niži prag može povećati stopu detekcije zloupotreba, porast odziva je znatno manji u poređenju sa padom preciznosti. Stoga, izbor optimalnog praga zavisi od specifičnih potreba i ciljeva koje se žele postići.

Prag	**Preciznost**	**Odziv**
0.05	0.8252	0.7973
0.10	0.8551	0.7973
0.15	0.8741	0.7973
0.20	0.9147	0.7973
0.25	0.9147	0.7973
0.30	0.9291	0.7973
0.35	0.9440	0.7973
0.40	0.9440	0.7973
0.45	0.9440	0.7973
0.50	0.9516	0.7973
0.55	0.9516	0.7973
0.60	0.9516	0.7973
0.65	0.9512	0.7905
0.70	0.9512	0.7905
0.75	0.9580	0.7703
0.80	0.9576	0.7635
0.85	0.9658	0.7635
0.90	0.9739	0.7568
0.95	0.9820	0.7365

Tabela 14: Vrijednost preciznosti i odziva za različite vrijednosti praga odlučivanja.

Zaključak

Razvoj interneta, mogućnost *online* trgovine i jednostavnost korišćenja platnih kartica uticali su na sve veću upotrebu ovog vida plaćanja. Iako sve veći broj banaka implementira sigurnosne metode u cilju zaštite korisnika od zloupotreba njihovih platnih kartica, veliki iznosi na godišnjem nivou se otuđe sa računa korisnika. Izvršioci zloupotreba se prilagođavaju sistemima detekcije i pronalaze način da se dokopaju kartice ili podataka koji se na njoj nalaze i da vrše neautorizovane transakcije. Osim same štete po korisnika kome su sredstva otuđena, veliki rizik snose banke čiji se integritet i sigurnost dovodi u pitanje.

Kako bi se riješio ovaj problem neophodno je napraviti model koji će biti u stanju da veoma brzom reakcijom spriječi pokušaj zloupotrebe i zaštiti korisnike. Ovaj rad nudi sistematičan pristup rješavanju problema detekcije zloupotreba platnih kartica, koji obuhvata različite tehnike za pretprocesiranje sirovih podataka i upotrebu klasifikatora u cilju detektovanja sumnjivih transakcija. Rad daje pregledni opis tehnika koje se mogu koristiti u cilju pripreme seta podataka za korišćenje klasifikatora, kao i uslove pod kojima ih ima smisla koristiti.

Za testiranje predloženog modela korišćen je veoma aktuelan set podataka među akademskom zajednicom. Iako su određene tehnike opisane u radu već primijenjene nad korišćenim setom ovdje su data njihova detaljna pojašnjenja, kao i određene alternative koje se mogu koristiti u problemima detekcije zloupotreba platnih kartica.

U radu su testirane performanse 8 klasifikatora nadgledanog mašinskog učenja, kao i uticaj primjena tehnika za balansiranje seta na njihove performanse. U predloženom modelu se naglašava neophodnost primjene pomenutih tehnika samo nad trening setom u cilju nepristrasne kasnije evaluacije klasifikatora.

Na osnovu rezultata dobijenih modela pokazuje se da odabir tehnike balansiranja u velikoj mjeri zavisi od korišćenog klasifikatora. Tehniku slučajnog pododabiranja (RUS) ima smisla upotrebljavati samo sa logističkom regresijom, dok se za ostale klasifikatore ne preporučuje za konkretan problem. Za razliku od nje, tehnike preodabiranja poput ROS i BSMOTE-a pokazuju da se povećavanjem broja uzoraka manjinske klase, ili primjenom tehnike Tomekove veze u cilju jasnijeg razdvajanja uzoraka može poboljšati vrijednost F_1 mjere. Hibridna tehnika SMENN je pokazala dobre rezultate odziva u primjeni sa RF-om, stoga je korisno razmatrati u kombinaciji sa ansambl metodama.

Iako postiže veoma visoku vrijednost F_1 mjere, KNN zbog dužine trajanja predikcije nije najbolje rješenje. Najbolje rezultate odziva i F_1 mjere postižu ansambl metodi zasnovani na stablu odlučivanja: RF, XGBoost i CatBoost. Za razliku od

njih GBDT i AdaBoost ne postižu značajni napredak u odnosu na osnovno stablo odlučivanja, što ih zajedno sa velikim vremenom potrebnim za treniranje modela ne preporučuje kao pogodne za rješavanje ovog problema.

Najbolje rezultate postiže CatBoost algoritam, sa podešenim hiperparametrima, treniran na balansiranom setu podataka tehnikom ROS u finalnom odnosu broja uzoraka manjinske i većinske klase 10-90 respektivno. CatBoost uvođenjem principa uređenosti prilikom kreiranja ansambla i brojnih optimizacija uspijeva da izađe na kraj sa problemom neizbalansiranosti i postigne značajne performanse $F_1 = 0.8694$. Osim dobrih performansi mogućnost izvršenja ovog algoritma na grafičkoj kartici omogućili su ovom algoritmu najkraće ukupno vrijeme treninga i predikcije u odnosu na sve ostale predložene modele. Dodatno je prikazano kako se uticajem na prag odlučivanja može direktno uticati na vrijednost preciznosti i odziva.

Literatura

[1] UK Finance. The definitive overview of payment industry fraud in 2023. Annual Fraud Report 2023, 2023. Retrieved August 10, 2023, from `https://www.ukfinance.org.uk/policy-and-guidance/reports-and-publications/annual-fraud-report-2023`.

[2] Esraa Faisal Malik, Khai Wah Khaw, Bahari Belaton, Wai Peng Wong, and XinYing Chew. Credit card fraud detection using a new hybrid machine learning architecture. *Mathematics*, 10(9):1480, 2022.

[3] Noor Saleh Alfaiz and Suliman Mohamed Fati. Enhanced credit card fraud detection model using machine learning. *Electronics*, 11(4):662, 2022.

[4] Zahra Salekshahrezaee, Joffrey L Leevy, and Taghi M Khoshgoftaar. The effect of feature extraction and data sampling on credit card fraud detection. *Journal of Big Data*, 10(1):1–17, 2023.

[5] Altyeb Altaher Taha and Sharaf Jameel Malebary. An intelligent approach to credit card fraud detection using an optimized light gradient boosting machine. *IEEE Access*, 8:25579–25587, 2020.

[6] Michele Carminati, Roberto Caron, Federico Maggi, Ilenia Epifani, and Stefano Zanero. Banksealer: A decision support system for online banking fraud analysis and investigation. *computers & security*, 53:175–186, 2015.

[7] John O Awoyemi, Adebayo O Adetunmbi, and Samuel A Oluwadare. Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 international conference on computing networking and informatics (ICCNI)*, pages 1–9. IEEE, 2017.

[8] Jonathan Kwaku Afriyie, Kassim Tawiah, Wilhelmina Adoma Pels, Sandra Addai-Henne, Harriet Achiaa Dwamena, Emmanuel Odame Owiredu, Samuel Amening Ayeh, and John Eshun. A supervised machine learning algorithm for detecting and predicting fraud in credit card transactions. *Decision Analytics Journal*, page 100163, 2023.

[9] Dejan Varmedja, Mirjana Karanovic, Srdjan Sladojevic, Marko Arsenovic, and Andras Anderla. Credit card fraud detection-machine learning methods. In *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*, pages 1–5. IEEE, 2019.

[10] MLG - ULB. Credit card fraud dataset. Online, 2019. Accessed: Sep. 4, 2019.

[11] Maram Alamri and Mourad Ykhlef. Survey of credit card anomaly and fraud detection using sampling techniques. *Electronics*, 11:4003, 12 2022.

[12] Abdulla Muaz, Manoj Jayabalan, and Vinesh Thiruchelvam. A comparison of data sampling techniques for credit card fraud detection. *International Journal of Advanced Computer Science and Applications*, 11(6), 2020.

[13] Prasetyo Wibowo and Chastine Fatichah. An in-depth performance analysis of the oversampling techniques for high-class imbalanced dataset. *Register: Jurnal Ilmiah Teknologi Sistem Informasi*, 7(1):63–71, 2021.

[14] Appala Hema. Machine learning methods for discovering credit card fraud. irjcs:: International research journal of computer science, volume viii, 01-06, 2020.

[15] Fabrizio Carcillo, Yann-Aël Le Borgne, Olivier Caelen, Yacine Kessaci, Frédéric Oblé, and Gianluca Bontempi. Combining unsupervised and supervised learning in credit card fraud detection. *Information sciences*, 557:317–331, 2021.

[16] Ugo Fiore, Alfredo De Santis, Francesca Perla, Paolo Zanetti, and Francesco Palmieri. Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. *Information Sciences*, 479:448–455, 2019.

[17] Roberto Saia, Salvatore Carta, et al. A frequency-domain-based pattern mining for credit card fraud detection. In *IoTBDS*, pages 386–391, 2017.

[18] Roberto Saia. A discrete wavelet transform approach to fraud detection. In *Network and System Security: 11th International Conference, NSS 2017, Helsinki, Finland, August 21–23, 2017, Proceedings 11*, pages 464–474. Springer, 2017.

[19] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

[20] Alan M Turing. *Computing machinery and intelligence*. Springer, 2009.

[21] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc., 2 edition, Sep 2019.

[22] Ch Sanjeev Kumar Dash, Ajit Kumar Behera, Satchidananda Dehuri, and Ashish Ghosh. An outliers detection and elimination framework in classification task of data mining. *Decision Analytics Journal*, page 100164, 2023.

[23] I. Tomek. Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, 6(11):769–772, 1976.

[24] I. Tomek. An experiment with the edited nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, 6(6):448–452, 1976.

[25] Paula Branco, Luis Torgo, and Rita Ribeiro. A survey of predictive modelling under imbalanced distributions, 2015.

[26] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[27] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, pages 1322–1328. Ieee, 2008.

[28] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[29] Fayaz Itoo and Satwinder Singh. Comparison and analysis of logistic regression, naïve bayes and knn machine learning algorithms for credit card fraud detection. *International Journal of Information Technology*, 13:1503–1511, 2021.

[30] Konduri Praveen Mahesh, Shaik Ashar Afrouz, and Anu Shaju Areeckal. Detection of fraudulent credit card transactions: A comparative analysis of data sampling and classification techniques. *Journal of Physics: Conference Series*, 2161(1):012072, jan 2022.

[31] Samidha Khatri, Aishwarya Arora, and Arun Prakash Agrawal. Supervised machine learning algorithms for credit card fraud detection: a comparison. In *2020 10th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pages 680–683. IEEE, 2020.

[32] Srinivasa Rao Dammavalam and Md Mukheed. Credit card fraud detection using machine learning. *International Journal of Advances in Engineering and Management*, 5(1):147–154, 2023.

[33] Maja Puh and Ljiljana Brkić. Detecting credit card fraud using selected machine learning algorithms. In *2019 42nd International Convention on Information*

and Communication Technology, Electronics and Microelectronics (MIPRO), pages 1250–1255. IEEE, 2019.

[34] Leif E Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009.

[35] Danny Coomans and Désiré Luc Massart. Alternative k-nearest neighbour rules in supervised pattern recognition: Part 1. k-nearest neighbour classification by using alternative voting rules. *Analytica Chimica Acta*, 136:15–27, 1982.

[36] Oliver Kramer and Oliver Kramer. K-nearest neighbors. *Dimensionality reduction with unsupervised nearest neighbors*, pages 13–23, 2013.

[37] Roger J Lewis. An introduction to classification and regression tree (cart) analysis. In *Annual meeting of the society for academic emergency medicine in San Francisco, California*, volume 14. Citeseer, 2000.

[38] Batta Mahesh. Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9(1):381–386, 2020.

[39] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[40] Adele Cutler, D Richard Cutler, and John R Stevens. Random forests. *Ensemble machine learning: Methods and applications*, pages 157–175, 2012.

[41] K. R. Seeja, Masoumeh Zareapoor, and Wenyu Zhang. Fraudminer: A novel credit card fraud detection model based on frequent itemset mining. *The Scientific World Journal*, 2014:252797, 2014.

[42] Robert E. Schapire. A brief introduction to boosting. In *International Joint Conference on Artificial Intelligence*, 1999.

[43] Robert E Schapire. Explaining adaboost. In *Empirical inference: festschrift in honor of vladimir N. Vapnik*, pages 37–52. Springer, 2013.

[44] Youwei Wang and Lizhou Feng. An adaptive boosting algorithm based on weighted feature selection and category classification confidence. *Applied Intelligence*, pages 1–22, 2021.

[45] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[46] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[47] Jerome H Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 38(4):367–378, 2002.

[48] XGBoost - ML winning solutions (incomplete list). GitHub. Archived from the original on 2017-08-24. Retrieved 2016-08-01.

[49] Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

[50] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

[51] Vaishnavi Nath Dornadula and Sa Geetha. Credit card fraud detection using machine learning algorithms. *Procedia computer science*, 165:631–641, 2019.

[52] Frank J Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.