SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 5997

Postupci strojnog učenja za popravljanje točnosti klasifikacije manjinskih klasa kod nebalansiranih skupova podataka

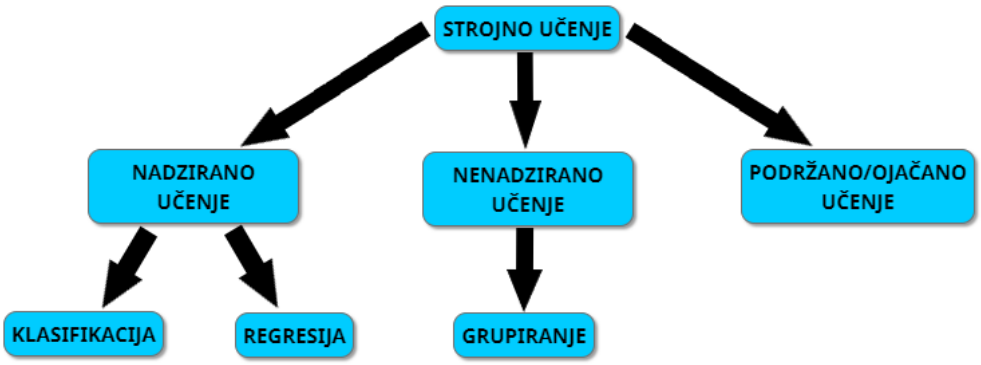
Marko Josipović

Zagreb, lipanj 2019.

SADRŽAJ

1. Uvod

Strojno učenje je grana umjetne inteligencije koja se bavi algoritmima za automatsku obradu podataka. Podatci se u današnje vrijeme najčešće pribavljaju u digitalnom obliku s velikom pouzdanošću i u velikim količinama poput zdravstvenih nalaza, meteoroloških mjerenja, navika ljudi koji pretražuju internet ili potrošačkih navika onih koji podižu kredite. Svi ti podatci nam nisu od velike pomoći ako iz njih ne možemo izvući korisne informacije. Algoritmi strojnog učenja na temelju podataka izgrađuju matematički model pomoću kojeg se mogu zaključiti znanja o skupu podataka ili predvidjeti svojstva novih skupova podataka. Postoji više vrsta strojnog učenja kao što je prikazano na slici 1.1, glavna podjela je na nadzirano i nenadzirano te podržano učenje. Nenadzirano učenje obrađuje podatke koji nisu označeni, razvrstani ili kategorizirani i pronalazi pravilnosti u njima, a glavna primjena ove vrste učenja je u statistici kod procjena gustoća. Najčešća tehnika nenadziranog učenja je grupiranje koje nam omogućava da organiziramo skup podataka u podgrupe koje dijele određeni stupanj sličnosti ali su dovoljno drugačiji od podataka iz drugih grupa (Raschka, 2015). Neki od problema koji se rješavaju grupiranjem su analiziranje sekvenci gena, prepoznavanje uzoraka i kompresija podataka.



Slika 1.1: Vrste strojnog učenja

Podržano učenje se temelji na tome da se agenta koji rješava određeni problem na-

gradi ako povuče dobar potez (npr. tijekom igranja šaha). Agent nastoji maksimizirati svoje nagrade i tako "uči" snalaziti se u određenoj okolini pomoću prošlih iskustava.

Algoritmi nadziranog učenja izgrađuju model na temelju ulaznih ali i poznatih izlaznih podataka, pri čemu skup podataka koji sadrži ulazne i izlazne podatke nazivamo podacima za treniranje. Nakon učenja na skupu podataka za treniranje model je sposoban s određenom točnošću predvidjeti rezultate obrade podataka koji nisu bili među podatcima za treniranje. Klasifikacija i regresija su dvije vrste nadziranog učenja a razlikuju se po obliku izlaznih podataka, kod klasifikacije izlaz je limitirani skup vrijednosti dok je izlaz regresijskih algoritama kontinuirana vrijednost. Klasifikacijom se rješavaju problemi poput označavanja neželjene elektroničke pošte, prepoznavanja objekata kod računalnog vida te raspoznavanja govora i rukom pisanih znakova dok se regresija može koristiti za predviđanje cijena nekretnina ili cijena dionica, predviđanje opterećenja na sustave koji imaju više izvora opterećenja ili za predviđanje utjecaja reklamiranja na prodaju proizvoda.

Fokus ovog rada je na klasifikaciji, točnije na problemu nebalansiranosti skupa podataka koji se koristi za učenje modela. Nebalansirani skup podataka je onaj skup u kojemu se uzorci jedne klase pojavljuju u malom broju naspram ostalih klasa. Primjer takvih skupova podataka su bankovne transakcije koristeći kreditne kartice kod kojih je broj legitimnih transakcija puno veći od broja transakcija koje su posljedica prijevara korisnika kreditnih kartica. Nebalansirani skupovi podataka su česti kada se proučavaju poremećaji ili bolesti koje su vrlo rijetke u odnosu na zdravu populaciju. Ovakvi skupovi podataka predstavljaju problem pri klasifikaciji zbog toga što će klasifikacijski algoritmi učeni na nebalansiranom skupu podataka imati sklonost klasificirati manjinske klase kao većinske. Kako bi doskočili ovom problemu potrebno je balansirati skup podataka ili odabrati algoritme koji pri učenju uzimaju u obzir nebalansiranost podataka, balansiranje skupa ponovnim uzorkovanjem je poželjan način rješavanja nebalansiranosti zbog jednostavnosti primjene. U ovom radu ću eksperimentalno vrednovati postupke strojnog učenja za popravljanje točnosti klasifikacije i usporediti njihovu učinkovitost.

2. Postupci za popravljanje točnosti

U ovom poglavlju ću opisati postupke za popravljanje točnosti koji su korišteni pri eksperimentalnom vrednovanju. U kasnijim poglavljima je opisan postupak vrednovanja i usporedbe rezultata različitih postupaka ili kombinacija postupaka te skupovi podataka nad kojima je obavljana klasifikacija. Postupci za popravljanje točnosti i klasifikacijski algoritmi su preuzeti iz python paketa `imbalanced-learn` (Lemaître et al., 2017) osim klasifikatora k-najbližih susjeda koji je preuzet iz paketa `scikit-learn`.

Postupci koji se analiziraju u ovom radu uključuju dva postupka naduzorkovanja skupova podataka, dva postupka poduzorkovanja te korištenje klasifikatora balansirane slučajne šume kao superiornijeg klasifikatora nad klasifikatorom k-najbližih susjeda kada su u pitanju nebalansirani skupovi podataka.

2.1. Naduzorkovanje

Naduzorkovanje je generiranje novih uzoraka bilo ponavljanjem originalnih uzoraka ili generiranjem novih određenom tehnikom. Povećanjem broja uzoraka manjinske klase postiže se balansiranost skupa podataka i smanjuje pristranost algoritama strojnog učenja prema većinskoj klasi.

2.1.1. Algoritam Synthetic Minority Oversampling Technique

Ovaj postupak (kraće: SMOTE) je implementacija tehnike koja je opisana u znanstvenom radu (Chawla et al., 2002) i vrlo je popularna. Tehnika obavlja ponovno uzorkovanje ulaznog skupa podataka tako da se broj podataka manjinskih klasa poveća umjetnom generacijom. Umjetna generacija izbjegava problem prenaučenosti modela. Novi podaci manjinske klase se generiraju interpolacijom oko broja najbližih susjeda koji je određen parametrom.

Definicija razreda:

```
class imblearn.over_sampling.SMOTE(sampling_strategy='minority',
                                   random_state=None, k_neighbors=3)
```

Pomoću `sampling_strategy` zadaje se informacija o ponovnom uzorkovanju. Moguće je zadati broj s pomičnim zarezom (`float`) koji označava željeni omjer broja uzoraka većinske klase i broja uzoraka manjinske klase, a ovaj način ponovnog uzorkovanja je moguć samo kod binarne klasifikacije. Zadajući niz znakova (`str`) određujemo klasu koja će biti ponovno uzorkovana: `'auto'` zapravo označava vrijednost `'not majority'` koja će uzorkovati sve klase osim većinske dok `'not minority'` označava sve klase osim manjinske. `'minority'` će uzorkovati samo manjinsku a `'all'` sve klase. Predana vrijednost može biti i funkcija koja vraća rječnik (`'dict'`) ili se može predati samo rječnik, a tada su ključevi rječnika klase koje treba uzorkovati a vrijednosti povezane s tim ključevima su željeni brojevi uzoraka pojedine klase.

Parametrom `random_state` kontroliramo način generacije slučajnih vrijednosti u algoritmu, `None` označava korištenje razreda `RandomState` paketa `np.random`.

`k_neighbors` određuje broj susjednih uzoraka na temelju kojih će se odrediti vrijednosti novih, generiranih uzoraka.

Parametri koji su korišteni pri eksperimentalnoj evaluaciji su upravo oni navedeni na početku odjeljka u definiciji razreda tj. naduzorkovati će se samo manjinska klasa pomoću generacije novih uzoraka na temelju 3 susjedne točke.

2.1.2. Algoritam Adaptive Synthetic Sampling Method (ADASYN)

Ovaj postupak također koristi naduzorkovanje, implementiran je na temelju znanstvenog rada (He et al., 2008). Poboljšanje u točnosti klasifikacije se postiže dinamičkom promjenom ponderirane aritmetičke sredine koja ovisi o težini učenja uzoraka klase gdje se generira više umjetnih uzoraka za klase koje je teže naučiti.

Definicija je jednaka onoj kod tehnike SMOTE što znači da se ova dva postupka mogu koristiti na isti način:

```
class imblearn.over_sampling.ADASYN(sampling_strategy='minority',
                                    random_state=None, n_neighbors=3)
```

Za generaciju uzoraka se koristi interpolacija ali razlika između uzoraka generiranih postupkom SMOTE i ADASYN je u tome što se ADASYN fokusira na generiranje uzoraka oko originalnih uzoraka koji su krivo klasificirani koristeći klasifikator

k-najbližih susjeda, tj. oko onih uzoraka koje je teže klasificirati. SMOTE ne uzima u obzir težinu klasifikacije.

Vrijednost `'minority'` parametra `sampling_strategy` označava da će se naduzorkovati samo manjinska klasa.

2.2. Poduzorkovanje

Poduzorkovanje je suprotno od naduzorkovanja, poduzorkovanjem se smanjuje broj uzoraka većinskih klasa. Ovakve metode mogu dovesti do gubitka informacija ako se ne primjene pažljivo.

2.2.1. Algoritam Random Under Sampler

Ovo je postupak koji omogućava brzo i jednostavno balansiranje skupa podataka tako što nasumično izabire uzorke koje će zadržati u skupu dok će ostale odbaciti.

```
class imblearn.under_sampling.RandomUnderSampler(sampling_strategy='
                                    auto', random_state=42)
```

`sampling_strategy` parametar određuje informaciju o ponovnom uzorkovanju isto kao i kod metoda naduzorkovanja. U ovom slučaju poduzorkovati će se sve klase osim manjinske jer je parametar `'auto'` ekvivalentan parametru `'not minority'` koji će poduzorkovati sve klase osim manjinske. Broj uzoraka većinskih klasa će se izjednačiti s brojem uzoraka manjinske klase.

`random_state` također ima istu funkciju kao i kod postupaka naduzorkovanja, broj `42` označava da nasumični generator koristi taj broj kao sjeme pri generaciji brojeva. Ponovno uzorkovanje skupa podataka koji sadrži više klasa se provodi pojedino za svaku klasu.

2.2.2. Algoritam NearMiss Under Sampler

Postupak poduzorkovanja *NearMiss* implementiran je na temelju znanstvenog rada (Zhang i Mani, 2003). Ovaj postupak koristi heuristiku pri odabiru uzoraka koje će zadržati.

```
class imblearn.under_sampling.NearMiss(sampling_strategy='auto',
                                    version=2, n_neighbors=3)
```

Implementirana su tri različita tipa heuristika koja se mogu odabrati parametrom `version` čije su moguće vrijednosti brojevi 1, 2 ili 3. Heuristike se temelje na algoritmu najbližih susjeda. Prva verzija heuristike odabire uzorke većinskih klasa čija je prosječna udaljenost od `n_neighbors` najbližih susjeda manjinske klase najmanja. Druga verzija odabire uzorke većinskih klasa čija je prosječna udaljenost od `n_neighbors` najdaljih susjeda manjinske klase najmanja. Zadnja verzija heuristike prvo odabire `n_neighbors_ver3` najbližih susjeda manjinskih klasa te zatim od tih izabranih uzoraka većinskih klasa odabire one čija je prosječna udaljenost od `n_neighbors` najveća.

Moguće vrijednosti parametara `sampling_strategy` su jednake onima prijašnjih postupaka, u ovom slučaju poduzorkovati će se sve klase osim manjinske.

2.3. Klasifikacijski algoritmi

2.3.1. Klasifikator k-najbližih susjeda

Algoritam k-najbližih susjeda je negeneralizirajuć, tj. ne izgrađuje model nego jednostavno pohranjuje uzorke skupa za učenje. Klasifikacija se obavlja pomoću većine glasova određenog broja (k) najbližih susjeda svakog uzorka, uzorku se pridaje klasa koja ima najviše glasova susjednih uzoraka poznate klase. Optimalan broj susjeda na temelju kojih se određuje klasa novog uzorka uvelike ovisi o podacima, veći broj susjeda koji se uzimaju u obzir će smanjiti utjecaj šuma u podacima ali će granice klasifikacije učiniti manje izrazitim.

Definicija algoritma u paketu `scikit-learn`:

```
class sklearn.neighbors.KNeighborsClassifier(n_neighbors=3, weights=
                                    'uniform', algorithm='auto',
                                    leaf_size=30, p=2, metric='
                                    minkowski', metric_params=None)
```

`n_neighbors` određuje broj susjeda na temelju kojih obavljamo klasifikaciju. Parametar `weights` označava težinsku funkciju kojom pojedinim susjedima pridjeljujemo značajnost njihovih glasova. Ako je vrijednost ovog parametra `'uniform'`, tada će glasovi svih susjeda biti jednako značajni dok vrijednost `'distance'` pridaje veću značajnost bližim susjedima tj. značajnost glasa pada kada udaljenost od točke upita raste. Za računanje značajnosti može se iskoristiti i vlastita funkcija tako da se preda pokazivač na funkciju koja prima udaljenosti susjeda od točke upita a vraća značajnosti.

Parametrom `algorithm` se odabire algoritam koji se koristi za odabir najbližih susjeda poput algoritama *ball tree*, *k-d tree* ili *brute force* koji nisu predmet ovog rada. Vrijednost `'auto'` označava da će se odabir algoritma automatski obaviti na temelju ulaznog skupa podataka. `leaf_size` se koristi pri traženju susjeda, veličina tog parametra utječe na brzinu izvođenja i na količinu potrebne memorije.

`metric` označava metričku udaljenost koja se koristi pri izradi stabala algoritama za odabir susjeda. Odabrana metrika je `'minkowski'` koja je za vrijednost parametra `p=2` zapravo euklidska norma. Parametrom `metric_params` može se predati dodatne parametre za metriku koji se u ovom slučaju ne koriste.

2.3.2. Klasifikator balansirane slučajne šume

Ovaj algoritam je zapravo ansambl procjenitelja, a implementiran je na temelju znanstvenog rada (Chen i Breiman, 2004). Algoritam nosi taj naziv zbog toga što se sastoji od više stabala odluke. Svako stablo odluke se izgrađuje na temelju vlastitog podskupa podataka koji se nasumično odabire iz početnog skupa podataka. Prilikom učenja na nebalansiranom skupu podataka moguće je da se među podatcima odabranim za izgradnju stabla nađe mali broj ili čak niti jedan uzorak manjinske klase što će umanjiti sposobnost stabla odluke da točno klasificira manjinsku klasu.

Kako bi se povećala točnost klasifikacije manjinske klase mijenja se način odabira podskupa podataka za izgradnju stabala. Najprije se nasumično odaberu uzorci manjinske klase a zatim isti broj uzoraka većinske klasa kako bi podskup postao balansiran. Na temelju odabranih uzoraka izgradi se stablo odluke. Postupak se ponavlja zadani broj puta. Predviđanje se obavlja većinom glasova pojedinih stabala odluke.

Definicija klasifikatora:

```
class imblearn.ensemble.BalancedRandomForestClassifier(n_estimators=
                                    100, random_state=0)
```

Algoritam ima puno neobaveznih parametara, spomenut ću samo par zanimljivih. `criterion='gini'` označava da se za mjerenje kvalitete međurezultata koristi Gini nečistoća. To je mjera koja pokazuje koliko često bi se slučajno odabrani uzorak pogrješno klasificirao ako se klasificira na temelju distribucije klasa u podskupu podataka. Teži se smanjiti nečistoću.

`max_depth` je najveća dubina stabla, kada je vrijednost `None` tada se čvorovi proširuju dok nisu svi listovi čisti, tj. sadrže samo jednu klasu ili dok svi listovi ne sadrže broj uzoraka manji od `min_samples_split`. Ako je vrijednost `min_samples_split` cijeli broj, tada on označava najmanji broj uzoraka koji je

potreban da se obavi podjela čvora, a ako je decimalan broj tada označava postotak prema kojemu je minimalni broj uzoraka potreban za podjelu jednak `ceil(min_samples_split*n_samples)`.

U ovom radu koristit ću samo dva parametra: `n_estimators=100` koji određuje broj procjenitelja i `random_state` s vrijednošću `0` koji ima isto značenje kao i u prijašnjim algoritmima, `0` označava sjeme koje koristi generator slučajnih brojeva pri nasumičnom poduzorkovanju.

3. Skupovi podataka i opis vrednovanja

3.1. Skupovi podataka

Skupovi podataka koji su korišteni pri eksperimentalnom vrednovanju preuzeti su sa UCI repozitorija (Dua i Graff, 2017). Tablica 3.1 prikazuje značajke 5 različitih skupova podataka nad kojima će se obaviti eksperimentalno vrednovanje.

Tablica 3.1: Skupovi podataka

Naziv	Broj uzoraka	Broj atributa	Broj klasa	Udio manjinske klase
Letter-samoglasnik	20000	16	2	19.39%
New-thyroid	215	5	3	13.95%
Satimage	6435	36	6	9.73%
Flag	194	28	2	8.76%
Glass	214	9	2	7.94%

Skup Letter se sastoji od 16 atributa koji opisuju pikselizirani prikaz 26 velikih slova engleske abecede. Svako slovo ima 20 fontova koji su nasumično iskrivljeni kako bi se dobilo 20 tisuća različitih uzoraka. Skup je promatran tako da ima dvije klase: samoglasnike koji su manjinski i suglasnike.

Sljedeći skup na temelju rezultata 5 laboratorijskih testova predviđa rad štitne žlijezde: normalan, hipertireoza u kojem proizvodi previše hormona i hipotireoza tj. nedovoljna proizvodnja hormona. Oznake klasa su temeljene na kompletnom medicinskom zapisu pacijenta uključujući anamnezu. Manjinska klasa je hipotireoza koja ima 30 uzoraka u skupu od 215.

Satimage je skup multi-spektralnih vrijednosti piksela u 3x3 susjedstvu satelitske snimke. Cilj je klasificirati središnji piksel u svakom susjedstvu u jednu od klasa: crveno tlo, sivo tlo, usjev pamuka, vlažno ili jako vlažno sivo tlo i tlo s biljnom vege-

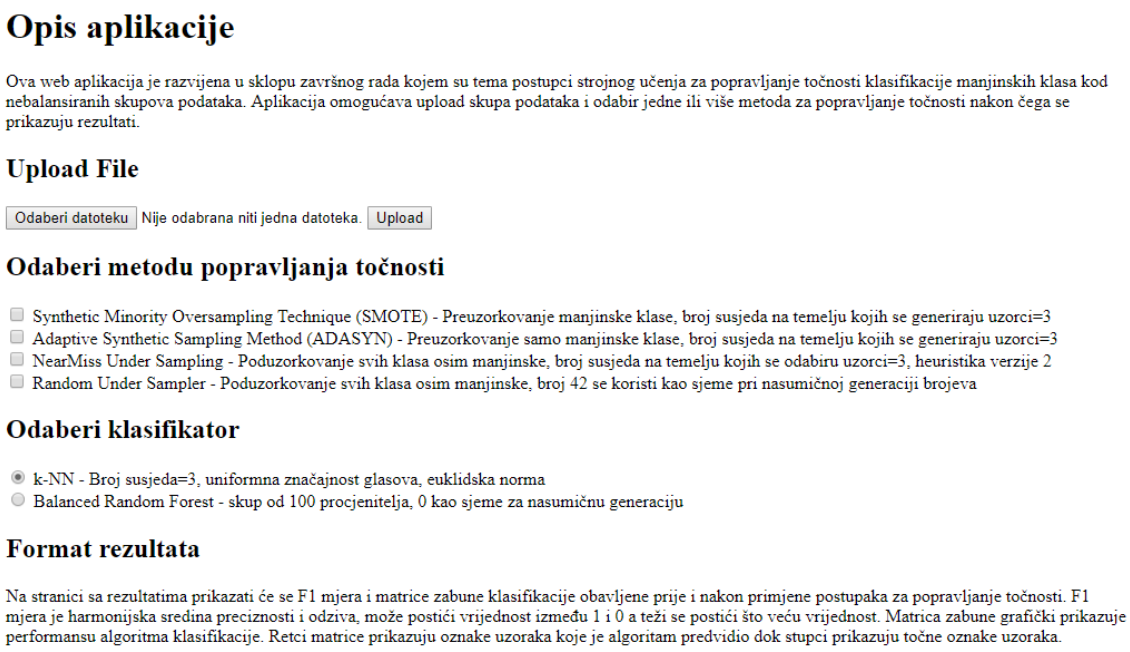
tacijom. Manjinska klasa je jako vlažno sivo tlo.

Skup Flag se sastoji od podataka koji opisuju zemlje i njihove zastave. Neki od atributa su površina u tisućama kvadratnih kilometara, populacija u milijunima, religija i razni parametri službenih zastava: prisutnost određenih boja, broj krugova, trokuta ili zvijezda na zastavi. Cilj klasifikacije je predvidjeti boju u donjem lijevom kutu zastave, bijela boja je manjinska klasa dok su ostale boje većinska.

Posljedni skup se može koristiti pri identifikaciji komada stakla pronađenih na mjestima zločina. Vrste stakla koje su zastupljene su prozori zgrada i automobila te stakleni spremnici, stolno posuđe ili svjetiljka. Atributi na temelju kojih se obavlja identifikacija su indeks loma i postotak nekih elemenata u staklu poput magnezija, aluminija, kalcija i željeza. Manjinska klasa je staklo automobilskog prozora koje je proizvedeno izlijevanjem rastopljenog stakla na sloj rastopljenog metala.

3.2. Postupak vrednovanja

Vrednovanje se obavlja pomoću web aplikacije koja je razvijena u programskom jeziku Python pomoću radnog okvira Flask. Izgled početne stranice aplikacije je prikazan na slici 3.1. Aplikacija korisniku omogućava upload skupa podataka i odabir jednog



Slika 3.1: Početna stranica aplikacije

ili više postupaka popravljanja točnosti klasifikacije koji pripremaju skup podataka za učenje modela strojnog učenja. Korisnik također odabire između jednog od dva

klasifikatora objašnjenih u prijašnjem poglavlju, aplikacija zahtjeva da skup podataka bude u csv formatu (comma separated values) i da oznaka svakog uzorka bude na kraju uzorka. Također, nastavak datoteke koja se učitava mora biti ".csv". Slika 3.2 prikazuje dio skupa Letter. Svaka vrijednost u skupu je razdvojena zarezom, prvih 16 vrijednosti svakog retka su atributi, a posljednja je oznaka uzorka.

letter-recognition.csv

```
2,8,3,5,1,8,13,0,6,6,10,8,0,8,0,8,zatvornik
5,12,3,7,2,10,5,5,4,13,3,9,2,8,4,10,otvornik
4,11,6,8,6,10,6,2,6,10,3,7,3,7,3,9,zatvornik
7,11,6,6,3,5,9,4,6,4,4,10,6,10,2,8,zatvornik
2,1,3,1,1,8,6,6,6,6,5,9,1,7,5,10,zatvornik
4,11,5,8,3,8,8,6,9,5,6,6,0,8,9,7,zatvornik
4,2,5,4,4,8,7,6,6,7,6,6,2,8,7,10,zatvornik
1,1,3,2,1,8,2,2,2,8,2,8,1,6,2,7,otvornik
2,2,4,4,2,10,6,2,6,12,4,8,1,6,1,7,zatvornik
11,15,13,9,7,13,2,6,2,12,1,9,8,1,1,8,zatvornik
3,9,5,7,4,8,7,3,8,5,6,8,2,8,6,7,zatvornik
6,13,4,7,4,6,7,6,3,10,7,9,5,9,5,8,otvornik
4,9,6,7,6,7,8,6,2,6,5,11,4,8,7,8,zatvornik
6,9,8,6,9,7,8,6,5,7,5,8,8,9,8,6,zatvornik
5,9,5,7,6,6,11,7,3,7,3,9,2,7,5,11,zatvornik
6,9,5,4,3,10,6,3,5,10,5,7,3,9,6,9,zatvornik
```

Slika 3.2: Primjer skupa podataka

Isječak koda u nastavku prikazuje postupak obrade skupa podataka, učenja modela pomoću algoritma kNN i zatim predviđanja oznaka.

```
dataset = pd.read_csv(filename, header=None)
dataset = dataset.apply(LabelEncoder().fit_transform)
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
X, y = SMOTE(sampling_strategy='auto', random_state=None,
                                    k_neighbors=3).fit_resample(X, y)
train(X, y)


def train(X, y):
    X_train, X_test, y_train, y_test = train_test_split(X, y,
                                        test_size=1/3, random_state=0)
    neigh = KNeighborsClassifier(n_neighbors=3, weights='uniform',
                                        algorithm='auto', leaf_size=30
                                        , p=2, metric='minkowski',
                                        metric_params=None)
    neigh.fit(X_train, y_train)
    y_pred = neigh.predict(X_test)
```

Najprije se iz učitane datoteke čija je staza spremljena u varijablu `filename` dohvate podatci metodom `read_csv(filename, header=None)`. Podatci se dohvaćaju u obliku `DataFrame`, metoda `read_csv()` i razred `DataFrame` su dio Python paketa `pandas` koji se koristi za analizu podataka. Vrijednost `None` parametra `header` označava da na početku datoteke nisu navedene oznake koje se nalaze u skupu podataka.

Razred `LabelEncoder` i funkcija `train_test_split()` su dio paketa `sklearn`. Pomoću metode `fit_transform` razreda `LabelEncoder` vrijednosti i oznake koje nisu u numeričkom obliku pretvaramo u numeričke oznake zbog toga što algoritmi klasifikacije pretpostavljaju da su podatci s kojima rade u numeričkom obliku.

Metodama

```
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

ulazni skup podataka dijelimo na značajke i oznake. Prva naredba će za cijeli skup podataka u varijablu `X` pohraniti sve vrijednosti u uzorcima osim posljednje koja se drugom naredbom sprema u varijablu `y`, tj. u varijablu `X` se spremaju značajke a u varijablu `y` oznake svih uzoraka.

Potom se poziva metoda `fit_resample()` kojoj predajemo varijable sa značajkama i oznakama. Ta metoda će primjeniti SMOTE nad značajkama i oznakama te će vratiti nove skupove koji su naduzorkovani.

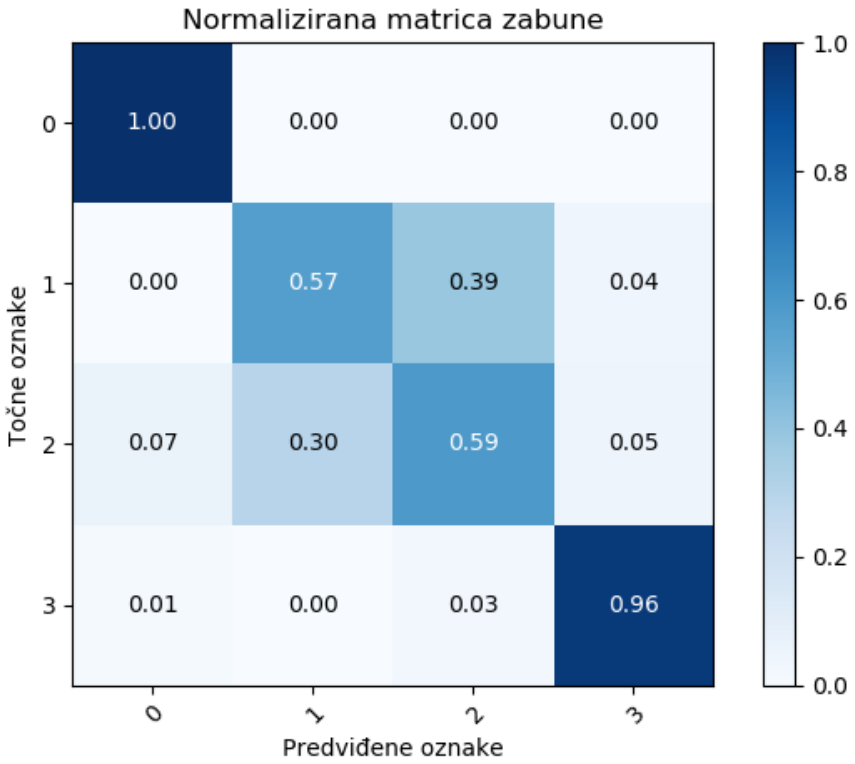
Učenje modela se odvija u funkciji `train()`. Funkcija prima varijable sa značajkama i oznakama. U funkciji se poziva funkcija `train_test_split()` koja dijeli skup podataka na skup za učenje i skup za testiranje. Parametar `test_size` određuje postotak skupa koji će se odvojiti za testiranje, u ovom slučaju omjer broja uzoraka za testiranje i broja uzoraka za učenje je `1/3`. Potom se obavlja inicijalizacija hiperparametara klasifikatora. Metodom `fit()` obavlja se konkretno učenje a metodom `predict()` se nakon učenja modela mogu predvidjeti oznake za novi skup podataka.

Na stranici s rezultatima prikazat će se mjere F1 i matrice zabune klasifikacije prije i nakon primjene postupaka popravljanja točnosti. Matrica zabune je matrica koja grafički prikazuje performansu klasifikatora. Primjer matrice zabune se može vidjeti na slici 3.3 koja prikazuje dio ispisa rezultata aplikacije. Retci matrice prikazuju točne oznake uzoraka a stupci su oznake koje je algoritam klasifikacije predvidio. Na dijagonali matrice se nalaze uzorci koje je algoritam točno klasificirao. Ova matrica

je normalizirana, tj. vrijednosti u svakom retku su podijeljene s brojem pojavljivanja oznake koju taj redak predstavlja.

Točnost klasifikacije nakon primjene postupaka:

F1 mjera klasifikacijom BRF algoritmom: 0.7738391161547539



Slika 3.3: Ispis rezultata aplikacije

Vrijednost po kojoj se uspoređuje učinak postupaka za popravljanje točnosti je mjera F1 koja je u paketu `scikit-learn` definirana ovako:

$$F1 = 2 * (preciznost * odziv)/(preciznost + odziv) \tag{3.1}$$

gdje $odziv$ označava udio točno klasificiranih primjera u skupu svih pozitivnih primjera a $preciznost$ označava udio točno klasificiranih primjera u skupu pozitivno klasificiranih primjera (Bašić i Šnajder, 2011). Mjera F1 je harmonijska sredina preciznosti i odziva, nastoji se postići vrijednost mjere jednaku 1 dok je najlošija vrijednost jednaka 0. Vrijednost mjere je proporcionalna broju točnih klasifikacija. Podaci koji su potrebni za izračun preciznosti i odziva mogu se lako iščitati iz matrice zabune koja nije normalizirana.

Za vrednovanje je odabrana ova mjera zbog toga što je prikladniji pokazatelj od drugih mjera kada se radi s nebalansiranim skupovima podataka. Primjer je mjera klasifikacijske točnosti koja je definirana kao omjer ispravno klasificiranih uzoraka i ukupnog broja uzoraka. Vrijednost klasifikacijske točnosti uvelike ovisi o točno klasificiranim uzorcima većinske klase što nam neće dati puno informacija o kvaliteti klasifikacije manjinske klase. Vrijednost mjere F1 opada vrlo brzo sa porastom broja netočno klasificiranih uzoraka što nam odgovara kada želimo dati veću važnost takvim uzorcima, npr. ako bolesnog pacijenta klasificiramo kao zdravog, šteta netočnog predviđanja će biti velika ako je pacijent zarazan (Shung, 2018).

Funkcija za izračun mjere F1 kao parametre prima točne i predviđene oznake za pojedine uzorke te parametar `average` koji određuje način računanja prosječne vrijednosti F1 mjere kod višeklasne klasifikacije:

```
sklearn.metrics.f1_score(y_true, y_pred, average='weighted')
```

`y_true` su točne oznake, a `y_pred` su oznake koje je algoritam predvidio. Parametar `average` ima više mogućih vrijednosti: `'binary'` ako su zadane oznake binarne, tada se računa mjera klase koja se određuje parametrom `pos_label`, vrijednost `'micro'` računa mjeru pomoću svih vrijednosti dok `'macro'` računa mjeru za svaku oznaku i pronalazi prosječnu mjeru ne uzimajući u obzir različite važnosti mjera, to znači da se ne uzima u obzir nebalansiranost. Vrijednost `'weighted'` označava da će se izračunati F1 mjera svake oznake i zatim pronaći prosječnu vrijednost na temelju važnosti koja se računa pomoću broja točnih klasifikacija svake oznake.

4. Eksperimentalno vrednovanje

Tablica 4.1 prikazuje rezultate klasifikacije prije primjene postupaka i nakon primjene pojedinih postupaka. Vrijednosti označavaju F1 mjere.

Tablica 4.1: Rezultati vrednovanja pojedinih postupaka

Skup	Klasifikator	Originalno	SMOTE	ADASYN	NearMiss	Rand Under
Letter	kNN	0.9854	0.9869	0.9840	0.9714	0.9659
	BRF	0.9729	0.9919	0.9899	0.9571	0.9675
New-thyroid	kNN	0.9596	0.9799	0.9731	0.9333	0.9666
	BRF	0.9737	0.9799	0.9732	0.8997	0.8623
Satimage	kNN	0.8817	**0.9239**	0.8646	0.8234	0.8171
	BRF	0.8629	0.9199	0.8587	0.8232	0.8376
Flag	kNN	0.8638	0.7732	0.7488	0.8381	0.6762
	BRF	0.5644	0.9492	0.9482	0.7556	0.6667
Glass	kNN	0.8296	**0.9466**	0.9388	0.9185	0.4571
	BRF	0.6485	0.9467	0.9466	0.9132	0.6667

Ostale vrijednosti dobivene vrednovanjem prikazane su tablicama 4.2 i 4.3. Te dvije tablice prikazuju F1 mjere nakon primjene svih mogućih kombinacija postupaka za popravljanje točnosti koji se uzimaju u obzir u ovom radu.

Podebljane vrijednosti predstavljaju najbolje mjere za klasifikaciju skupa pojedinim algoritmom. Iz tablica se može vidjeti da postupci u većini slučajeva poboljšavaju točnost klasifikacije i da ona uvelike ovisi o skupu podataka. Također se može primjetiti da korištenje BRF klasifikatora daje bolje rezultate.

Upotreba postupka SMOTE ili kombinacije sva 4 postupka daju najbolje rezultate u najviše slučajeva (po 2 ili 3 puta). Postupci koji najčešće postižu rezultate bolje od klasifikacije bez primjene postupaka su SMOTE te kombinacije SMOTE+ADASYN, SMOTE+Random Under, SMOTE+ADASYN+NearMiss, SMOTE+ADASYN+Random Under, SMOTE+NearMiss+Random Under i kombinacija sva 4 postupka koje to pos-

Tablica 4.2: Rezultati vrednovanja kombinacija postupaka

Skup	Klasifikator	(1)	(2)	(3)	(4)	(5)	(6)
Letter	kNN	0.9876	**0.9881**	0.9878	0.9826	0.9833	0.9779
	BRF	0.9923	0.9909	0.9919	0.9882	0.9886	0.9539
New-thyroid	kNN	0.9933	0.9867	0.9933	**1**	0.9111	0.8667
	BRF	0.9800	0.9666	**0.9933**	0.9714	0.8800	0.9666
Satimage	kNN	0.9235	0.9093	0.9039	0.8584	0.8619	0.8294
	BRF	0.9201	0.9128	0.9187	0.8872	0.8617	0.8374
Flag	kNN	0.7209	0.8042	0.8307	0.7084	0.7446	0.7552
	BRF	0.9320	0.9151	0.9406	0.9213	0.9649	0.7556
Glass	kNN	0.9388	0.9238	0.9238	0.9315	0.9238	0.8381
	BRF	0.8938	0.9393	0.9621	0.9469	**0.9773**	0.6762

(1): SMOTE+ADASYN (2): SMOTE+NearMiss (3): SMOTE+Random Under
(4): ADASYN+NearMiss (5): ADASYN+Random Under (6): NearMiss+Random Under

Tablica 4.3: Rezultati vrednovanja kombinacija postupaka

Skup	Klasifikator	(7)	(8)	(9)	(10)	(11)
Letter	kNN	0.9867	0.9878	0.9873	0.9847	0.9871
	BRF	0.9890	**0.9929**	0.9915	0.9903	0.9927
New-thyroid	kNN	0.9933	0.9667	0.9867	0.9422	0.9867
	BRF	0.9799	0.9867	0.9799	0.9713	**0.9933**
Satimage	kNN	0.9009	0.9007	0.9231	0.8672	0.9136
	BRF	0.9304	0.9225	0.9266	0.8855	**0.9328**
Flag	kNN	0.8378	0.8346	0.8036	0.7237	0.8460
	BRF	0.9321	0.9407	0.9661	0.9126	**0.9746**
Glass	kNN	0.9315	0.9238	0.9001	0.9001	0.9001
	BRF	0.9392	0.9621	0.9621	0.9545	0.9621

(7): SMOTE+ADASYN+NearMiss (8): SMOTE+ADASYN+Random Under
(9): SMOTE+NearMiss+Random Under (10): ADASYN+NearMiss+Random Under
(11): SMOTE+ADASYN+NearMiss+Random Under

tižu ukupno 9 od mogućih 10 puta. Odmah iza njih su kombinacije postupaka SMOTE+NearMiss i ADASYN+NearMiss s 8 i 6 rezultata boljih od originalnih. Ostale kombinacije imaju 5 ili manje boljih rezultata.

5. Zaključak

Na temelju eksperimentalnog vrednovanja došao sam do zaključka da su najučinkovitiji postupci za popravljanje točnosti klasifikacije nad odabranim skupovima podataka: SMOTE te kombinacije SMOTE+ADASYN, SMOTE+Random Under, SMOTE+ADASYN+NearMiss, SMOTE+ADASYN+Random Under, SMOTE+NearMiss+Random Under i kombinacija sva 4 postupka. Iz rezultata vrednovanja se također može zaključiti da uspješnost postupaka uvelike ovisi o skupu podataka što znači da se odabir postupaka mora obaviti vrlo pažljivo. U radu su korištena dva različita klasifikatora, klasifikator k-najbližih susjeda (kNN) i ansambl klasifikatora balansirane slučajne šume (BRF). Usporedbom učinkovitost algoritama klasifikacije, BRF daje puno bolje rezultate kada su u pitanju nebalansirani skupovi podataka.

Tijekom izrade ovog rada upoznao sam se s osnovama strojnog učenja i glavnim problemima koji se javljaju u toj grani umjetne inteligencije. Iako prije početka izrade rada nisam imao nikakvog iskustva sa strojnim učenjem, proučavanjem literature i istraživanjem sam riješio sve zapreke koje su se javljale. Entuzijastično iščekujem proširiti znanje koje sam stekao izradom ovog rada na kolegijima diplomskog studija.

LITERATURA

Bojana Dalbelo Bašić i Jan Šnajder. Vrednovanje klasifikatora, 2011. URL https://www.fer.unizg.hr/_download/repository/SU-12-VrednovanjeKlasifikatora.pdf.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, i W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16, 2002. URL https://jair.org/index.php/jair/article/view/10302.

Chao Chen i Leo Breiman. Using random forest to learn imbalanced data. *University of California, Berkeley*, 01 2004.

Dheeru Dua i Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Haibo He, Yang Bai, Edwardo A. Garcia, i Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. U *IEEE INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS (IEEE WORLD CONGRESS ON COMPUTATIONAL INTELLIGENCE), IJCNN 2008*, stranice 1322–1328, 2008.

Guillaume Lemaître, Fernando Nogueira, i Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017. URL http://jmlr.org/papers/v18/16-365.html.

Sebastian Raschka. *Python Machine Learning*. Packt Publishing Ltd., 2015.

Koo Ping Shung. Accuracy, precision, recall or f1?, 2018. URL https://towardsdatascience.com/accuracy-precision-recall-or-f1-331fb37c5cb9.

J. Zhang i I. Mani. KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction. U *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets*, 2003.

Postupci strojnog učenja za popravljanje točnosti klasifikacije manjinskih klasa kod nebalansiranih skupova podataka

Sažetak

Ovaj rad uspoređuje utjecaj postupaka za popravljanje točnosti i njihovih kombinacija na klasifikaciju manjinskih klasa kod nebalansiranih skupova podataka. Za potrebe eksperimentalnog vrednovanja razvijena je web aplikacija. Na temelju rezultata eksperimentalnog vrednovanja donesen je zaključak da odabir postupaka za popravljanje točnosti ovisi o karakteristikama skupa nad kojim se obavlja klasifikacija. Postupci ponovnog uzorkovanja koji su postigli najbolje rezultate nad skupovima podataka odabranima za vrednovanje su: SMOTE te kombinacije SMOTE+ADASYN, SMOTE+Random Under, SMOTE+ADASYN+NearMiss, SMOTE+ADASYN+Random Under, SMOTE+NearMiss+Random Under i kombinacija sva 4 postupka. Osim postupaka ponovnog uzorkovanja uspoređena su dva algoritma klasifikacije, rezultati pokazuju da je za klasifikaciju nebalansiranih skupova podataka ansambl klasifikatora balansirane slučajne šume prikladniji od algoritma k-najbližih susjeda.

Ključne riječi: poduzorkovanje, naduzorkovanje, ansambl klasifikatora, matrica zabune, F1 mjera

Machine learning methods for improving classification accuracy of minority classes on unbalanced datasets

Abstract

This paper compares the influence of methods and combinations of methods for improving classification accuracy of minority classes on unbalanced datasets. Web application has been developed for the purpose of experimental evaluation. Based on the results of the experimental evaluation, it was concluded that the choice of methods depends heavily on the characteristics of datasets on which the classification was performed. Resampling methods that achieved the best results are: SMOTE and combinations SMOTE+ADASYN, SMOTE+Random Under, SMOTE+ADASYN+NearMiss, SMOTE+ADASYN+Random Under, SMOTE+NearMiss+Random Under and the combination of all 4 methods. In addition to resampling methods, the paper compares two classification algorithms. The results show that Balanced Random Forest ensemble is better suited for classification of unbalanced datasets than k Nearest Neighbors classifier.

Keywords: undersampling, oversampling, ensemble of classifiers, confusion matrix, F1 score