UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Domen Košir

Usmerjeno generiranje sintetičnih učnih primerov v strojnem učenju na podlagi ocen zanesljivosti klasifikacij

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

Mentor: doc. dr. Zoran Bosnić

Ljubljana, 2010

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01692/2010
Datum: 01.09.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **DOMEN KOŠIR**

Naslov: **USMERJENO GENERIRANJE SINTETIČNIH UČNIH PRIMEROV V STROJNEM UČENJU NA PODLAGI OCEN ZANESLJIVOSTI KLASIFIKACIJ**

DIRECTED GENERATION OF SYNTHETIC EXAMPLES IN MACHINE LEARNING BASED ON THE CLASSIFICATION RELIABILITY ESTIMATES

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Pri napovedovanju z metodami strojnega učenja se srečujemo s situacijami, kadar na uspešnost klasifikatorjev močno vplivajo razlike v zastopanosti posameznih razredov v primerih v učni množici. Realni scenariji takšnih problemov so npr. v medicini, biologiji, finančnih aplikacijah in drugje, kjer je kritičnega pomena napovedati redke pojave (bolezen, finančna nedisciplina itd.), ki so med opazovanimi primeri le malokrat zastopani.

V diplomski nalogi naj kandidat predstavi različne pogoste pristope za reševanje težav z neuravnoteženimi podatki z uporabo metod generiranja sintetičnih primerov. V svojo primerjvo naj vključi tudi razširjeni algoritem SMOTE, ki naj ga poizkusi izboljšati s selektivnim generiranjem sintetičnih učnih primerov, in sicer na podlagi rezultatov metode za ocenjevanje zanesljivosti klasifikacij. V diplomi naj primerja uspešnosti obstoječih in predlaganih algoritmov ter naj jih ovrednoti.

Mentor:

doc. dr. Zoran Bosnić

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Franc Solina

Namesto te strani **vstavite** original izdane teme diplomskega dela s podpisom mentorja in dekana ter žigom fakultete, ki ga diplomant dvigne v študentskem referatu, preden odda izdelek v vezavo!

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani: Domen Košir,

z vpisno številko: 63990241,

sem avtor diplomskega dela z naslovom:

Usmerjeno generiranje sintetičnih učnih primerov v strojnem učenju na podlagi ocen zanesljivosti klasifikacij

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelala samostojno pod mentorstvom doc. dr. Zorana Bosnića,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 1.9.2010 Podpis avtorja:

Zahvala

Najprej bi se rad zahvalil svojemu mentorju, doc. dr. Zoranu Bosniću, za strokovno svetovanje in vodenje po poti izdelave tega diplomskega dela. Še posebej sem mu hvaležen za optimizem in potrpežljivost, ki sem ju bil deležen.

Mojima staršema gre zahvala za prirojeno/privzgojeno trmo, delavnost in željo po znanju ter za leta podpore in spodbujanja pri izobraževanju.

Bratu in prijateljem pa bi se rad zahvalil za moralno oporo in vse dragocene trenutke sprostitve.

Kazalo

Seznam uporabljenih kratic in simbolov

- AUC - Area Under the Curve
- ROC - Receiver Operating Characteristic
- SMOTE - Synthetic Minority Over-sampling TEchnique
- SMOTER - Synthetic Minority Over-sampling TEchnique using Reliable examples
- SMOTERAND - Synthetic Minority Over-sampling TEchnique using RANDom examples

Povzetek

V zadnjem času, ko se v strojnem učenju vedno več dela z realnimi podatkovnimi množicami, se opaža problem, ko se nekateri klasifikatorji slabo učijo iz neuravnoteženih podatkovnih množic. Podatki iz realnega sveta so pogosto sestavljeni iz veliko "normalnih" primerov in le nekaj "zanimivih" primerov. Tipični primeri so na primer iskanje goljufivih plačil s kreditnimi karticami, detekcija in diagnoza obolelih tkiv ter detekcija sumljivega vedenja ljudi v video posnetkih. Poleg omenjenih primerov, ko je neuravnoteženost primerov "naravna", imamo lahko opravka z neuravnotežimi podatkovnimi množicami tudi zaradi omejenega dostopa do podatkov (temu lahko botrujejo ekonomski ali zasebnostni razlogi).

Nekateri izmed najpogosteje uporabljenih klasifikatorjev so občutljivi na neuravnotežene podatke. Ko je delež manjšinskih primerov v učni množici zelo majhen, lahko klasifikator začne zanemarjati manjšinski razred, kar vodi do manjše klasifikacijske točnosti.

Rešitev za to situacijo je več, tako na nivoju podatkov, kot na nivoju uporabljenih klasifikacijskih algoritmov. Mi smo se osredotočili za uravnoteženje podatkov s pomočjo algoritmov za uravnoteženje podatkovnih množic: naključno podvzorčenje, naključno nadvzorčenje in SMOTE [4]. Poleg omenjenih smo za potrebe te diplome implementirali še tri variacije algoritma SMOTE, ki pri svojem delovanju uporabljajo oceno zanesljivosti klasifikacij primerov [5, 6].

Algoritme za uravnoteženje podatkov smo z uporabo štirih klasifikatorjev (odločitvena drevesa, naivni Bayes, metoda najbližjih sosedov in metoda podpornih vektorjev) med seboj primerjali s pomočjo 10-kratnega prečnega preverjanja na 10 podatkovnih množicah iz repozitorija UCI Machine Learning Repository [19].

Ugotovili smo, da lahko z uporabo naših variacij algoritma SMOTE v večini primerov pridemo do večjih klasifikacijskih točnosti kot z uporabo ostalih omenjenih algoritmov za uravnoteženje podatkov. Do največjih izboljšanj klasifikacijskih točnosti smo prišli z uravnoteženjem majhnih podatkovnih množic z nizkim deležem manjšinskih primerov.

Ključne besede: strojno učenje, neuravnoteženi podatki, naključno podvzorčenje, naključno nadvzorčenje, SMOTE, SMOTER ASC, SMOTER DESC, SMOTERAND

Abstract

The field of machine learning has made a lot of progress in recent years. As it is used more frequently in real-world problems a new issue has emerged. Studies have shown that imbalanced data can lead to poor performance by some classificators. Imbalanced datasets are composed of many "normal" examples and few "interesting" ones. Typical examples are credit card fraud detection, detection and diagnosis of diseases in tissue samples and detection of suspicious behaviour in surveillance camera videos. The imbalance in data can be "natural" or we can have imbalanced data due to economic or privacy reasons.

When presented with highly imbalanced data, some standard classifiers can ignore the minority class which leads to lower classification accuracy.

Various solutions have been proposed to counter this problem. Some solutions include modifications of classification algorithms while other solutions modify the data itself. In this thesis, we focus onto the latter. Random undersampling, random oversampling and SMOTE (Synthetic Minority Oversampling TEchnique) have been implemented and tested. In addition, three new varations of SMOTE algorithm have been proposed in this thesis. All three estimate classification reliability (Kukar et al.) of minority examples and then use these estimates while generating synthetic examples.

The data balancing algorithms were tested with 10-fold cross validation using 10 datasets from the UCI Machine Learning Repository and four different classifiers (decision trees, naive Bayes, k-nearest neighbors algorithm and support vector machines).

The results have shown that it is feasible to improve classifiers' performance by balancing the data with one of our versions of SMOTE algorithm. The most significant improvements in classification accuracy were observed when we balanced small datasets with low shares of minority examples.

Key words: machine learning, imbalanced data, imbalanced datasets, ran-

dom undersampling, random oversampling, SMOTE

Poglavje 1

Uvod in motivacija

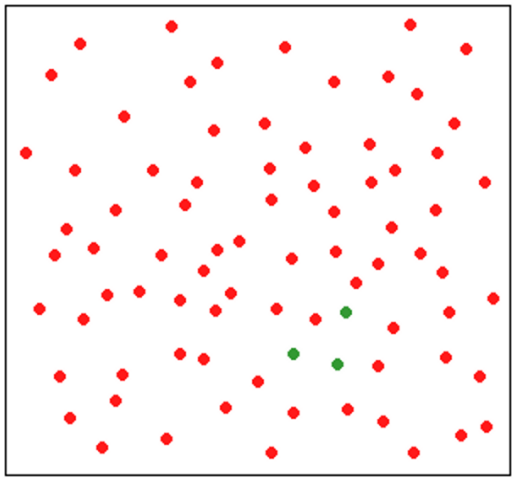
Strojno učenje je v zadnjih letih naredilo preskok iz večinoma raziskovalne veje znanosti v vedo, ki se vedno pogosteje uporablja v praksi. V zadnjem času smo vedno pogosteje v situaciji, ko imamo za učenje na razpolago relativno malo primerov iz enega od razredov. To se še posebej rado dogaja, ko delamo s podatki, pridobljenimi iz nekaterih domen iz realnega sveta, na primer področja zdravstva in iskanja prevar. Pridobivanje novih podatkov je velikokrat težavno, ali celo nemogoče, zato je potrebno delati takimi podatki, kot so nam na voljo.

Neuravnotežena podatkovna množica (primer neuravnotežene podatkovne množice je ilustriran na sliki 1.1) sama po sebi ne predstavlja nobenega problema. Problem predstavljajo nekateri klasifikatorji, ki se lahko iz takih podatkov slabo učijo. Taka sta na primer naivni Bayes in odločitvena drevesa [1]. Oba omenjena algoritma sta v strojnem učenju pogosto uporabljena in pri učenju iz neuravnoteženih podatkov dajeta občutno slabše rezultate kot nekateri klasifikatorji, ki na neuravnoteženost učnih podatkov niso občutljivi. Na nauravnoteženost naj bi bila neobčutljivi na primer metoda podpornih vektorjev (SVM) [1] in metoda k najbližjih sosedov (kNN) [1].

Če je klasifikator občutljiv na uravnoteženost učnih podatkov, lahko to za nas predstavlja velik problem. Če na primer 99% primerov pripada večinskemu razredu, se velikokrat bolje kot odločitveno drevo obnese preprost klasifikator, ki vse primere uvrsti v večinski razred in ima posledično 99% klasifikacijsko točnost. Tak, preprost klasifikator, pa kljub visoki klasifikacijski točnosti ni koristen, saj zanemari cel manjšinski razred.

Rešitev za problem neuravnoteženosti podatkov je več, tako na podatkovnem, kot na algoritmičnem nivoju. Na nivoju algoritmov lahko:

- damo primerom iz manjšinskega razreda večjo veljavo/težo (tako lahko



Slika 1.1: Primer neuravnotežene podatkovne množice. Pripadniki večinskega razreda so rdeče barve in pripadniki manjšinskega pa zelene.

vsako napačno klasifikacijo pripadnika manjšinskega razreda kaznujemo huje, kot napačno klasifikacijo večinskega primera),

- namesto klasifikatorja, ki poskuša razlikovati med razredi, uporabimo več klasifikatorjev, med katerimi vsak detektira pripadnike posameznega razreda.

Na podatkovnem nivoju so nekatere izmed znanih rešitev:

- naključno brisanje primerov iz podatkovne množice,
- selektivno brisanje primerov,
- naključno podvajanje primerov,
- selektivno podvajanje primerov,
- spreminjanje teže posameznih primerov in
- generiranje novih primerov iz že obstoječih.

Nas bo najbolj zanimala zadnja rešitev. Najbolj znan algoritem za generiranje novih primerov je SMOTE [4]. Ta algoritem uporablja pripadnike manjšinskega razreda in iz njih generira nove manjšinske primere. Delovanje algoritma SMOTE je bolj podrobno opisano v 2.2.3 Za uravnoteženje podatkovne množice pa lahko uporabimo tudi kombinacije prijemov:

- SMOTE z naključnim podvzorčenjem: ob testiranju algoritma SMOTE [4] so med drugim eksperimentirali tudi s kombinacijo algoritma SMOTE in naključnega podvzorčenja večinskega razreda. Z algoritmom SMOTE so povečevali število manjšinskih primerov, z naključnim podvzorčenjem pa zmanjševali število primerov v večinskem razredu. Tako so obrnili razmerje med številom primerov v manjšinskem in večinskem razredu. Ugotovili so, da je ta kombinacija boljša (klasifikatorji, ki so naučeni iz teh podatkov, delajo pri klasifikaciji primerov manj napak) od naključnega podvzorčenja. Rezultati pa se niso kaj dosti razlikovali od rezultatov testiranj algoritma SMOTE.

- SMOTEBoost [13] je kombinacija algoritmov SMOTE in AdaBoost. Ta na začetku dodeli vsem napačno klasificiranim primerom enake uteži, potem pa povečuje in zmanjšuje število na novo generiranih primerov. To ponavlja in vsakič se podatkovna množica uporabi za učenje odločitvenega drevesa. Algoritem konča s svojim delovanjem, ko odločitveno drevo naredi najmanj napak pri klasifikaciji primerov iz obeh razredov.

Cilj te diplome je usmerjeno generiranje novih primerov. Originalna oblika algoritma SMOTE za generiranje novih primerov uporablja vse manjšinske primere, brez kakršnih koli preferenc. Mi bomo algoritem SMOTE priredili in naključno izbiro manjšinskega primera, ki bo uporabljen za generiranje novega primera, nadomestili z deterministično. Na tem mestu bomo za primerjavo med manjšinskimi primeri uporabili oceno zanesljivosti klasifikacije primerov [5]. Slednja ocena zanesljivosti klasifikacije za izbrani primer je definirana kot evklidsko razdaljo med dvema porazdelitvama po razredih, od katerih je prva porazdelitev napovedana od začetnega klasifikatorja, druga pa od klasifikatorja na učni množici z dodanim učnim primerom, ki ga klasificiramo. Ocenjevanje zanesljivosti klasifikacije primerov je bolj podrobno opisano v poglavju 3.1).

Za potrebe diplome bomo implementirali tri že obstoječe algoritme za uravnoteženje podatkovnih množic:

- naključno podvzorčenje,
- naključno nadvzorčenje in
- SMOTE.

Algoritem SMOTE bomo priredili in implementirali bomo še tri variante tega algoritma, ki pri izboru manjšinskih primerov, iz katerih se generirajo novi primeri, na različne načine upoštevajo oceno zanesljivosti klasifikacij primerov. Te algoritme smo poimenovali:

- SMOTER ASC,
- SMOTER DESC in
- SMOTERAND.

Pričakujemo, da se bo izkazalo, da se bodo lahko testirani klasifikatorji bolje učili na uravnoteženih podatkovnih množicah s predlaganimi algoritmi in da bodo posledično pri klasifikaciji primerov delali manj napak.

Zaradi enostavnosti se bomo v tej diplomi omejili le na dvorazredne podatkovne množice.

Vse algoritme za predobdelavo podatkovnih množic in pomožne skripte, ki so se večinoma uporabljale za testiranje algoritmov, smo implementirali v okolju R. Za R smo se odločili predvsem zaradi primernosti tega programskega jezika, proste dostopnosti izvornih kod in veliko uporabnih, že implementiranih funkcij.

Poglavje 2

Učenje iz neuravnoteženih podatkov

Raziskave so pokazale, da so neuravnoteženi učni podatki pogosto vzrok za zmanjšanje klasifikacijske točnosti klasifikatorjev. Veliko standardnih klasifikacijskih algoritmov namreč predvideva, da so primeri v učni množici enakomerno porazdeljeni po razredih. V praksi temu ni tako. Velikokrat se zgodi, da v učnih množicah iz realnih domen nekemu razredu pripada le malo primerov. Cilj klasifikacije pa je ponavadi točno napovedovanje tudi teh (manjšinskih) primerov, ker so lahko ravno ti primeri predmet študije ali opazovanja. Praktični problemi so na primer iskanje obolelih tkiv, iskanje goljufivih transakcij v bančništvu, detekcija sumljivega obnašanja iz posnetkov video kamer, ...

Že leta 1997 se je Miroslav Kubat s sodelavci spopadel z izzivom detekcije naftnih razlitij v radarskih slikah morske gladine [15]. Naftna razlitja so se pojavljala na le 4% slik. V članku so primerjali algoritma 1-NN in SHRINK. Algoritem 1-NN klasificira primer na podlagi razreda njegovega najbližjega soseda in se v njihovih testiranjih ni dobro obnesel. Večjo točnost je pokazal algoritem SHRINK, ki pri učenju generira le pravila za detekcijo manjšinskega razreda in je zato neobčutljiv na neuravnoteženost učnih podatkov.

Leta 2000 je Foster Provost [16] z univerze v New Yorku predaval na delavnici o strojnem učenju iz neuravnoteženih podatkov. Po njegovem mnenju ima problem učenja iz neuravnoteženih podatkovnih množic korenine v nerazumevanju takih podatkov. Nekateri klasifikatorji predvidevajo, da bo število primerov enakomerno porazdeljeno po razredih in so zato neuporabni. Spet drugi klasifikacijski algoritmi predvidevajo, da bo razmerje med številom večinskih in manjšinskih primerov enako v učni in v testni množici. Tudi ta predpo-

stavka je lahko vzrok za manjšo klasifikacijsko točnost. Na koncu zaključuje, da je za uspešnost učenja na neuravnoteženih podatkih potrebno predvsem razumeti obnašanje klasifikacijskih algoritmov, ki se uporabljajo.

S problemom detekcije goljufij se je srečal J. Zhang s sodelavci [18]. Za potrebe policije so razvili algoritem RLSD (Rule Learning for Skewed Data), ki je dal vzpodbudne rezultate tudi, ko je bil delež manjšinskih primerov v učni množici le 0.01%.

Leta 2008 sta David A. Cieslak in Nitesh V. Chawla [8] na konferenci European Conference on Machine Learning and Knowledge Discovery in Databases predstavila rezultate svojih testiranj. Med seboj sta primerjala štiri klasifikacijske algoritme, ki vsi temeljijo na odločitvenih drevesih: HDDT (Hellinger Distance Decision Tree [8]), DKM [10], C4.5 [9] in CART (Classification And Regression Tree [11]). Vse štiri klasifikatorje sta najprej učila na neuravnoteženih podatkih in ugotovila, da sta HDDT in DKM veliko bolj točna pri klasifikaciji primerov. Ko sta učno množico s pomočjo algoritma SMOTE uravnotežila, sta bila HDDT in DKM še vedno bolj uspešna pri klasifikaciji primerov. Zanimivo pa je, da se jima je klasifikacijska točnost znižala. Na drugi strani pa se je klasifikatorjema C4.5 in CART klasifikacijska točnost zvišala.

2.1 Neuravnoteženost učnih množic

Večinski in manjšinski primeri. Ker se ukvarjamo le z neuravnoteženimi podatkovnimi množicami, bo število razredov neenakomerno porazdeljeno po razredih. Večkrat zastopanemu razredu pravimo večinski razred, manj zastopanemu pa manjšinski. Primeri, ki pripadajo večinskemu razredu, so večinski primeri, pripadniki manjšinskega pa manjšinski primeri.

V tej diplomi bomo spreminjali število primerov v obeh razredih - večinski razred bo po obdelavi podatkovne množice velikokrat postal manjšinski in obratno. Da bi bilo čim manj nejasnosti, bomo v tej diplomi uporabljali izraza "večinski primeri" in "manjšinski primeri" za pripadnike istega razreda tako pred, kot tudi po obdelavi podatkovnih množic, čeprav se zna zgoditi, da bo imel večinski razred po obdelavi v resnici manj primerov, kot manjšinski.

Želena stopnja uravnoteženosti S pojmom "želena stopnja uravnoteženosti" označujemo želeno razmerje med številom manjšinskih in večinskih primerov po obdelavi podatkovne množice.

Stopnja 200% na primer pomeni, da naj bo po obdelavi manjšinskih primerov dvakrat toliko, kot večinskih. Stopnja 100% pa pomeni, da naj bo po

obdelavi manjšinskih primerov natanko toliko, kot večinskih.

2.2 Generiranje sintetičnih primerov

Zakaj se sploh ukvarjamo z generiranjem sintetičnih primerov? Za izenačitev števila primerov v vseh razredih bi bilo dovolj, da bi primere iz manjšinskih razredov le podvajali dokler ne bi dobili dovolj enakega števila primerov v vseh razredih. Treba je vedeti, da naš cilj ni uravnotežena podatkovna množica. Ta je le sredstvo, s katerim lahko pridobimo večjo točnost klasifikatorjev, ki se učijo na teh podatkih. Cilj v strojnem učenju je doseči čim večjo klasifikacijsko točnost in zato vedno iščemo postopke, s katerimi bi jo lahko še izboljšali.

Algoritem SMOTE je bil razvit s točno tem namenom - izboljšati klasifikacijsko točnost klasifikatorjev, ki jih učimo na neuravnoteženih podatkih. V raziskavah se je pokazalo, da je SMOTE pri tem uspešen. V tem poglavju bomo opisali tri že obstoječe algoritme za uravnoteženje podatkovnih množic, ki smo jih implementirali in nekoliko prilagodili za potrebe tega diplomskega dela: naključno podvzorčenje, naključno nadvzorčenje in SMOTE.

2.2.1 Naključno podvzorčenje

Naključno podvzorčenje (angl. random undersampling) je edini od tu uporabljenih algoritmov, ki ne povečuje števila manjšinskih primerov. Namesto tega zmanjšuje število primerov v večinskem razredu. Kot se da sklepati že iz imena, algoritem pri brisanju pripadnikov večinskega razreda nima nikakršnih preferenc in jih briše povsem naključno (glej Algoritem 1).

Na prvi pogled se zdi, da bo brisanje primerov iz učne množice negativno vplivalo na uspešnost klasifikatorja, vendar pa se je ta algoritem v preteklosti izkazal za uporabnega pri velikih podatkovnih množicah.

Algoritem 1 Psevdokoda algoritma "naključno podvzorčenje"

```
while število primerov v večinskem razredu je preveliko do
    izberi naključen primer iz večinskega razreda
    zbriši izbrani primer iz podatkovne množice
end while
```

2.2.2 Naključno nadvzorčenje

Naključno nadvzorčenje (angl. random oversampling) deluje ravno obratno kot prej opisani algoritem. Število manjšinskih primerov povečuje tako, da si v vsakem koraku izbere naključen primer iz manjšinskega razreda in ga podvoji. Kljub temu, da ta algoritem ne generira novih primerov, ki bi se od obstoječih razlikovali, je glede na svojo enostavnost in računsko nezahtevnost lahko zelo koristen. Psevdokoda algoritma je opisana kot Algoritem 2.

Rezultat naključnega nadvzorčenja je praktično enak, kot če bi v originalni podatkovni množici naključne pripadnike manjšinskega razreda dodatno utežili.

Algoritem 2 Psevdokoda algoritma "naključno nadvzorčenje"

```
while število primerov v manjšinskem razredu je premajhno do
    izberi naključen primer iz manjšinskega razreda
    dodaj kopijo izbranega primera med dodatne primere
end while
združi originalno podatkovno množico in dodatne primere
```

2.2.3 SMOTE

Algoritem SMOTE je delo avtorjev Chawla et al. [4]. SMOTE (glej Algoritem 4) deluje tako, da povečuje število manjšinskih primerov, dokler ne zgenerira dovolj novih primerov. Nove primere generira enega po enega. Sprehaja se po pripadnikih manjšinskega razreda in v vsakem koraku iz trenutnega manjšinskega primera X in enega (naključnega) od njegovih najbližjih sosedov Y s pomočjo naključne uteži w (realno število med 0 in 1) zgenerira nov sintetičen primer. Za računanje bližin med primeri smo uporabili kar evklidske razdalje. Iskanje najbližjih sosedov je ilustrirano na sliki 2.1.

Vrednosti zveznih atributov novega primera Z izračunamo z naslednjim postopkom:

Zgornja formula je neuporabna za računanje vrednosti diskretnih atributov. Zato vrednostim diskretnih atributov za nov primer pripišemo najpogostejšo vrednost tega atributa med njegovimi najbližjimi sosedi.

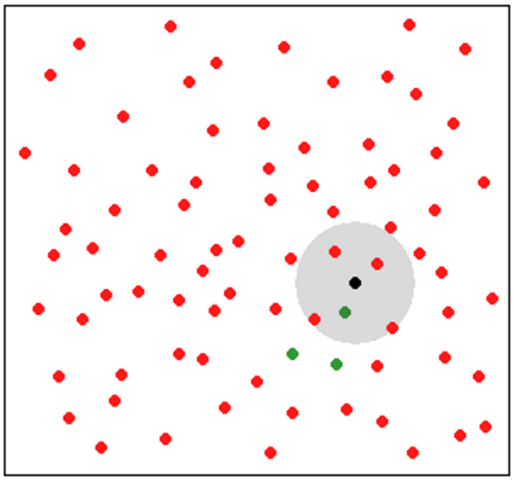
Če si predstavljamo prostor, v katerem se nahajajo primeri, kot dvodimenzionalen prostor (ravnino), se sintetični primeri, zgenerirani z algoritmom SMOTE vedno nahajajo na premici med manjšinskim primerom in enim od njegovih sosedov (glej sliko 2.2). Kje na premici leži nov primer, določa vrednost uteži w, ki je med 0 in 1.

Algoritem 3 Postopek izračuna vrednosti zveznih atributov sintetičnih primerov pri algoritmu SMOTE

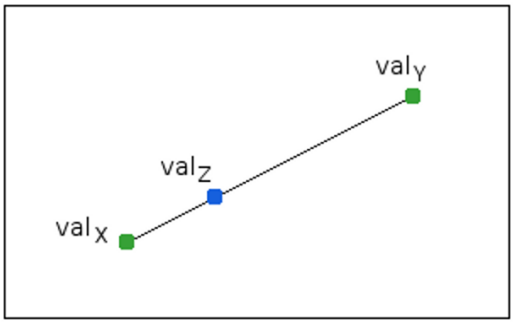
X je trenutno izbrani manjšinski primer
Y je naključno izbran sosed primera X
w je vrednost uteži, ki se bo uporabila za generiranje sintetičnega primera, $w \in \Re, w \in [0, 1]$
Z je sintetičen primer, za katerega računamo vrednost atributa val_Z
vrednost atributa val_Z izračunamo po formuli: $val_Z = val_X + (val_Y - val_X) * w$

Algoritem 4 Psevdokoda algoritma SMOTE

for vsak manjšinski primer **do**
  poišči njegove najbližje sosede, ki pripadajo istemu razredu
end for
while število primerov v manjšinskem razredu je premajhno **do**
  izberi naslednji manjšinski primer
  naključno izberi enega od njegovih najbližjih sosedov
  izberi naključno utež w
  ustvari nov, sintetičen primer
  dodaj nov primer med dodatne primere
end while
združi originalno podatkovno množico in dodatne primere



Slika 2.1: Primer iskanja 5 najbližjih sosedov. Med sosede črno obarvanega pripadnika rdečega razreda štejemo vseh 5 bližnjih rdečih primerov, ki se nahajajo znotraj sivega območja. Zelenega primera ne smatramo za soseda, ker pripada drugemu razredu.



Slika 2.2: Dva zelena primerav dvodimenzionalnem prostoru in sintetičen, moder primer, zgeneriran z algoritmom SMOTE

Poglavje 3

Sintetični primeri na podlagi zanesljivosti

V tem poglavju bomo predstavili dva nova pristopa h generiranju sintetičnih primerov. Oba nova algoritma sta izpeljanki algoritma SMOTE. V glavni zanki algoritma SMOTE (glej Algoritem 4) se v vsakem koraku zgenerira en sintetičen primer. Kot osnovo za ta sintetičen primer je potrebno najprej izbrati enega pripadnika manjšinskega razreda (ter njegovega soseda in utež). Pri tem izboru algoritem SMOTE nima nobenih preferenc in v vsakem koraku izbere le "naslednji" manjšinski primer. Vrstni red manjšinskih primerov tu ni pomemben, uporabljeni bodo po vrsti, eden za drugim. Naša dva pristopa sta izboljšavi tega dela algoritma SMOTE. Namesto slepega izbiranja manjšinskih primerov bomo za vsak primer najprej ocenili zanesljivost njegove klasifikacije in to oceno potem uporabili pri izboru manjšinskih primerov, ki se bodo uporabili za osnovo pri generiranju sintetičnih primerov. Pričakujemo, da bo generiranje novih primerov na podlagi ocen zanesljivosti klasifikacij klasifikatorju pomagalo povečati točnost v delih problemskega prostora, kjer prihaja do največje napake.

V razdelku 3.1 bomo najprej predstavili metodologijo ocenjevanja zanesljivosti klasifikacij primerov, v razdelkih 3.2 in 3.3 pa nova algoritma, ki to oceno uporabljata pri svojem delovanju.

3.1 Ocenjevanje zanesljivosti klasifikacij

Za potrebe novih pristopov generiranja sintetičnih primerov (imenujemo ju SMOTER in SMOTERAND) moramo za potrebe selektivnega generiranja primerov najprej izračunati zanesljivost klasifikacij le-teh. Zanesljivost klasifikacij

primerov bomo ocenjevali s postopkom [5, 6], ki je prikazan kot Algoritem 5.

Algoritem 5 Računanje ocene zanesljivosti klasifikacije primerov

for vsak primer **do**
klasifikator naučimo na učni množici in klasificiramo izbrani primer
dobimo prvo porazdelitev napovedi po razredih $[a_x, a_y]$
učni množici dodamo še primer, ki ga napovedujemo (označimo ga z najbolj verjetnim razredom iz prejšnjega koraka) in ponovimo klasifikacijo
dobimo drugo porazdelitev po razredih $[b_x, b_y]$
zanesljivost klasifikacije primera izračunamo kot evklidsko razdaljo med obema porazdelitvama po razredih: $\sqrt{(a_x - b_x)^2 + (a_y - b_y)^2}$
end for

Zanesljivost klasifikacije določenega primera nam pove, koliko lahko zaupamo uvrstitvi tega primera v njegov razred. Z upoštevanjem zanesljivosti klasifikacije primerov upamo, da bomo lahko pri generiranju novih primerov z algoritmoma SMOTER in SMOTERAND prišli do boljših rezultatov, kot pri algoritmu SMOTE.

Pri izbiri klasifikatorja, ki bo uporabljen pri ocenjevanju zanesljivosti klasifikacije primerov smo pazili, da je bil vedno uporabljen enak klasifikator, kot kasneje pri prečnem preverjanju.

3.2 SMOTER

Algoritem SMOTER (glej Algoritem 6) je izpeljanka algoritma SMOTE. Pri originalnem algoritmu SMOTE se za generiranje sintetičnih primerov uporabljajo manjšinski primeri v enakem vrstnem redu, kot nastopajo v originalni učni množici. Spremenjen algoritem SMOTER se od svojega "prednika" razlikuje po tem, da manjšinske primere pred začetkom generiranja novih primerov razvrsti po zanesljivosti (naraščajoče ali padajoče). Tako razvrščeni primeri se potem eden za drugim uporabljajo za generiranje novih primerov.

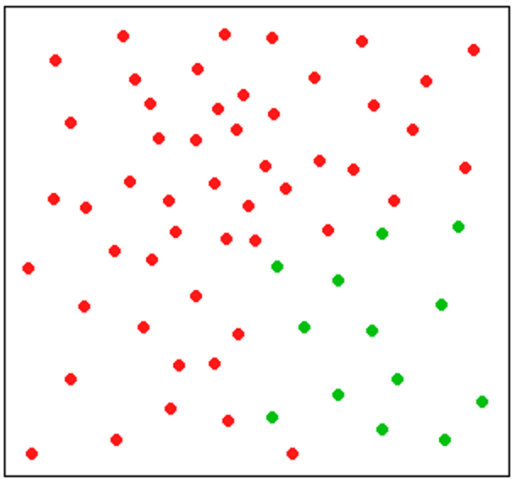
Implementirali smo dve različici algoritma SMOTER, od katerih ima vsaka svojo intuitivno motivacijo:

- SMOTER ASC - ta uredi manjšinske primere po oceni zanesljivosti klasifikacij od najmanj zanesljivega do najbolj zanesljivega (angl. ascending). Ta algoritem večjo pozornost posveča primerom, za katere smo manj sigurni, da spadajo v njihov razred (manj zanesljive). Novi primeri se

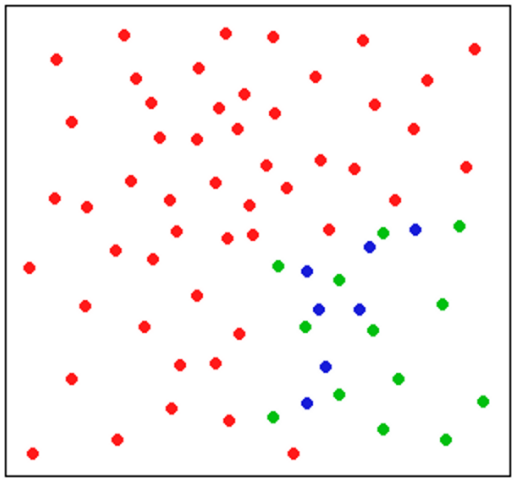
zato večkrat nahajo ob meji med obema razredoma in lahko rečemo, da "krepijo mejo med razredoma".

- SMOTER DESC - ta uredi primere po oceni zanesljivosti klasifikacij od najbolj zanesljivega do najmanj (angl. descending). SMOTER DESC za generiranje novih primerov najprej uporabi primere iz originalne množice, za katere smo bolj sigurni v njihovo klasifikacijo (bolj zanesljive). Posledica tega obnašanja algoritma je, da lahko v njihovo uvrščenost v razrede bolj zaupamo. Lahko bi rekli, da ta algoritem "okrepi jedro manjšinskega razreda".

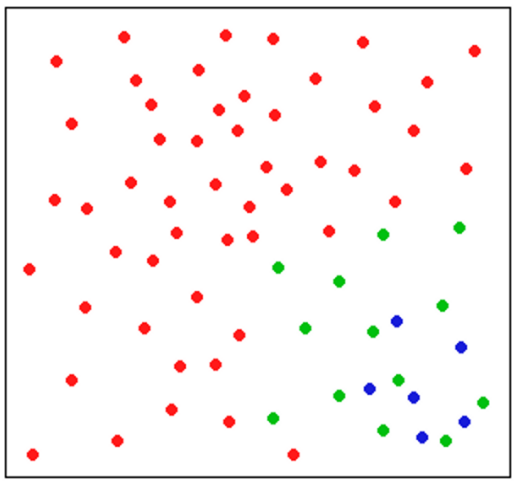
V pomoč pri predstavi so vam lahko slike 3.1-3.3. Slika 3.1 simbolično predstavlja primer dvorazredne podatkovne množice. Na sliki 3.2 je poleg vseh primerov iz originalne podatkovne množice še nekaj novih, sintetičnih primerov, ki so obarvani modro. Ker so zgenerirani iz nezanesljivih manjšinskih primerov (kot pri algoritmu SMOTE ASC), so tudi sintetični primeri relativno nezanesljivi in se večinoma nahajajo blizu meje med razredoma. Algoritem SMOTER DESC pa po drugi strani generira nove primere predvsem iz zanesljivih manjšinskih primerov. Ti se ponavadi ne nahajajo blizu meje med razredoma. Nekaj sintetičnih primerov, zgeneriranih iz zanesljivih primerov, lahko vidimo na sliki 3.3.



Slika 3.1: Dvorazredna podatkovna množica z večinskim rdečim in manjšinskim zelenim razredom, brez sintetičnih primerov



Slika 3.2: Dvorazredna podatkovna množica z večinskim rdečim in manjšinskim zelenim razredom, z nekaj sintetičnimi (modrimi) primeri, generirani iz nezanesljivih manjšinskih primerov (SMOTE ASC)



Slika 3.3: Dvorazredna podatkovna množica z večinskim rdečim in manjšinskim zelenim razredom, z nekaj sintetičnimi (modrimi) primeri, generirani iz zanesljivih manjšinskih primerov (SMOTE DESC)

Algoritem 6 Psevdo algoritma SMOTER (različici ASC in DESC)

for vsak manjšinski primer **do**
  izračunaj zanesljivost tega primera
end for
for vsak manjšinski primer **do**
  poišči njegove najbližje sosede, ki pripadajo istemu razredu
end for
uredi manjšinske primere po zanesljivosti (naraščajoče - ASC ali padajoče - DESC)
while število primerov v manjšinskem razredu je premajhno **do**
  izberi naslednji manjšinski primer
  naključno izberi enega od njegovih najbližjih sosedov
  izberi naključno utež w
  ustvari nov primer na enak način, kot pri algoritmu SMOTE
  dodaj nov primer med dodatne primere
end while
združi originalno podatkovno množico in dodatne primere

3.3 SMOTERAND

SMOTERAND (glej Algoritem 7) je stohastična izpeljanka algoritma SMOTER, pri kateri primerov za generiranje novih primerov ne izbiramo glede na vrstni red ocen zanesljivosti njihovih klasifikacij, temveč naključno in v skladu s pripisanimi utežmi. Tudi pri tej različici pred generiranjem novih za vse manjšinske primere najprej izračunamo njihovo zanesljivost. Verjetnost vsakega primera X, da bo izbran kot osnova za generiranje novega primera, določa utež $weight_X$, ki jo iz zanesljivosti $reliability_X$ izračunamo po formuli:

$$weight_X = reliability_X^4$$

Tako se za generiranje novih primerov preferira bolj zanesljive primere, ki se v povprečju uporabijo večkrat, kot manj zanesljivi.

Algoritem 7 Psevdokoda algoritma SMOTERAND

```
for vsak manjšinski primer do
    izračunaj zanesljivost tega primera
end for
for vsak manjšinski primer do
    poišči njegove najbližje sosede, ki pripadajo istemu razredu
end for
while število primerov v manjšinskem razredu je premajhno do
    s pomočjo uteženega naključja izberi naslednji manjšinski primer
    naključno izberi enega od njegovih najbližjih sosedov
    izberi naključno utež w
    ustvari nov primer na enak način, kot pri algoritmu SMOTE
    dodaj nov primer med dodatne primere
end while
združi originalno podatkovno množico in dodatne primere
```

Poglavje 4

Empirično testiranje algoritmov

V tem poglavju bomo opisali testiranje algoritmov, rezultate testiranja in interpretacijo le-njih.

4.1 Uporabljene podatkovne množice

Vse v prejšnjih poglavjih opisane algoritme smo testirali na 10 podatkovnih množicah iz repozitorija UCI Machine Learning Repository [19]. Omejili smo se le na dvorazredne podatkovne množice. Izbrane podatkovne množice imajo različno število primerov, različne stopnje uravnoteženosti in različno število zveznih in diskretnih atributov. Vse uporabljene podatkovne množice so prosto dostopne na spletni strani UCI [19]. Naštete in opisane so v tabeli 4.1.

ime domene	št. primerov	št. zveznih atrib.	št. diskretnih atrib.
bupa	345 (200 + 145)	6	0
cmc	1473 (844 + 629)	6	3
crx	690 (383 + 307)	6	9
haberman	306 (225 + 81)	3	0
hepatitis	155 (123 + 32)	6	13
mammographic masses	961 (561 + 445)	1	4
parkinsons	195 (147 + 48)	22	0
pima-indians-diabetes	768 (500 + 268)	8	0
post-operative	88 (64 + 24)	1	7
statlog heart	270 (150 + 120)	10	3

Tabela 4.1: Opis uporabljenih podatkovnih množic (prikazano je skupno število primerov in število primerov v posameznih razredih)

4.2 Algoritmi

Da bi lahko algoritme za uravnoteženje podatkovnih množic čim bolj realno primerjali med seboj, smo jih priredili za potrebe testiranja. Vsakemu algoritmu prek vhodnih parametrov podamo:

- originalno podatkovno množico,
- stopnjo želene uravnoteženosti, ki predstavlja želen delež manjšinskih primerov glede na število večinskih primerov. Na primer, stopnja uravnoteženosti 200% pove algoritmu, da naj bo po obdelavi v podatkovni množici manjšinskih primerov dvakrat toliko kot večinskih,
- pri nekaterih algoritmih so možni še drugi parametri

Kot rezultat obdelave originalne podatkovne množice z izbranim algoritmom vedno dobimo uravnoteženo podatkovno množico z želenimi števili primerov v vsakem razredu.

Da bi lahko ocenili kvaliteto uravnoteženih podatkovnih množic, bomo na njih s pomočjo štirih klasifikatorjev izvajali 10-kratno prečno preverjanje [1]. Uporabili bomo naslednje klasifikatorje:

- odločitvena drevesa (R knjižnica rpart [23]),
- naivni Bayes (R knjižnica e1071 [21]),
- kNN - metoda k najbližjih sosedov (R knjižnica RWeka [28] [29], uporabili smo inverzno uteževanje primerov z oddaljenostjo in upoštevali 5 najbližjih sosedov in
- SVM - metoda podpornih vektorjev (R knjižnica e1071 [21], uporabili smo polinomsko jedro $kernel = "polynomial"$).

4.3 Metodologija testiranja algoritmov

Z vsakim algoritmom smo obdelali vseh 10 podatkovnih množic. Postopek (glej sliko 4.1) je bil sledeč:

1. podatkovno množico smo uravnotežili z enim od algoritmom za uravnoteženje podatkovnih množic (pri algoritmih SMOTER ASC, SMOTER DESC in SMOTERAND smo za ocenjevanje zanesljivosti klasifikacij primerov vedno uporabili enak klasifikator, kot je bil kasneje uporabljen za prečno preverjanje),

2. na uravnoteženi podatkovni množici smo s pomočjo klasifikatora izvedli 10-kratno prečno preverjanje [1],

3. iz rezultatov 10-kratnega prečnega preverjanja smo izračunali klasifikacijsko točnost, podatke za krivuljo ROC in vrednost AUC (angl. area under curve),

4. rezultate prečnega preverjanja smo uporabili še za medsebojno primerjavo algoritmov za uravnoteženje podatkov (z uporabo t-testa).

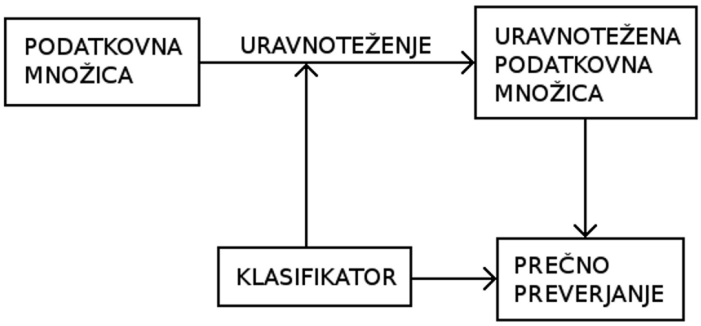
Ta postopek smo ponovili za vsako možno kombinacijo naslednjih parametrov:

- podatkovna množica (za testiranje smo izbrali 10 dvorazrednih podatkovnih množic, ki so naštete v tabeli 4.1),

- algoritem za uravnoteženje podatkovnih množic (podatkovne množice smo obdelovali z naključnim podvzorčenjem, naključnim nadvzorčenjem, SMOTE, SMOTER ASC, SMOTER DESC in SMOTERAND, za primerjavo pa smo testirali tudi na originalnih, neuravnoteženih podatkih),

- stopnja uravnoteženosti (za stopnje uravnoteženosti smo si izbrali tri vrednosti: 100%, 200% in 400%),

- klasifikator (za klasificiranje smo uporabili 4 klasifikatorje: odločitvena drevesa, naivni Bayes, metoda najbližjih sosedov in metoda podpornih vektorjev).



Slika 4.1: Ponazoritev poteka testiranja.

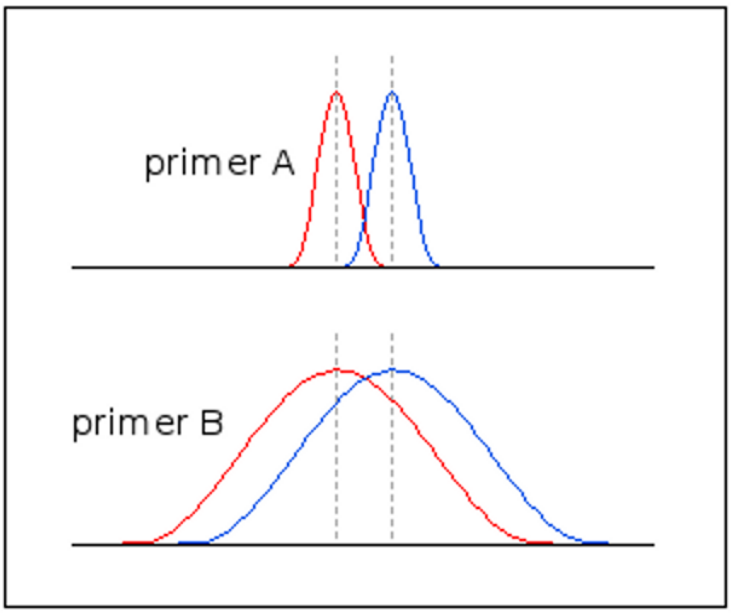
Najbolj nas bodo zanimali rezultati za stopnjo uravnoteženosti 100%, saj bo tu število primerov iz obeh razredov popolnoma enako, zanimivo pa bo videti tudi ostale rezultate.

Da bi lahko algoritme za uravnoteženje podatkovnih množic primerjali med seboj, smo uporabili še Welchev t test. S pomočjo vrednosti stopnje značilnosti α (angl. p-value) smo ocenili, ali so razlike med klasifikacijskimi točnostmi značilne ali ne.

Welchev t-test Welchev t-test se uporablja v statistiki, in sicer za medsebojno primerjavo dveh vzorcev (v našem primeru so to rezultati 10-kratnega prečnega preverjanja). Obstajata dve obliki tega testa: parni in neparni t-test. Parna oblika je uporabna, ko imamo v obeh vzorcih komponente, izračunane iz istih podatkov. Ker pa naši algoritmi pri uravnoteženju podatkovnih množic brišejo in dodajajo učne primere, parne oblike ne moremo uporabiti. Zato bomo uporablili neparno obliko t testa (angl. Welch t-test).

Pri rezultatih testa nas bo zanimala stopnja značilnosti testa (angl. p-value). Z njeno pomočjo bomo preverjali, ali je razlika med aritmetičnima sredinama obeh vzorcev (klasifikacijski rezultati 10-kratnega prečnega preverjanja) značilna, ali ne (primer na sliki 4.2). Za mejno vrednost smo si izbrali $\alpha = 0.05$, kar pomeni, da:

- vrednost $\alpha \leq 0.05$ pomeni, da je razlika med rezultatoma statistično značilna in
- vrednost $\alpha > 0.05$ pomeni, da razlika med rezultatoma ni statistično značilna.



Slika 4.2: Na sliki sta dva para vzorcev. V obeh primerih je razlika med aritmetično sredino modrega in rdečega vzorca enaka. Presek med vzorcema je v primeru A veliko manjši kot v primeru B, zato pravimo, da je razlika med vzorcema v primeru A bolj značilna.

4.4 Rezultati

4.4.1 Klasifikacijske točnosti in vrednosti AUC

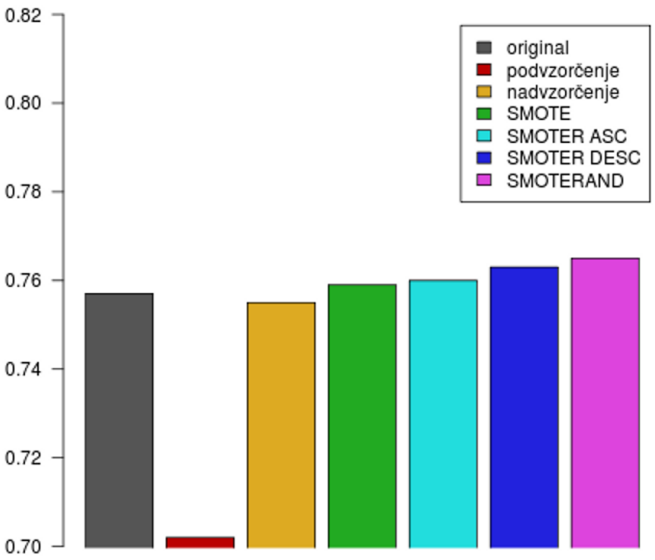
Povprečne klasifikacijske točnosti in vrednosti AUC za vse teste s stopnjo uravnoteženosti 100% so navedene v tabelah 4.2 in 4.3 in grafično prikazane v slikah 4.3 in 4.4. Bolj podrobni rezultati testiranj se nahajajo v tabelah v dodatku.

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)	povp.
original	0.769	0.736	0.761	0.764	0.757
podvzorčenje	0.703	0.696	0.697	0.712	0.702
nadvzorčenje	0.764	0.709	0.807	0.739	0.755
SMOTE	0.781	0.734	0.794	0.728	0.759
SMOTER ASC	0.763	0.732	0.783	0.762	0.760
SMOTER DESC	0.787	0.744	0.789	0.734	0.763
SMOTERAND	0.784	0.735	0.807	0.736	0.765
povprečje	0.764	0.727	0.777	0.739	

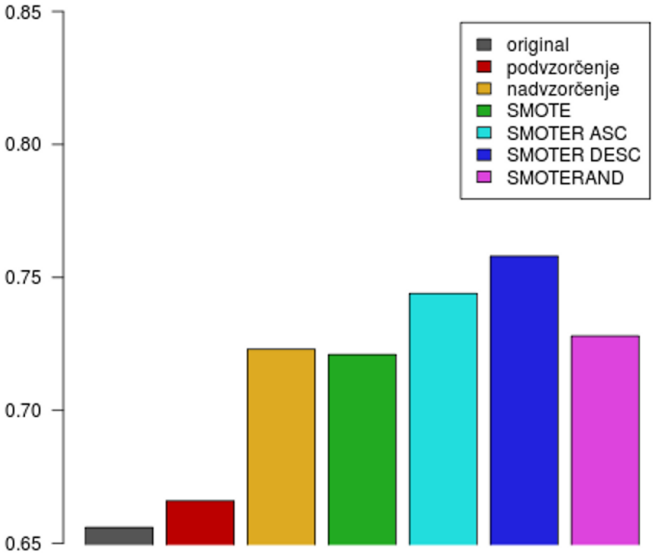
Tabela 4.2: Povprečne klasifikacijske točnosti vseh klasifikatorjev preko vseh domen (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)	povp.
original	0.725	0.753	0.750	0.398	0.656
podvzorčenje	0.720	0.746	0.754	0.445	0.666
nadvzorčenje	0.798	0.784	0.898	0.411	0.723
SMOTE	0.806	0.809	0.863	0.405	0.721
SMOTER A.	0.791	0.805	0.847	0.532	0.744
SMOTER D.	0.815	0.815	0.860	0.542	0.758
SMOTERAND	0.815	0.809	0.871	0.418	0.728
povprečje	0.781	0.789	0.835	0.450	

Tabela 4.3: Povprečne vrednosti AUC vseh klasifikatorjev preko vseh domen (stopnja uravnoteženosti 100%)



Slika 4.3: Povprečne klasifikacijske točnosti za algoritme za uravnoteženje podatkovnih množic (st. uravnoteženosti 100%)

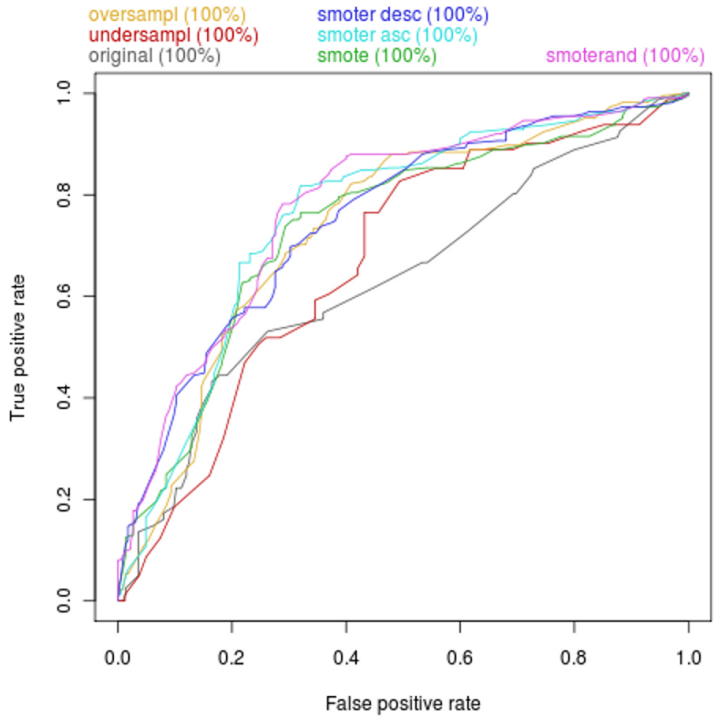


Slika 4.4: Povprečne vrednosti AUC za algoritme za uravnoteženje podatkovnih množic (st. uravnoteženosti 100%)

Iz tabel 4.2 in 4.3 vidimo, da je naključno podvzorčenje daleč najslabši algoritem za uravnoteženje podatkovnih množic. Klasifikatorji, ki so se učili iz podatkov, ki smo jih obdelali z njim, delajo največ napak pri klasificiranju, kar se kaže tako na slabi klasifikacijski točnosti, kot na tudi na nizkih vrednostih AUC. Še več - v veliko primerih delajo klasifikatorji več napak, če se na podatkovno množico uravnotežimo z naključnim podvzorčenjem, kot če bi pustili podatke neobdelane.

Naključno nadvzorčenje je po pričakovanjh botrovalo višjim točnostim pri klasifikaciji primerov in tudi višjim vrednostim AUC. Klasifikacijske točnosti in vrednosti AUC so pri algoritmu SMOTE v istem rangu kot pri naključnem nadvzorčenju, izpeljanke algoritma SMOTE pa se obnesejo malo bolje od vseh treh prej omenjenih algoritmov.

Tipičen primer uspešnosti algoritmov za uravnoteženje podatkovnih množic je prikazan na sliki 4.5. Na njej so prikazane ROC krivulje za odločitvena drevesa, ki smo jih naučili na podatkovni množici *hepatitis* - na originalni množici in na isti množici, uravnoteženi z različnimi algoritmi za uravnoteženje podatkovnih množic.



Slika 4.5: ROC krivulje odločitvenih dreves, naučenih na podatkovni množici "hepatitis", uravnoteženo z različnimi algoritmi (st. uravnoteženosti 100%)

4.4.2 Statistična primerjava algoritmov

Dobljene klasifikacijske točnosti z uporabo različnih tehnik vzorčenja primerov smo med seboj primerjali s t-testom. Za vsak možen par algoritmov smo testu podali rezultate 10-kratnega prečnega preverjanja in spremljali značilnost testne statistike. S primerjavo značilnosti in klasifikacijske točnosti ločimo tri situacije, ki so za nas zanimive:

- značilnost $\alpha \leq 0.05$ in klasifikacijska točnost prvega algoritma (vrstica) je višja od klasifikacijske točnosti drugega algoritma (stolpec),
- vrednost $\alpha \leq 0.05$ in klasifikacijska točnost prvega algoritma (vrstica) je nižja od klasifikacijske točnosti drugega algoritma (stolpec),
- vrednost $\alpha > 0.05$ - razlika ni značilna.

V celicah tabel 4.4, 4.5 in 4.6 so po tri vrednosti, ločene s poševnico. Prva vrednost je delež (delež je izražen v procentih in pomnožen s 100) testov, ko je

bil prvi algoritem (vrstica) značilno bolj točen pri klasifkaciji kot drugi (stolpec), druga vrednost v trojčku je delež testov, ko je bil prvi algoritem značilno manj natančen od drugega, zadnja vrednost pa predstavlja delež testov z neznačilnimi razlikami.

V zadnji vrstici so za lažjo primerjavo še povprečja za vse trojčke vrednosti.

algoritem	podvz.	nadvz.	SMOTE	SMOTER A.	SMOTER D.	SMOTERAND
podvzorčenje	0/0/100	2/22/75	2/28/70	0/22/78	2/25/72	2/32/65
nadvzorčenje	22/2/75	0/0/100	2/2/95	0/8/92	0/5/95	2/8/90
SMOTE	28/2/70	2/2/95	0/0/100	0/2/98	0/0/100	2/0/98
SMOTER A.	22/0/78	8/0/92	2/0/98	0/0/100	8/5/88	2/2/95
SMOTER D.	25/2/72	5/0/95	0/0/100	5/8/88	0/0/100	2/0/98
SMOTERAND	32/2/65	8/2/90	0/2/98	2/2/95	0/2/98	0/0/100
povprečje	22/2/77	4/5/91	1/5/93	1/7/92	2/6/92	2/7/91

Tabela 4.4: Primerjalna tabela algoritmov za uravnoteženje podatkovnih množic za stopnjo uravnoteženosti 100%.

algoritem	podvz.	nadvz.	SMOTE	SMOTER A.	SMOTER D.	SMOTERAND
podvzorčenje	0/0/100	0/45/55	0/55/45	0/52/48	0/55/45	0/55/45
nadvzorčenje	45/0/55	0/0/100	5/12/82	2/12/85	2/12/85	2/15/82
SMOTE	55/0/45	12/5/82	0/0/100	2/0/98	0/0/100	0/0/100
SMOTER A.	52/0/48	12/2/85	0/2/98	0/0/100	0/2/98	0/2/98
SMOTER D.	55/0/45	12/2/85	0/0/100	2/0/98	0/0/100	0/0/100
SMOTERAND	55/0/45	15/2/82	0/0/100	2/0/98	0/0/100	0/0/100
povprečje	44/0/56	9/10/82	1/12/88	2/11/88	0/12/88	0/12/88

Tabela 4.5: Primerjalna tabela algoritmov za uravnoteženje podatkovnih množic za stopnjo uravnoteženosti 200%

Iz tabel 4.4, 4.5 in 4.6 lahko razberemo, da so bile razlike med rezultati prečnega preverjanja pri algoritmu SMOTE in njegovih izpeljankah največkrat neznačilne. Kadar pa so bile razlike med SMOTE in njegovimi izpeljankami značilne, so pokazale na rahlo prednost pri uporabi izpeljank pred SMOTE. Tako majhne izboljšave kličejo po bolj obširnem testiranju.

algoritem	podvz.	nadvz.	SMOTE	SMOTER A.	SMOTER D.	SMOTERAND
podvzorčenje	0/0/100	0/35/65	0/40/60	0/45/55	0/48/52	0/48/52
nadvzorčenje	35/0/65	0/0/100	5/20/75	2/12/85	2/18/80	0/20/80
SMOTE	40/0/60	20/5/75	0/0/100	2/0/98	0/0/100	0/8/92
SMOTER A.	45/0/55	12/2/85	0/2/98	0/0/100	0/2/98	0/8/92
SMOTER D.	48/0/52	18/2/80	0/0/100	2/0/98	0/0/100	0/5/95
SMOTERAND	48/0/52	20/0/80	8/0/92	8/0/92	5/0/95	0/0/100
povprečje	36/0/64	12/8/81	2/10/88	2/10/88	1/11/88	0/15/85

Tabela 4.6: Primerjalna tabela algoritmov za uravnoteženje podatkovnih množic za stopnjo uravnoteženosti 400%

Več statistično značilnih razlik je opaziti, če primerjamo algoritem naključno nadvzorčenje s katerokoli varianto algoritma SMOTE. Tu lahko s sigurnostjo rečemo, da so algoritmi SMOTE in njegove izpeljanke občutna izboljšava glede na nalkjučno nadvzorčenje.

Najbolj izstopa naključno podvzorčenje, pri katerem so se pogosto pojavljale statistično značilne razlike v primerjavi z vsemi ostalimi algoritmi za uravnoteženje podatkovnih množic. Brez dvoma lahko rečemo, da je algoritem naključno podvzorčenje v primerjavi z ostalimi inferioren.

4.4.3 Izboljšave AUC klasifikatorjev

Tabele 4.7 - 4.10 (vsaka za en klasifikator) prikazujejo vse uporabljene podatkovne množice in za vsako od podatkovnih množic:

- št. primerov v vsaki (originalni) podatkovni množici (v večinskem in manjšinskem razredu),
- vrednost AUC za klasifikator (odločitveno drevo), naučen na originalni podatkovni množici,
- najvišjo vrednost AUC med klasifikatorji, naučenimi na uravnoteženih podatkovnih množicah (z vsemi v tej diplomi omenjenimi algoritmi),
- iz prejšnjih dveh vrednosti AUC izračunano relativno povečanje vrednosti AUC.

ime domene	št. primerov	orig. AUC	najboljši AUC	izboljšava
bupa	200+145	0.710	SMOTER D. 0.739	+4.1%
cmc	844+629	0.721	SMOTER D. 0.772	+7.1%
crx	383+307	0.885	SMOTER D. 0.902	+1.9%
haberman	225+81	0.635	SMOTERAND 0.776	+22.2%
hepatitis	123+32	0.657	SMOTERAND 0.878	+33.6%
mammographic masses	561+445	0.843	SMOTER A. 0.863	+2.4%
parkinsons	147+48	0.850	nadvzorčenje 0.955	+12.4%
pima-indians-diabetes	500+268	0.786	nadvzorčenje 0.813	+3.4%
post-operative	64+24	0.367	SMOTERAND 0.746	+103.3%
statlog heart	150+120	0.810	SMOTER D. 0.841	+3.8%

Tabela 4.7: Vrednosti AUC za originalno podatkovno množico in najuspešnejši algoritem pri stopnji uravnoteženosti 100% (klasifikator: **odločitveno drevo**)

ime domene	št. primerov	orig. AUC	najboljši AUC	izboljšava
bupa	200+145	0.621	SMOTER D. 0.678	+9.2%
cmc	844+629	0.683	SMOTER A. 0.697	+2.0%
crx	383+307	0.882	SMOTER D. 0.902	+2.3%
haberman	225+81	0.617	nadvzorčenje 0.672	+8.9%
hepatitis	123+32	0.819	SMOTER A. 0.943	+15.1%
mammographic masses	561+445	0.901	SMOTER D. 0.915	+1.6%
parkinsons	147+48	0.859	SMOTERAND 0.886	+3.1%
pima-indians-diabetes	500+268	0.809	SMOTER D. 0.824	+1.9%
post-operative	64+24	0.432	SMOTER D. 0.778	+80.1%
statlog heart	150+120	0.906	SMOTER D. 0.925	+2.1%

Tabela 4.8: Vrednosti AUC za originalno podatkovno množico in najuspešnejši algoritem pri stopnji uravnoteženosti 100% (klasifikator: **naivni Bayes**)

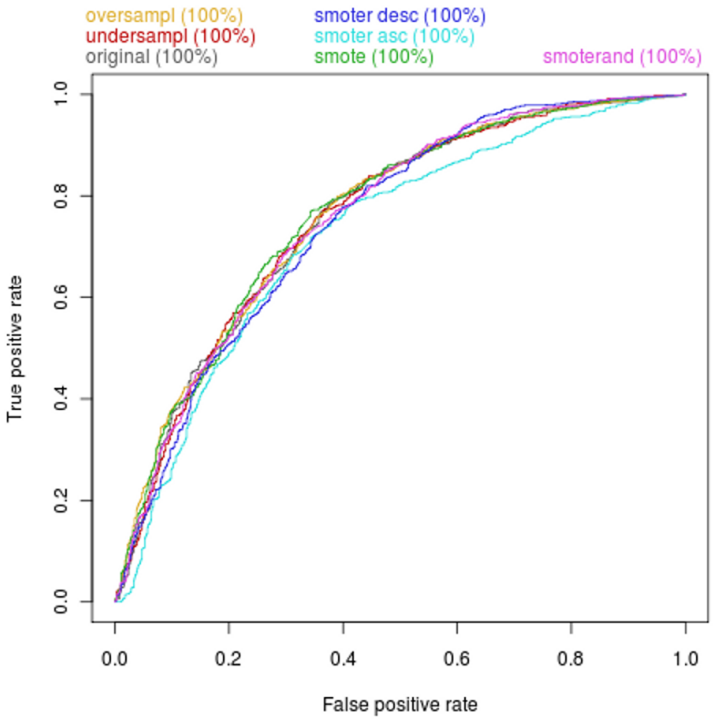
ime domene	št. primerov	orig. AUC	najboljši AUC	izboljšava
bupa	200+145	0.685	nadvzorčenje 0.829	+21.0%
cmc	844+629	0.640	nadvzorčenje 0.784	+22.5%
crx	383+307	0.901	nadvzorčenje 0.929	+3.1%
haberman	225+81	0.625	nadvzorčenje 0.892	+42.7%
hepatitis	123+32	0.839	SMOTERAND 0.969	+15.5%
mammographic masses	561+445	0.857	SMOTER D. 0.880	+2.7%
parkinsons	147+48	0.989	nadvzorčenje 0.998	+0.9%
pima-indians-diabetes	500+268	0.778	nadvzorčenje 0.918	+18.0%
post-operative	64+24	0.293	nadvzorčenje 0.875	+198.7%
statlog heart	150+120	0.889	nadvzorčenje 0.925	+4.0%

Tabela 4.9: Vrednosti AUC za originalno podatkovno množico in najuspešnejši algoritem pri stopnji uravnoteženosti 100% (klasifikator: **metoda najbližjih sosedov**)

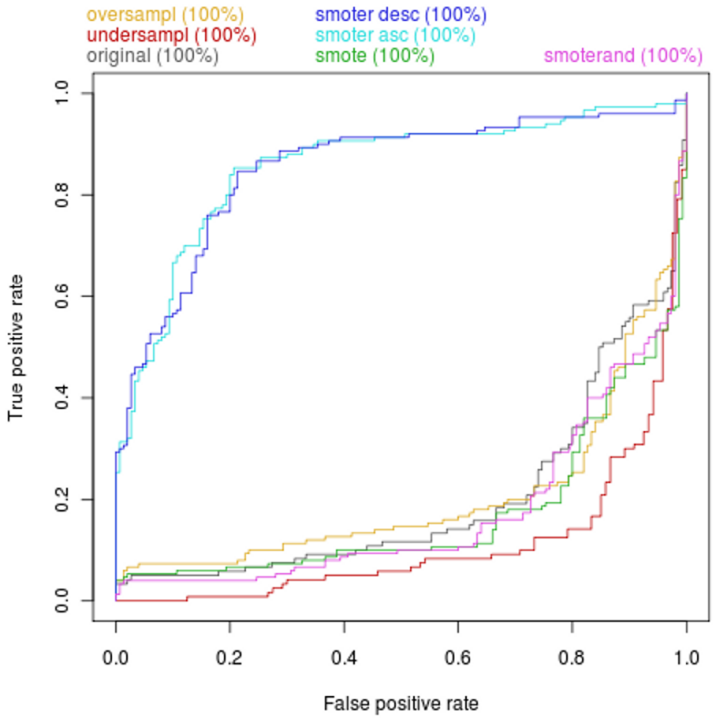
ime domene	št. primerov	orig. AUC	najboljši AUC	izboljšava
bupa	200+145	0.634	nadvzorčenje 0.633	-0.2%
cmc	844+629	0.758	SMOTE 0.763	+0.7%
crx	383+307	0.880	SMOTE 0.895	+1.7%
haberman	225+81	0.434	nadvzorčenje 0.676	+55.8%
hepatitis	123+32	0.150	SMOTER D. 0.952	+534.7%
mammographic masses	561+445	0.101	SMOTER D. 0.845	+736.6%
parkinsons	147+48	0.100	podvzorčenje 0.913	+813.0%
pima-indians-diabetes	500+268	0.261	SMOTERAND 0.251	-3.8%
post-operative	64+24	0.457	SMOTER A. 0.712	+55.8%
statlog heart	150+120	0.201	SMOTER A. 0.856	+325.9%

Tabela 4.10: Vrednosti AUC za originalno podatkovno množico in najuspešnejši algoritem pri stopnji uravnoteženosti 100% (klasifikator: **metoda podpornih vektorjev**)

Občutljivost klasifikatorjev. Če analiziramo tabele 4.7 - 4.10, nam najprej padejo v oči ogromna povečanja vrednosti AUC v zadnji tabeli, kjer smo pri 10-kratnem prečnem preverjanju za klasifikator uporabili metodo podpornih vektorjev (SVM). Ta povečanja so predvsem posledica zelo nizkih vrednosti AUC za prečna preverjanja s SVM na nauravnoteženih množicah. Iz tega bi lahko sklepali, da je bila naša prvotna predpostavka glede metode podpornih vektorjev pravilna. Iz podatkov v tabeli 4.10 namreč lahko sklepamo, da je metoda podpornih vektorjev občutljiva na neuravnoteženost podatkov. Vse tri domene iz tabele 4.10, pri katerih je vrednost AUC za prečno preverjanje na neuravnoteženih podatkih večja od 0.5, imajo namreč visok delež manjšinskih primerov. To so: *bupa*, *cmc* in *crx*. Na slikah 4.6, 4.7 in 4.8 lahko vidimo velike razlike v poteku ROC krivulj za klasifikatorje po metodi podpornih vektorjev (SVM) za originalne množice (siva krivulja) *cmc*, *heart* in *pima* in za vse množice, predobdelane z algoritmi za uravnoteženje podatkovnih množic.

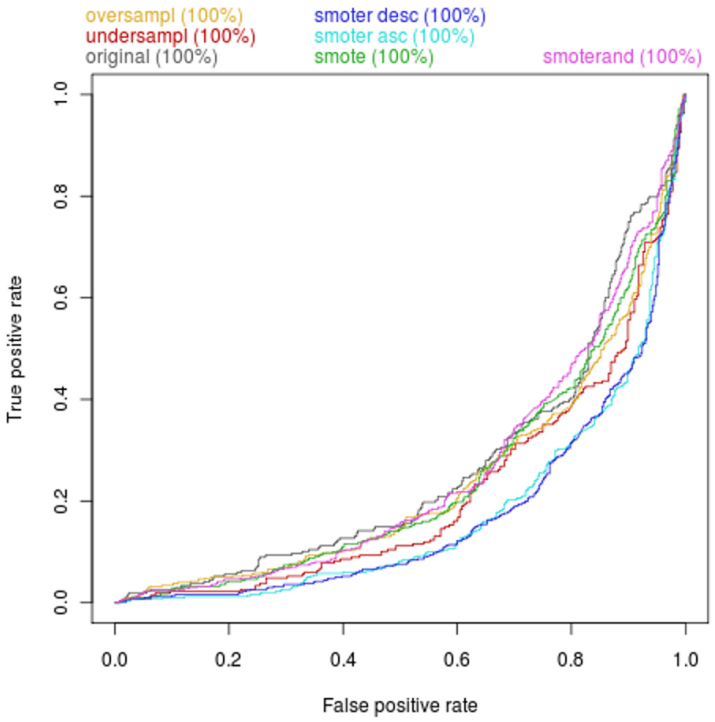


Slika 4.6: ROC krivulje klasifikatorjev SVM, naučenih na podatkovni množici "cmc", uravnoteženo z različnimi algoritmi (st. uravnoteženosti 100%)



Slika 4.7: ROC krivulje klasifikatorjev SVM, naučenih na podatkovni množici "heart", uravnoteženo z različnimi algoritmi (st. uravnoteženosti 100%)

Vpliv nizkega deleža manjšinjskih primerov. Druga zanimiva značilnost tabel 4.7 - 4.10 je, da so imajo vsi štirje klasifikatorji nizke vrednosti AUC za prečno preverjanje na neuravnoteženi domeni *post-operative*. Vrednosti AUC za vse štiri klasifikatorje se po uravnoteženju te podatkovne množice povečajo in postanejo primerljive z vrednostmi za ostale domene. Podatkovna množica *post-operative* je najmanjša med vsemi in ima zelo majhen delež manjšinskih primerov (le 27% - manjši delež ima le še *hepatitis*). Ker je podatkovna množica *post-operative* nizek delež manjšinskih primerov, se pri uravnoteženju te množice generira relativno veliko manjšinskih primerov (relativno več, kot pri večini ostalih domen). Ker je zvišanje vrednosti AUC pri *post-operative* najbolj konstantno ne glede na klasifikator, lahko rečemo, da se pri tej podatkovni množici najbolj izplača. Lahko bi tudi sklepali, da se uravnoteženje najbolj izplača pri majhnih podatkovnih množicah z nizkim deležem manjšinskih primerov, a to predpostavko bi bilo potrebno dokazati z bolj podrobno raziskavo.

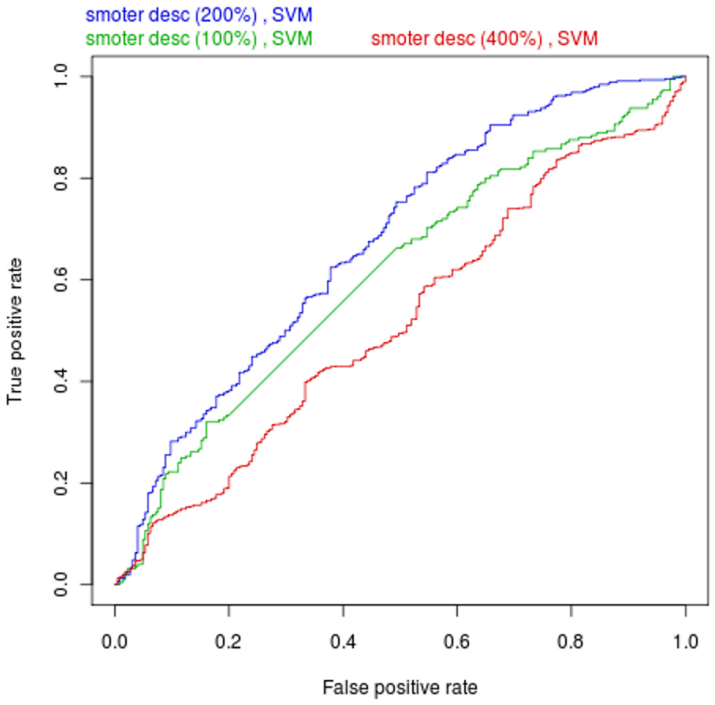


Slika 4.8: ROC krivulje klasifikatorjev SVM, naučenih na podatkovni množici "pima", uravnoteženo z različnimi algoritmi (st. uravnoteženosti 100%)

Pristranost metode k-NN. Še enkrat si poglejmo izboljšanja vrednosti AUC. Tokrat nas zanima, koliko teh izboljšanj se je pojavilo pri naših izpeljankah algoritma SMOTE. Spet lahko opazimo eno zanimivost. V tabeli 4.9, kjer so napisana izboljšanja vrednosti AUC za klasifikacijo po metodi najbližjih sosedov, se pri kar osmih od desetih domen (80%) največje izboljšanje pojavi za algoritem naključno nadvzorčenje. To je posledica delovanja tega klasifikatorja in algoritma naključno nadvzorčenje. Klasifikacija po metodi najbližjih sosedov vsak primer klasificira na osnovi klasifikacij njegovih sosedov. Ker pa z algoritmom naključno nadvzorčenje v bistvu podvajamo manjšinske primere, bo naš klasifikator med sosede manjšinskih primerov velikokrat uvrstil njegove duplikate. Tako se velikokrat zgodi, da je ena kopija primera v učni, druga pa v testni množici, kar posledično vodi k višji klasifikacijski točnosti.

Iz ostalih treh tabel (4.7, 4.8 in 4.10) lahko vidimo, da je v 22 od 30 primerov (torej v dobrih 73% primerov) največjemu izboljšanju botroval eden od naših algoritmov za uravnoteženje podatkovnih množic (SMOTER ASC, SMOTER DESC in SMOTERAND). Iz tega lahko sklepamo, da je za odločitvena drevesa, naivni Bayes in metodo podpornih vektorjev generiranje sintetičnih primerov na podlagi ocen zanesljivosti klasifikacij primerov dober pristop k izboljšanju klasifikacijskih točnosti na neuravnoteženih podatkovnih množicah.

Prednost izpeljank metode SMOTE. Ob koncu tega poglavja se dotaknimo še rezultatov za različne stopnje uravnoteženosti. Do sedaj smo se namreč ukvarjali le z rezultati za stopnjo uravnoteženosti 100%, ki nam v uravnoteženi podatkovni množici zagotavlja enako število manjšinskih in večinskih primerov. Vsa testiranja smo opravili tudi za stopnji uravnoteženosti 200% in 400%. V večini primerov se klasifikacijska točnost in vrednost AUC povečujeta v sorazmerju s stopnjo uravnoteženosti, ne pa povsod (slika 4.9). Ko povečujemo stopnjo uravnoteženosti, se razmerje med številom manjšinskih in večinskih primerov začne obračati in zopet pridemo do problema neuravnotežene podatkovne množice. Zato sklepamo, da za vsako kombinacijo (podatkovna množica, klasifikator, algoritem za uravnoteženje podatkovne množice) obstaja stopnja uravnoteženosti, za katero je vrednost ROC maksimalna. Ta idealna stopnja uravnoteženosti pa v nasprotju s prvim pomislekom ni nujno 100%.



Slika 4.9: ROC krivulje klasifikatorjev SVM, naučenih na podatkovni množici "haberman", uravnoteženi z algoritmom SMOTER DESC (različne stopnje uravnoteženosti)

Poglavje 5

Zaključek

Cilj diplomske naloge je bil pokazati, da lahko pridemo do večjih klasifikacijskih točnosti, če pri generiranju sintetičnih primerov uporabljamo bolj usmerjeno generiranje sintetičnih primerov od naključnega. S tem namenom smo uporabili oceno zanesljivosti klasifikacij primerov in iz algoritma SMOTE izpeljali tri nove algoritme (SMOTER ASC, SMOTER DESC in SMOTERAND), ki pri svojem delovanju upoštevajo to oceno zanesljivosti klasifikacij (razdelek 3.1. Algoritem SMOTE in njegove izpeljanke smo primerjali še z algoritmoma naključno podvzorčenje in naključn nadvzorčenje. Eksperimentirali smo z različnimi klasifikatorji (odločitvena drevesa, naivni Bayes, metoda najbližjih sosedov, metoda podpornih vektorjev) in različnimi stopnjami uravnoteženosti, rezultate prečnih preverjanj pa predstavili v poglavju 4.4.

S praktičnega stališča so najbolj zanimivi rezultati testiranj s stopnjo uravnoteženosti 100%. Ta stopnja nam zagotavlja, da sta po obdelavi v podatkovni množici enako pogosto zastopana oba razreda, kar naj bi izničilo morebitno klasifikatorjevo občutljivost na neuravnoteženost podatkov. Zato bomo v tem poglavju skoncentrirali le rezultate testiranj s stopnjo uravnoteženosti 100%.

Iz tabel 4.2 in 4.3 lahko vidimo, da so razlike v klasifikacijskih točnostih med algoritmom SMOTE in njegovimi izpeljankami minimalne. Rezultati testiranj naključnega nadvzorčenja so primerljivi z algoritmom SMOTE in njegovimi izpeljankami, vsi omenjeni algoritmi pa se obnesejo bolje kot naključno podvzorčenje. Isti zaključek lahko potegnemo tudi za povprečne vrednosti AUC. To pomeni, da se v večini primerov najbolj izplača uravnotežiti podatkovno množico z algoritmom SMOTE ali pa s katero od njegovih izpeljank. Izpeljanke algoritma SMOTE se v povprečju obnesejo malenkost bolje kot SMOTE, vendar pa so razlike zelo majhne in bi lahko prišlo do nje že zaradi zaporedij primerov v podatkovnih množicah ali pa zaradi nedeterminizma pri razdelitvi

podatkovne množice za potrebe 10-kratnega prečnega preverjanja. Rezultati torej kažejo na veliko podobnost pri delovanju teh algoritmov.

Ko pogledamo, koliko in na katerih podatkovnih množicah je klasifikatorjem uravnoteženje podatkovne množice koristilo, opazimo zanimiv vzorec. V tabelah 4.7 - 4.10 opazimo največje izboljšave vrednosti AUC pri majhnih podatkovnih množicah, ki imajo v svoji prvotni (neuravnoteženi) obliki tudi relativno najmanjše deleže manjšinskih primerov. Ko pa smo uravnotežili velike podatkovne množice s skoraj enakima deležema večinskih in manjšinskih primerov in na njih učili klasifikatorje, ni prišlo do večjih izboljšav v primerjavi s klasifikatorji, ki so bili naučeni na originalnih množicah.

Iz tabel 4.7 - 4.10 je videti tudi, da je postopek uravnoteženja podatkovne množice zelo koristen na majhnih in neuravnoteženih množicah. Uravnoteženje velikih množic z majhno relativno razliko med številom večinskih in manjšinskih primerov pa očitno ne prinaša večjih izboljšav. Iz danih podatkov bi lahko sklepali, da je za koristnost uravnoteženja podatkovne množice bolj kot relativen delež manjšinskih primerov pomembna velikost množice, vendar pa tega ne moremo trditi z gotovostjo. To teorijo bi bilo treba preveriti še na umetnih podatkovnih množicah, kjer bi lahko med drugim kontrolirali tudi stopnjo kompleksnosti podatkov. Še posebej primerne se zdijo umetno zgenerirane podatkovne množice, kot so opisane v [12] (poglavje 3.1).

5.1 Nadaljnje delo

V tej diplomi smo se omejili le na dvorazredne podatkovne množice. Logična razširitev tu opravljenega dela bi bila posplošitev problema na večrazredne množice. Poleg tega bi lahko implementirali še kakšno varianto algoritma SMOTE, kot so algoritmi SMOTEBoost [13], SMOTE-NC [4] in SMOTE-N [4].

Merjenje uspešnosti v prejšnjih poglavjih omenjenih algoritmov na realnih podatkih se sliši zanimivo. Zagotovo pa bi se lahko dokopali do bolj oprijemljivih rezultatov, če bi merili uspešnost na povsem umetnih podatkovnih množicah, kjer bi lahko nadzorovali med drugim [12]:

- stopnjo uteženosti podatkovne množice,
- število primerov v posameznem razredu,
- število diskretnih in numeričnih atributov v podatkovni množici ter
- kompleksnost učenja za klasifikator, ki ga uporabljamo za testiranje.

Iz rezultatov smo potegnili še nekaj zaključkov, ki bi jih bilo treba preveriti z obširnejšim testiranjem:

- Do največjih izboljšav klasifikacijske točnosti in vrednosti AUC prihaja pri uravnoteženju majhnih podatkovnih množic in množic z majhnim deležem manjšinskih primerov.
- Obstaja idealna stopnja uravnoteženosti, pri kateri dosega določen klasifikator z uporabo določenega algoritma za uravnoteženje na določeni podatkovni množici največjo možno klasifikacijsko točnost.
- Statistične razlike med rezultati prečnih preverjanj kažejo na rahlo prednost izpeljank algoritma SMOTE pred njegovo originalno obliko.

Literatura

[1] I. Kononenko, *Strojno učenje*, Ljubljana: Založba fakultete za elektrotehniko in fakultete za računalništvo in informatiko, 2005.

[2] I. Kononenko, M. Kukar, *Machine Learning and Data Mining: Introduction to Principles and Algorithms*, Horwood Publishing 2007.

[3] I. Bratko, *Prolog in umetna inteligenca*, Ljubljana: Založba fakultete za elektrotehniko in fakultete za računalništvo in informatiko, 1997.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," v zborniku *Journal of Artificial Intelligence Research 16*, jun. 2002, str. 321-357.

[5] M. Kukar, *Ocenjevanje zanesljivosti klasifikacij in cenovno občutljivo kombiniranje metod strojnega učenja*, doktorska disertacija, Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Ljubljana, 2001.

[6] M. Kukar, I. Kononenko, "Reliable Classifications with Machine Learning,", v zborniku *Proceedings of the 13th European Conference on Machine Learning*, 2002, str. 219-231.

[7] J. Stefanowski, S. Wilk, "Selective Pre-processing of Imbalanced Data for Improving Classification Performance," v zborniku *Proceedings of the 10th international conference on Data Warehousing and Knowledge Discovery*, 2008, str. 283-292.

[8] D. A. Cieslak, N. V. Chawla, "Learning Decision Trees for Unbalanced data," v zborniku *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases - Part I*, 2008, str. 241-256.

[9] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.

[10] T. Dietterich, M. Kearns, Y. Mansour, "Applying the Weak Learning Framework to Understand and Improve C4.5," v zborniku *In Proceedings of the Thirteenth International Conference on Machine Learning*, 1996, str. 96-104.

[11] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, *Classification and regression trees*, Monterey, CA: Wadsworth & Brooks/Cole Advanced Books & Software, 1984.

[12] N. Japkowicz, S. Stephen, "The class imbalance problem: A systematic study," v zborniku *Intelligent Data Analysis, Volume 6, Number 5*, 2002, str. 429-449.

[13] N. V. Chawla, A. Lazarevic, L. O. Hall, K. Bowyer, "SMOTEBoost: Improving prediction of the Minority Class in Boosting," v zborniku *In Proceedings of the Principles of Knowledge Discovery in Databases*, 2003, str. 107-119.

[14] N. V. Chawla, N. Japkowicz, A. Kolcz, "Editorial: Special Issue on Learning from Imbalanced Data Sets," *ACM SIGKDD Explorations, Volume 6, Number 1*, jun. 2004, str. 1-6.

[15] M. Kubat, R. Holte, S. Matwin, "Machine Learning for the Detection of Oil Spills in Satellite Radar Images," v zborniku *Machine Learning, Volume 30, Issue 2-3*, 1998, str. 195-215.

[16] F. Provost, "Machine Learning from Imbalanced Data Sets 101 (Extended Abstract)," na delavnici *AAAI'2000 Workshop on Imbalanced Data Sets*, 2000.

[17] M. Kubat, S. Matwin, "Addressing the curse of imbalanced data sets: One-sided sampling," v zborniku *In Proceedings of the Fourteenth International Conference on Machine Learning*, 1997, str. 179-186.

[18] J. Zhang, E. Bloedorn, L. Rosen, D. Venese, "Learning Rules from Highly Unbalanced Data Sets," na konferenci *International Conference on Data Mining 2004*, 2004, str. 571-574.

[19] UCI Machine Learning Repository. Dostopno na: http://archive.ics.uci.edu/ml/

[20] R Development Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2009. Dostopno na:
http://www.R-project.org

[21] E. Dimitriadou, K. Hornik, F. Leisch, D. Meyer, A. Weingessel, *e1071: Misc Functions of the Department of Statistics*, TU Wien, 2009. Dostopno na:
http://CRAN.R-project.org/package=e1071 (različica R paketa 1.5-22)

[22] C. Weihs, U. Ligges, K. Luebke. N. Raabe, *klaR Analyzing German Business Cycles*, 2005. Dostopno na:
http://CRAN.R-project.org/package=klaR

[23] T. M. Therneau, B. Atkinson, B. Ripley, *rpart: Recursive Partitioning*, 2009. Dostopno na:
http://CRAN.R-project.org/package=rpart (različica R paketa 3.1-44)

[24] J. Tuszynski, *caTools: Tools: moving window statistics, GIF, Base64, ROC AUC, etc.*, 2009. Dostopnno na:
http://CRAN.R-project.org/package=caTools (različica R paketa 1.10.)

[25] NCAR - Research Application Program, *verification: Forecast verification utilities*, 2010. Dostopno na:
http://CRAN.R-project.org/package=verification (različica R paketa 1.31)

[26] T. Sing, O. Sander, N. Beerenwinkel, T. Lengauer, *ROCR: Visualizing the performance of scoring classifiers*, 2007. Dostopno na:
http://rocr.bioinf.mpi-sb.mpg.de/ (različica R paketa 1.0-2)

[27] A. Dunlap Brooks, *knnflex: A more flexible KNN*, 2007. Dostopno na:
http://CRAN.R-project.org/package=knnflex (različica R paketa 1.1.1)

[28] K. Hornik, C. Buchta, A. Zeileis, "Open-Source Machine Learning: R Meets Weka," v zborniku *Computational Statistics, Volume 24, Issue 2*, maj 2009, str. 225-232.

[29] I. H. Witten, E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques. 2nd Edition*, San Francisco: Morgan Kaufmann, 2005.

Dodatek A

Podrobni rezultati testiranj

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.690	0.571	0.641	0.592
podvzorčenje	0.610	0.562	0.614	0.579
nadvzorčenje	0.645	0.562	0.700	0.622
SMOTE	0.693	0.565	0.700	0.598
SMOTER ASC	0.637	0.560	0.688	0.637
SMOTER DESC	0.725	0.578	0.655	0.598
SMOTERAND	0.708	0.560	0.693	0.605

Tabela A.1: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "bupa" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.701	0.621	0.685	0.634
podvzorčenje	0.615	0.581	0.664	0.624
nadvzorčenje	0.672	0.614	0.829	0.633
SMOTE	0.720	0.666	0.810	0.623
SMOTER ASC	0.675	0.609	0.762	0.343
SMOTER DESC	0.739	0.678	0.772	0.605
SMOTERAND	0.712	0.644	0.783	0.623

Tabela A.2: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "bupa" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.716	0.649	0.625	0.715
podvzorčenje	0.687	0.630	0.595	0.696
nadvzorčenje	0.700	0.630	0.697	0.695
SMOTE	0.708	0.630	0.671	0.707
SMOTER ASC	0.687	0.646	0.653	0.723
SMOTER DESC	0.738	0.611	0.653	0.685
SMOTERAND	0.695	0.626	0.655	0.693

Tabela A.3: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "cmc" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.721	0.683	0.640	0.758
podvzorčenje	0.714	0.679	0.636	0.757
nadvzorčenje	0.737	0.687	0.784	0.761
SMOTE	0.745	0.681	0.715	0.763
SMOTER ASC	0.733	0.697	0.703	0.723
SMOTER DESC	0.772	0.655	0.722	0.748
SMOTERAND	0.741	0.678	0.726	0.757

Tabela A.4: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "cmc" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.839	0.774	0.858	0.796
podvzorčenje	0.835	0.754	0.845	0.748
nadvzorčenje	0.858	0.744	0.876	0.799
SMOTE	0.869	0.757	0.863	0.804
SMOTER ASC	0.867	0.747	0.858	0.800
SMOTER DESC	0.864	0.789	0.860	0.820
SMOTERAND	0.876	0.760	0.876	0.795

Tabela A.5: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "crx" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.885	0.882	0.901	0.880
podvzorčenje	0.870	0.878	0.901	0.149
nadvzorčenje	0.885	0.878	0.929	0.881
SMOTE	0.892	0.893	0.913	0.895
SMOTER ASC	0.859	0.886	0.906	0.895
SMOTER DESC	0.902	0.902	0.918	0.0935
SMOTERAND	0.901	0.886	0.924	0.895

Tabela A.6: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "crx" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.712	0.751	0.663	0.732
podvzorčenje	0.647	0.617	0.643	0.579
nadvzorčenje	0.691	0.631	0.782	0.596
SMOTE	0.722	0.624	0.742	0.533
SMOTER ASC	0.738	0.613	0.751	0.587
SMOTER DESC	0.687	0.620	0.744	0.567
SMOTERAND	0.733	0.607	0.791	0.609

Tabela A.7: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "haberman" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.635	0.617	0.625	0.434
podvzorčenje	0.669	0.600	0.630	0.373
nadvzorčenje	0.736	0.672	0.892	0.676
SMOTE	0.735	0.661	0.816	0.561
SMOTER ASC	0.758	0.654	0.807	0.610
SMOTER DESC	0.751	0.666	0.827	0.607
SMOTERAND	0.776	0.646	0.877	0.625

Tabela A.8: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "haberman" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.781	0.846	0.840	0.844
podvzorčenje	0.802	0.790	0.707	0.752
nadvzorčenje	0.833	0.855	0.858	0.874
SMOTE	0.850	0.883	0.877	0.919
SMOTER ASC	0.837	0.886	0.882	0.907
SMOTER DESC	0.834	0.890	0.886	0.919
SMOTERAND	0.866	0.878	0.903	0.914

Tabela A.9: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "hepatitis" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.657	0.819	0.839	0.150
podvzorčenje	0.790	0.886	0.863	0.869
nadvzorčenje	0.864	0.911	0.952	0.0543
SMOTE	0.841	0.925	0.966	0.0467
SMOTER ASC	0.867	0.943	0.962	0.0401
SMOTER DESC	0.843	0.932	0.962	0.952
SMOTERAND	0.878	0.942	0.969	0.0404

Tabela A.10: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "hepatitis" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.823	0.829	0.797	0.817
podvzorčenje	0.830	0.830	0.792	0.817
nadvzorčenje	0.824	0.828	0.805	0.816
SMOTE	0.816	0.828	0.801	0.817
SMOTER ASC	0.825	0.823	0.810	0.821
SMOTER DESC	0.828	0.840	0.812	0.828
SMOTERAND	0.821	0.824	0.813	0.820

Tabela A.11: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "mammographic masses" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.843	0.901	0.857	0.101
podvzorčenje	0.852	0.899	0.850	0.103
nadvzorčenje	0.852	0.903	0.877	0.102
SMOTE	0.843	0.901	0.874	0.101
SMOTER ASC	0.863	0.894	0.867	0.0996
SMOTER DESC	0.856	0.915	0.880	0.845
SMOTERAND	0.857	0.901	0.879	0.102

Tabela A.12: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "mammographic masses" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.867	0.701	0.939	0.877
podvzorčenje	0.748	0.778	0.877	0.761
nadvzorčenje	0.925	0.775	0.946	0.860
SMOTE	0.912	0.779	0.945	0.860
SMOTER ASC	0.884	0.776	0.952	0.888
SMOTER DESC	0.918	0.786	0.956	0.864
SMOTERAND	0.847	0.790	0.942	0.871

Tabela A.13: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "parkinsons" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.850	0.859	0.989	0.100
podvzorčenje	0.779	0.844	0.964	0.913
nadvzorčenje	0.955	0.869	0.998	0.0507
SMOTE	0.927	0.875	0.995	0.0491
SMOTER ASC	0.908	0.872	0.994	0.870
SMOTER DESC	0.915	0.874	0.996	0.048
SMOTERAND	0.900	0.886	0.995	0.052

Tabela A.14: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "parkinsons" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.738	0.750	0.737	0.725
podvzorčenje	0.726	0.737	0.709	0.709
nadvzorčenje	0.760	0.746	0.819	0.752
SMOTE	0.782	0.734	0.817	0.738
SMOTER ASC	0.772	0.729	0.807	0.739
SMOTER DESC	0.754	0.741	0.799	0.744
SMOTERAND	0.765	0.741	0.807	0.736

Tabela A.15: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "pima-indians-diabetes" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.786	0.809	0.778	0.261
podvzorčenje	0.768	0.813	0.790	0.213
nadvzorčenje	0.813	0.822	0.918	0.237
SMOTE	0.811	0.823	0.897	0.241
SMOTER ASC	0.792	0.812	0.886	0.172
SMOTER DESC	0.797	0.824	0.880	0.170
SMOTERAND	0.804	0.819	0.890	0.251

Tabela A.16: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "pima-indians-diabetes" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.700	0.656	0.667	0.733
podvzorčenje	0.350	0.435	0.350	0.670
nadvzorčenje	0.605	0.487	0.707	0.548
SMOTE	0.674	0.677	0.697	0.505
SMOTER ASC	0.604	0.682	0.582	0.697
SMOTER DESC	0.716	0.712	0.683	0.499
SMOTERAND	0.705	0.704	0.735	0.490

Tabela A.17: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "post-operative" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.367	0.432	0.293	0.457
podvzorčenje	0.339	0.371	0.351	0.339
nadvzorčenje	0.631	0.580	0.875	0.512
SMOTE	0.727	0.744	0.731	0.594
SMOTER ASC	0.646	0.767	0.681	0.712
SMOTER DESC	0.736	0.778	0.743	0.498
SMOTERAND	0.746	0.767	0.754	0.655

Tabela A.18: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "post-operative" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.826	0.837	0.848	0.807
podvzorčenje	0.792	0.829	0.838	0.804
nadvzorčenje	0.797	0.833	0.877	0.823
SMOTE	0.787	0.863	0.827	0.803
SMOTER ASC	0.777	0.853	0.843	0.823
SMOTER DESC	0.807	0.870	0.840	0.817
SMOTERAND	0.820	0.857	0.850	0.830

Tabela A.19: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "statlog heart" (stopnja uravnoteženosti 100%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.810	0.906	0.889	0.201
podvzorčenje	0.802	0.911	0.893	0.110
nadvzorčenje	0.835	0.902	0.925	0.206
SMOTE	0.822	0.921	0.909	0.172
SMOTER ASC	0.803	0.914	0.903	0.856
SMOTER DESC	0.841	0.925	0.904	0.853
SMOTERAND	0.837	0.922	0.912	0.175

Tabela A.20: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "statlog heart" (stopnja uravnoteženosti 100%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.647	0.559	0.608	0.577
podvzorčenje	0.649	0.669	0.650	0.674
nadvzorčenje	0.768	0.663	0.790	0.690
SMOTE	0.758	0.653	0.767	0.700
SMOTER ASC	0.722	0.658	0.768	0.705
SMOTER DESC	0.747	0.658	0.767	0.698
SMOTERAND	0.758	0.670	0.777	0.698

Tabela A.21: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "bupa" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.635	0.637	0.653	0.610
podvzorčenje	0.658	0.624	0.660	0.505
nadvzorčenje	0.762	0.663	0.904	0.648
SMOTE	0.757	0.668	0.888	0.676
SMOTER ASC	0.675	0.633	0.886	0.267
SMOTER DESC	0.757	0.691	0.869	0.275
SMOTERAND	0.732	0.679	0.898	0.649

Tabela A.22: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "bupa" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.707	0.649	0.635	0.708
podvzorčenje	0.672	0.663	0.651	0.697
nadvzorčenje	0.708	0.675	0.799	0.709
SMOTE	0.735	0.665	0.751	0.716
SMOTER ASC	0.746	0.681	0.747	0.729
SMOTER DESC	0.725	0.668	0.749	0.711
SMOTERAND	0.731	0.659	0.767	0.712

Tabela A.23: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "cmc" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.707	0.684	0.648	0.756
podvzorčenje	0.696	0.689	0.623	0.714
nadvzorčenje	0.729	0.686	0.891	0.755
SMOTE	0.715	0.677	0.800	0.769
SMOTER ASC	0.709	0.692	0.790	0.261
SMOTER DESC	0.750	0.674	0.810	0.240
SMOTERAND	0.715	0.674	0.855	0.771

Tabela A.24: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "cmc" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.849	0.770	0.858	0.794
podvzorčenje	0.848	0.735	0.831	0.878
nadvzorčenje	0.891	0.752	0.910	0.894
SMOTE	0.898	0.748	0.887	0.900
SMOTER ASC	0.896	0.741	0.896	0.896
SMOTER DESC	0.898	0.769	0.898	0.902
SMOTERAND	0.897	0.750	0.899	0.896

Tabela A.25: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "crx" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.885	0.881	0.913	0.886
podvzorčenje	0.886	0.887	0.874	0.0965
nadvzorčenje	0.882	0.899	0.968	0.930
SMOTE	0.890	0.898	0.936	0.938
SMOTER ASC	0.881	0.892	0.931	0.932
SMOTER DESC	0.904	0.904	0.941	0.0609
SMOTERAND	0.916	0.900	0.947	0.939

Tabela A.26: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "crx" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.719	0.748	0.670	0.732
podvzorčenje	0.651	0.587	0.643	0.654
nadvzorčenje	0.790	0.635	0.837	0.685
SMOTE	0.793	0.650	0.827	0.687
SMOTER ASC	0.806	0.643	0.819	0.681
SMOTER DESC	0.793	0.635	0.809	0.692
SMOTERAND	0.808	0.634	0.845	0.680

Tabela A.27: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "haberman" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.649	0.639	0.599	0.411
podvzorčenje	0.661	0.581	0.540	0.631
nadvzorčenje	0.744	0.663	0.936	0.634
SMOTE	0.816	0.654	0.877	0.682
SMOTER ASC	0.828	0.656	0.864	0.651
SMOTER DESC	0.814	0.654	0.853	0.674
SMOTERAND	0.798	0.670	0.927	0.627

Tabela A.28: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "haberman" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.787	0.845	0.844	0.845
podvzorčenje	0.855	0.765	0.835	0.760
nadvzorčenje	0.913	0.835	0.881	0.902
SMOTE	0.865	0.913	0.916	0.941
SMOTER ASC	0.902	0.918	0.924	0.940
SMOTER DESC	0.900	0.927	0.927	0.943
SMOTERAND	0.894	0.932	0.927	0.948

Tabela A.29: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "hepatitis" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.744	0.853	0.845	0.171
podvzorčenje	0.741	0.758	0.871	0.805
nadvzorčenje	0.896	0.905	0.988	0.0299
SMOTE	0.841	0.947	0.984	0.0257
SMOTER ASC	0.887	0.945	0.985	0.875
SMOTER DESC	0.895	0.943	0.984	0.984
SMOTERAND	0.866	0.952	0.983	0.0263

Tabela A.30: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "hepatitis" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.812	0.827	0.801	0.819
podvzorčenje	0.823	0.834	0.808	0.840
nadvzorčenje	0.837	0.838	0.864	0.848
SMOTE	0.844	0.853	0.852	0.855
SMOTER ASC	0.844	0.842	0.862	0.853
SMOTER DESC	0.848	0.866	0.865	0.855
SMOTERAND	0.857	0.856	0.859	0.845

Tabela A.31: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "mammographic masses" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.840	0.901	0.860	0.102
podvzorčenje	0.864	0.896	0.856	0.107
nadvzorčenje	0.867	0.904	0.916	0.0976
SMOTE	0.862	0.914	0.912	0.0891
SMOTER ASC	0.866	0.903	0.899	0.0929
SMOTER DESC	0.879	0.927	0.920	0.0836
SMOTERAND	0.871	0.918	0.925	0.0897

Tabela A.32: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "mammographic masses" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.882	0.692	0.927	0.871
podvzorčenje	0.816	0.811	0.834	0.795
nadvzorčenje	0.927	0.839	0.937	0.921
SMOTE	0.909	0.832	0.961	0.900
SMOTER ASC	0.932	0.835	0.950	0.903
SMOTER DESC	0.900	0.823	0.946	0.907
SMOTERAND	0.934	0.828	0.964	0.907

Tabela A.33: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "parkinsons" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.871	0.854	0.986	0.0918
podvzorčenje	0.855	0.877	0.928	0.861
nadvzorčenje	0.921	0.876	0.999	0.0267
SMOTE	0.902	0.872	0.998	0.0342
SMOTER ASC	0.960	0.878	0.998	0.969
SMOTER DESC	0.919	0.872	0.997	0.0292
SMOTERAND	0.937	0.882	0.998	0.027

Tabela A.34: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "parkinsons" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.728	0.755	0.738	0.736
podvzorčenje	0.729	0.721	0.759	0.726
nadvzorčenje	0.817	0.745	0.860	0.795
SMOTE	0.819	0.773	0.853	0.798
SMOTER ASC	0.810	0.781	0.859	0.808
SMOTER DESC	0.806	0.770	0.849	0.807
SMOTERAND	0.812	0.772	0.867	0.810

Tabela A.35: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "pima-indians-diabetes" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.777	0.812	0.776	0.288
podvzorčenje	0.755	0.789	0.777	0.231
nadvzorčenje	0.809	0.820	0.953	0.213
SMOTE	0.794	0.827	0.928	0.223
SMOTER ASC	0.801	0.832	0.938	0.226
SMOTER DESC	0.812	0.828	0.930	0.224
SMOTERAND	0.816	0.825	0.947	0.211

Tabela A.36: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "pima-indians-diabetes" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.689	0.633	0.700	0.733
podvzorčenje	0.558	0.567	0.383	0.617
nadvzorčenje	0.712	0.642	0.767	0.777
SMOTE	0.808	0.778	0.814	0.792
SMOTER ASC	0.738	0.793	0.695	0.828
SMOTER DESC	0.787	0.798	0.809	0.793
SMOTERAND	0.812	0.783	0.838	0.817

Tabela A.37: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "post-operative" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.358	0.372	0.319	0.436
podvzorčenje	0.321	0.441	0.0938	0.384
nadvzorčenje	0.625	0.544	0.875	0.719
SMOTE	0.780	0.853	0.870	0.868
SMOTER ASC	0.724	0.871	0.802	0.777
SMOTER DESC	0.840	0.848	0.809	0.829
SMOTERAND	0.779	0.854	0.895	0.786

Tabela A.38: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "post-operative" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.811	0.852	0.833	0.785
podvzorčenje	0.800	0.828	0.750	0.817
nadvzorčenje	0.856	0.844	0.896	0.884
SMOTE	0.849	0.887	0.887	0.873
SMOTER ASC	0.842	0.891	0.878	0.869
SMOTER DESC	0.858	0.887	0.887	0.882
SMOTERAND	0.856	0.884	0.880	0.880

Tabela A.39: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "statlog heart" (stopnja uravnoteženosti 200%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.810	0.911	0.880	0.229
podvzorčenje	0.830	0.909	0.860	0.125
nadvzorčenje	0.874	0.906	0.963	0.143
SMOTE	0.879	0.940	0.945	0.158
SMOTER ASC	0.882	0.937	0.941	0.842
SMOTER DESC	0.887	0.943	0.941	0.0658
SMOTERAND	0.874	0.943	0.946	0.144

Tabela A.40: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "statlog heart" (stopnja uravnoteženosti 200%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.679	0.565	0.609	0.583
podvzorčenje	0.762	0.761	0.806	0.806
nadvzorčenje	0.848	0.766	0.842	0.826
SMOTE	0.817	0.765	0.847	0.814
SMOTER ASC	0.835	0.767	0.848	0.817
SMOTER DESC	0.846	0.765	0.848	0.828
SMOTERAND	0.829	0.755	0.869	0.822

Tabela A.41: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "bupa" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.671	0.635	0.652	0.558
podvzorčenje	0.504	0.667	0.684	0.322
nadvzorčenje	0.758	0.645	0.932	0.706
SMOTE	0.740	0.674	0.927	0.702
SMOTER ASC	0.742	0.642	0.935	0.251
SMOTER DESC	0.767	0.693	0.927	0.769
SMOTERAND	0.780	0.701	0.947	0.695

Tabela A.42: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "bupa" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.712	0.650	0.601	0.713
podvzorčenje	0.794	0.772	0.772	0.796
nadvzorčenje	0.810	0.788	0.871	0.807
SMOTE	0.829	0.786	0.845	0.815
SMOTER ASC	0.834	0.784	0.847	0.820
SMOTER DESC	0.817	0.798	0.849	0.814
SMOTERAND	0.826	0.784	0.860	0.811

Tabela A.43: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "cmc" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.723	0.683	0.631	0.756
podvzorčenje	0.676	0.696	0.631	0.656
nadvzorčenje	0.631	0.682	0.920	0.767
SMOTE	0.730	0.681	0.862	0.789
SMOTER ASC	0.719	0.684	0.860	0.256
SMOTER DESC	0.698	0.679	0.856	0.785
SMOTERAND	0.739	0.680	0.920	0.781

Tabela A.44: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "cmc" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.854	0.775	0.858	0.799
podvzorčenje	0.878	0.747	0.870	0.864
nadvzorčenje	0.923	0.739	0.939	0.916
SMOTE	0.931	0.783	0.939	0.934
SMOTER ASC	0.931	0.780	0.934	0.931
SMOTER DESC	0.931	0.773	0.935	0.933
SMOTERAND	0.934	0.790	0.941	0.935

Tabela A.45: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "crx" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.886	0.884	0.906	0.888
podvzorčenje	0.890	0.885	0.906	0.118
nadvzorčenje	0.866	0.890	0.985	0.933
SMOTE	0.876	0.908	0.952	0.950
SMOTER ASC	0.895	0.910	0.956	0.947
SMOTER DESC	0.887	0.905	0.952	0.0534
SMOTERAND	0.887	0.908	0.966	0.947

Tabela A.46: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "crx" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.715	0.745	0.683	0.735
podvzorčenje	0.773	0.772	0.713	0.813
nadvzorčenje	0.882	0.800	0.884	0.800
SMOTE	0.868	0.802	0.879	0.800
SMOTER ASC	0.863	0.800	0.877	0.800
SMOTER DESC	0.872	0.801	0.880	0.802
SMOTERAND	0.861	0.801	0.909	0.800

Tabela A.47: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "haberman" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.580	0.585	0.613	0.401
podvzorčenje	0.559	0.521	0.582	0.388
nadvzorčenje	0.758	0.656	0.974	0.511
SMOTE	0.839	0.647	0.894	0.561
SMOTER ASC	0.825	0.651	0.900	0.480
SMOTER DESC	0.826	0.670	0.899	0.518
SMOTERAND	0.798	0.663	0.954	0.610

Tabela A.48: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "haberman" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.794	0.839	0.818	0.872
podvzorčenje	0.925	0.775	0.800	0.800
nadvzorčenje	0.932	0.862	0.927	0.976
SMOTE	0.927	0.950	0.951	0.966
SMOTER ASC	0.951	0.955	0.954	0.966
SMOTER DESC	0.942	0.953	0.951	0.966
SMOTERAND	0.945	0.964	0.959	0.971

Tabela A.49: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "hepatitis" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.699	0.853	0.843	0.162
podvzorčenje	0.828	0.590	0.824	0.680
nadvzorčenje	0.880	0.903	1.000	0.0157
SMOTE	0.871	0.950	0.993	0.0246
SMOTER ASC	0.899	0.948	0.994	0.874
SMOTER DESC	0.895	0.951	0.993	0.978
SMOTERAND	0.934	0.965	0.992	0.0226

Tabela A.50: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "hepatitis" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.815	0.824	0.797	0.815
podvzorčenje	0.851	0.870	0.858	0.847
nadvzorčenje	0.892	0.880	0.923	0.857
SMOTE	0.891	0.878	0.909	0.867
SMOTER ASC	0.899	0.872	0.907	0.860
SMOTER DESC	0.902	0.883	0.908	0.863
SMOTERAND	0.898	0.879	0.916	0.859

Tabela A.51: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "mammographic masses" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.841	0.901	0.860	0.102
podvzorčenje	0.852	0.876	0.863	0.155
nadvzorčenje	0.894	0.908	0.921	0.130
SMOTE	0.899	0.922	0.927	0.102
SMOTER ASC	0.896	0.909	0.919	0.115
SMOTER DESC	0.907	0.928	0.935	0.895
SMOTERAND	0.900	0.922	0.945	0.103

Tabela A.52: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "mammographic masses" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.866	0.703	0.929	0.866
podvzorčenje	0.867	0.867	0.867	0.850
nadvzorčenje	0.967	0.868	0.952	0.931
SMOTE	0.942	0.875	0.965	0.932
SMOTER ASC	0.958	0.872	0.962	0.933
SMOTER DESC	0.955	0.879	0.963	0.932
SMOTERAND	0.948	0.882	0.971	0.928

Tabela A.53: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "parkinsons" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.854	0.858	0.985	0.101
podvzorčenje	0.579	0.833	0.885	0.533
nadvzorčenje	0.936	0.866	1.000	0.0332
SMOTE	0.940	0.882	0.999	0.0324
SMOTER ASC	0.960	0.875	1.000	0.874
SMOTER DESC	0.950	0.880	1.000	0.0274
SMOTERAND	0.939	0.896	0.999	0.041

Tabela A.54: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "parkinsons" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.758	0.759	0.742	0.732
podvzorčenje	0.820	0.806	0.827	0.824
nadvzorčenje	0.889	0.819	0.884	0.869
SMOTE	0.883	0.846	0.894	0.872
SMOTER ASC	0.882	0.853	0.894	0.878
SMOTER DESC	0.880	0.842	0.898	0.870
SMOTERAND	0.882	0.857	0.922	0.869

Tabela A.55: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "pima-indians-diabetes" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.785	0.814	0.783	0.272
podvzorčenje	0.729	0.807	0.776	0.221
nadvzorčenje	0.812	0.823	0.982	0.205
SMOTE	0.803	0.835	0.967	0.208
SMOTER ASC	0.783	0.836	0.966	0.136
SMOTER DESC	0.789	0.827	0.970	0.206
SMOTERAND	0.824	0.835	0.974	0.223

Tabela A.56: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "pima-indians-diabetes" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.711	0.633	0.678	0.733
podvzorčenje	0.800	0.567	0.667	0.767
nadvzorčenje	0.806	0.812	0.839	0.900
SMOTE	0.882	0.848	0.855	0.891
SMOTER ASC	0.842	0.858	0.833	0.882
SMOTER DESC	0.900	0.882	0.879	0.909
SMOTERAND	0.870	0.848	0.897	0.870

Tabela A.57: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "post-operative" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.292	0.374	0.324	0.600
podvzorčenje	0.0625	0.562	0.417	0.444
nadvzorčenje	0.549	0.548	0.905	0.770
SMOTE	0.842	0.900	0.919	0.815
SMOTER ASC	0.808	0.895	0.872	0.838
SMOTER DESC	0.889	0.888	0.856	0.114
SMOTERAND	0.849	0.887	0.945	0.849

Tabela A.58: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "post-operative" (stopnja uravnoteženosti 400%)

algoritem	CA(odl. drevo)	CA(bayes)	CA(kNN)	CA(SVM)
original	0.819	0.859	0.833	0.800
podvzorčenje	0.873	0.853	0.833	0.800
nadvzorčenje	0.884	0.849	0.917	0.928
SMOTE	0.888	0.908	0.929	0.915
SMOTER ASC	0.896	0.915	0.933	0.913
SMOTER DESC	0.889	0.907	0.925	0.913
SMOTERAND	0.899	0.915	0.929	0.927

Tabela A.59: Klasifikacijske točnosti vseh klasifikatorjev na podatkovni množici "statlog heart" (stopnja uravnoteženosti 400%)

algoritem	AUC(odl. drevo)	AUC(bayes)	AUC(kNN)	AUC(SVM)
original	0.827	0.909	0.891	0.114
podvzorčenje	0.762	0.876	0.880	0.212
nadvzorčenje	0.839	0.913	0.983	0.147
SMOTE	0.884	0.950	0.955	0.146
SMOTER ASC	0.897	0.952	0.957	0.853
SMOTER DESC	0.895	0.948	0.957	0.0494
SMOTERAND	0.904	0.954	0.971	0.138

Tabela A.60: Vrednosti AUC vseh klasifikatorjev na podatkovni množici "statlog heart" (stopnja uravnoteženosti 400%)