Univerza v Ljubljani
Fakulteta za računalništvo in informatiko
Fakulteta za matematiko in fiziko

Jerneja Mislej

Analiza signalov membranske kapacitivnosti živih celic

diplomsko delo
univerzitetni študijski program prve stopnje
računalništvo in matematika

Mentor: prof. dr. Igor Kononenko
Somentor: prof. dr. Marko Kreft

Ljubljana 2015

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

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 podpisana Jerneja Mislej, z vpisno številko **6090452**, sem avtorica diplomskega dela z naslovom:

Analiza signalov membranske kapacitivnosti živih celic

S svojim podpisom zagotavljam, da:

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

V Ljubljani, dne 15. februarja 2015 Podpis avtorja:

Kazalo

Povzetek

Namen diplomskega dela je (pol)avtomatska detekcija skokovitih sprememb v signalu membranske kapacitivnosti celic, in sicer sprememb, ki so posledice eksocitoze ali endocitoze. Ustrezne spremembe v signalu so v preteklosti že detektirali avtomatsko, in sicer s pomočjo matematične analize signala, vendar pa je matematična analiza možna le, če je razmerje med signalom in šumom dovolj veliko. Pri nekaterih tipih celic so amplitude iskanih sprememb tako majhne, da se skoraj popolnoma skrijejo v šumu in preprosta matematična analiza ni več možna. Posledično detekcija ustreznih skokovitih sprememb trenutno poteka ročno. Označenih je bilo že kar nekaj signalov in tako se ponuja možnost olajšanja ročnega pregledovanja s pomočjo strojnega učenja, kar je delo te diplomske naloge.
Podatki se na začetku predobdelajo, pri čemer se iz meritve odstranijo kalibracijski pulzi, obsežno šumni predeli meritve ter popravi bazna linija signala, ki predstavlja meritev. Sledi priprava množice podatkov, ki zajema ekstrakcijo učnih primerov, izračun atributov učnih primerov in prevzorčenje učne množice. Zatem se na podlagi učne in testne množice ter različnih metod strojnega učenja predstavijo in primerjajo rezultati klasifikacij novih primerov.

Ključne besede
Eksocitoza, endocitoza, klasifikacija, membranska kapacitivnost celice, signal, skokovita sprememba, strojno učenje, šum.

Abstract

The purpose of the thesis is a (semi)automatic detection of rapid changes in the cell membrane capacitance signal, namely changes that are a consequence of exocytosis and endocytosis. Corresponding changes in the signal were previously detected automatically, using mathematical signal analysis. However, mathematical analysis is only possible when the signal-to-noise ratio is sufficiently large. In some cell types the amplitude of the change is so small that it is almost completely hidden in the noise, therefore simple mathematical analysis is no longer possible. Consequently, the detection of the corresponding rapid changes is currently done manually. A number of signals have been labeled and so offers a possibility of facilitating the manual inspection by applying machine learning, which is the work of this thesis. The data is initially pre-processed, where calibration pulses and areas with extensive noise are removed from the measurements and the baseline of the signal representing the measurement is corrected. This is followed by a preparation of the dataset, which includes the extraction of instances, calculation of the attributes of instances and resampling of the training dataset. Then, based on the training set, test set and a variety of machine learning methods, results of the classifications of new cases are presented and compared.

Keywords

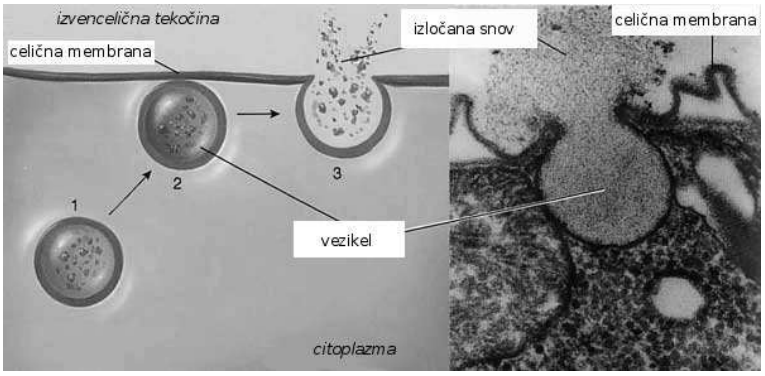
Exocytosis, endocytosis, cell membrane capacitance, classification, signal, rapid change, machine learning, noise.

Poglavje 1

Uvod

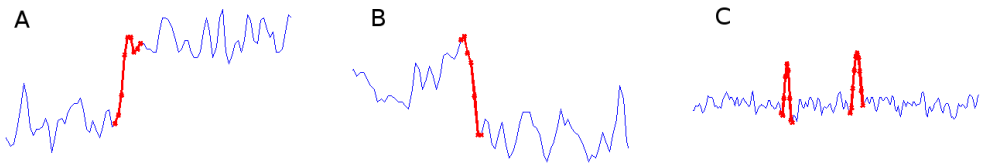
1.1 Motivacija

Endocitoza in eksocitoza sta procesa, ki sodelujeta pri mnogih celičnih funkcijah. Njuna glavna zadolžitev je transport tovora v celico in iz nje, eksocitoza, ki jo prikazuje Slika 1.1, pa poleg tega sodeluje tudi pri obnavljanju membrane in translokaciji membranskih receptorjev ter drugih proteinov k membrani. Posledično endocitoza in eksocitoza pri živih bitjih predstavljata izredno pomemben signalni in regulatorni mehanizem ter sta zato predmet neštetih raziskav.



Slika 1.1: Grafičen prikaz eksocitoze poleg resničnega posnetka

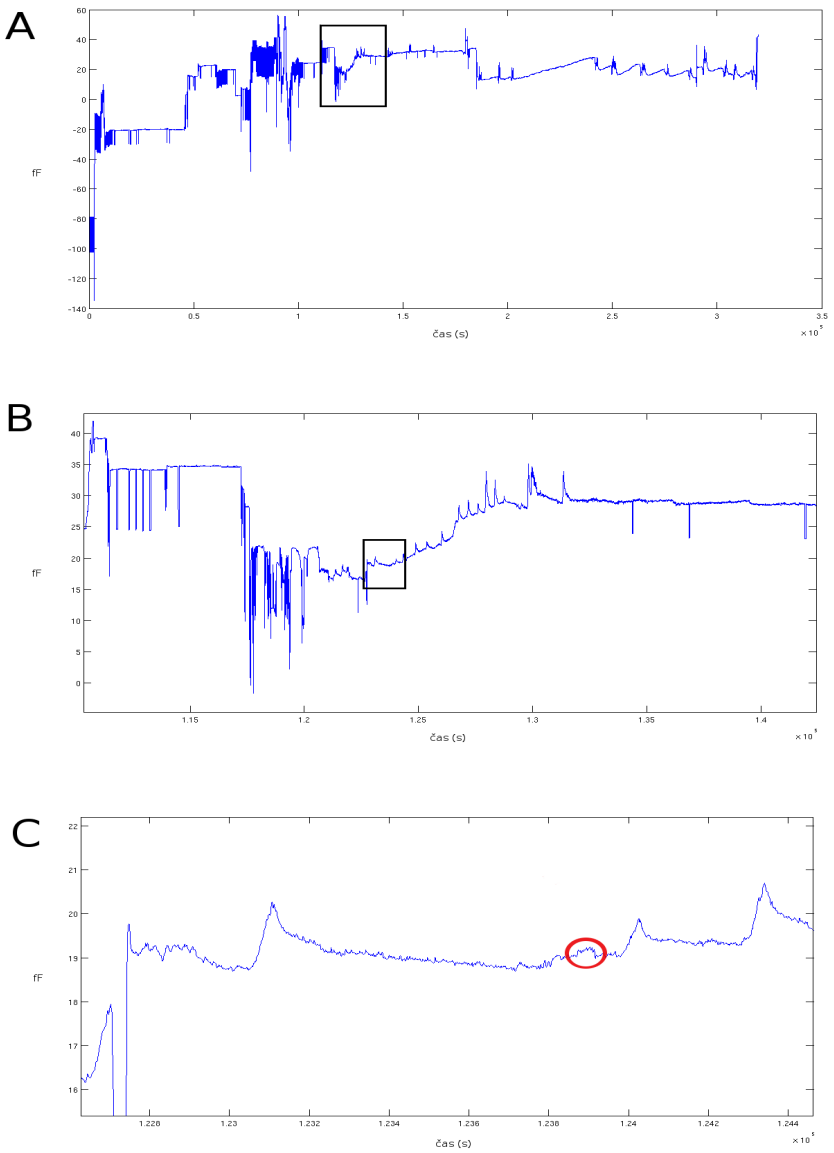
Merjenje membranske kapacitivnosti celice (Cm) predstavlja eleganten način raziskovanja endocitoze in eksocitoze, saj je ta parameter sorazmeren s celično površino in sicer $1mF/cm^2$. Celična površina pa se med fuzijo pri eksocitozi in endocitozi poveča oziroma zmanjša. Slika 1.2 prikazuje skokovito spremembo Cm zaradi eksocitoze in endocitoze. Teoretično je pozitivna sprememba Cm posledica eksocitoze, negativna sprememba pa posledica endocitoze. Pogosto pa v meritvi pride do prehodnega oziroma reverzibilnega eksocitotičnega dogodka, kjer se vezikel ne izlije popolnoma. Ena ali več takih delnih fuzij privedejo do tipičnih skokovitih sprememb v Cm, kjer pozitivni spremembi v zelo kratkem času sledi negativna sprememba iste velikosti.



Slika 1.2: Označena sprememba na primeru A je posledica eksocitoze, označena sprememba na primeru B je posledica endocitoze, označeni spremembi na primeru C pa sta posledici zaporedja prehodnih eksocitoz.

Pod okriljem Medicinske fakultete Univerze v Ljubljani, v Laboratoriju za nevroendokrinologijo, na področju molekularne celične fiziologije, že mnoga leta s pomočjo elektrofizioloških metod uspešno merijo celično membransko kapacitivnost. Dovolj velike skokovite spremembe Cm, kot so na primer spremembe, ki so posledice interakcij med sekrecijskimi vezikli in celično membrano, so merjene v velikosti reda femtofaradov. Pri takih meritvah je razmerje med signalom in šumom dovolj veliko, da se iskane spremembe lahko alocirajo s pomočjo preproste matematične analize [1]. Pri opazovanju konstitutivnega prometa veziklov in pri nekaterih tipih celic pa so spremembe Cm merjene v atofaradih in so tako majhne, da se popolnoma skrijejo v šumu [2], [3]. Slika 1.3 prikazuje iskano spremembo, skrito v šumu. Šum, ki

je vseskozi prisoten, je posledica elektrofizioloških naprav, hkrati pa tudi še neraziskanih in nepredvidljivih dogajanj znotraj ter zunaj celice.



Slika 1.3: Signal na sliki A predstavlja celotno meritev Cm, označeni del signala je povečan na sliki B, katerega označeni del sledi povečan na sliki C, kjer je z rdečo obkrožena iskana ustrezna sprememba.

To onemogoča preprosto odstranitev šuma, zaradi česar je matematična analiza signala v takem primeru neuspešna pri alociranju ustreznih skokovitih sprememb. Posledično detekcija skokovitih sprememb tako majhnega velikostnega reda poteka ročno. Operater se pri pregledu signala na podlagi različnih karakteristik spremembe in njene okolice odloči, ali je sprememba signala posledica raziskovanega dogodka ali le posledica šuma. Te karakteristike vključujejo naklon spremembe, velikost spremembe, razmerje med signalom in šumom v okolici spremembe, pojav spremembe obratnega predznaka v kratkem časovnem intervalu itd. Na meritev se pojavi od 20 do dobrih 100 iskanih dogodkov, v povprečju 50. Poleg tega, da so spremembe, ki jih ti dogodki povzročijo, skrite v šumu, so pogosto zaradi njega tudi izkrivljene, kar jih naredi še bolj težko opazne.
Posledično je ročno pregledovanje signala zelo zamudno, naporno in nagnjeno k nepopolnosti. Potreba po avtomatizaciji in računalniški natančnosti skupaj z množico že označenih meritev ponuja možnost rešitve problema s pomočjo strojnega učenja.

1.2 Cilji

Cilj diplomskega dela je konstrukcija sistema, ki bo na podlagi lastnosti množice že označenih meritev s pomočjo ustreznih algoritmov zmožen klasificirati spremembe signala v novih meritvah Cm. Osnova takega sistema je množica podatkov, ki vsebuje primere s svojimi lastnostmi. Ker pa je meritev Cm le zaporedje diskretnih vrednosti, je treba iz množice vrednosti določiti, kaj bo tisto, kar bo v množici predstavljalo posamezen primer. Glede na to, da mora sistem klasificirati skokovite spremembe, je najbolj smiselno in priročno, da se kot primeri določijo prav take spremembe v signalu. Take spremembe se pojavijo, kjer je razlika med zaporednimi diskretnimi vrednostmi dovolj velika. Da pa bi bila ekstrakcija teh sprememb čim bolj preprosta in smiselna, je treba podatke predhodno pripraviti oziroma predobdelati. Meritve namreč vsebujejo veliko šumnih delov, ki so posledice

elektrofizioloških metod ali različnih dogajanj znotraj in zunaj celice. Moteča je tudi bazna linija signala, ki se konstantno in relativno na velikost iskanih sprememb, drastično spreminja.

Po predobdelavi podatkov se pod določenimi pogoji lahko izločijo skokovite spremembe v signalu, ki predstavljajo primere v množici podatkov. To množico podatkov je za namen ocenjevanja strojnega učenja treba razdeliti v učno in testno množico. Učna množica je tista, s pomočjo katere učni algoritem zgradi model za klasifikacijo. Na podlagi testne množice pa se lahko na koncu ocenijo rezultati izvajalnega algoritma. Ker pa je nezmožnost matematične analize prvotni problem, je končni rezultat ekstrakcije skokovitih sprememb prav tako nepopoln. Zaradi relativno velike variance med meritvami in v meritvi sami ter zaradi konstantnega relativno obsežnega šuma množica vsebuje izredno veliko negativno klasificiranih primerov. Neuravnotežena učna množica predstavlja problem pri strojnem učenju. Učni algoritem namreč posledično zanemari manjšinski razred, saj je število napačno klasificiranih primerov manjšinskega razreda zanemarljivo v primerjavi s številom potencialno napačno klasificiranih primerov večinskega razreda. Da bo model pravilen, je treba učno množico prevzorčiti, in sicer tako, da bo število pozitivno označenih primerov primerljivo s številom negativno označenih primerov. Na podlagi take množice lahko učni algoritem pravilno zgradi model, s pomočjo katerega izvajalni algoritem opravi dejansko klasifikacijo. Da bo učni algoritem zmožen zgraditi tak model, je treba primerom prirediti njihove atribute ali značilke. Ti atributi morajo čim bolje opisati lastnosti vseh sprememb, tako da bodo do izraza prišle unikatne značilnosti iskanih skokovitih sprememb. Klasifikacija, ki jo mora opraviti izvajalni algoritem, je zaradi preprostosti binarna. Izvajalni algoritem mora skokovito spremembo klasificirati kot posledico raziskovanega dogodka, torej pozitivno, ali pa kot posledico šuma, torej negativno. Klasifikacije ročno pregledanih vzorcev so shranjene v tekstovni datoteki kot koordinate x skokovitih sprememb. Te podatke je treba preslikati v prostor primerov in njihovim atributom dodati še en stolpec ničel in enk, ki prikazujejo negativno oziroma

pozitivno klasificirane primere. S tem je množica podatkov pripravljena na modeliranje.

Modeliranje opravi učni algoritem. Obstaja mnogo različnih vrst in podvrst strojnega učenja. Vsaka zase na neki način vsebuje učni algoritem in izvajalni algoritem, ki sta določena z izbiro metode. Izbira metode je pomemben del načrtovanja strojnega učenja za klasifikacijo skokovitih sprememb. Upoštevati je treba cilj in ocene strojnega učenja. Pri testiranju je treba izračunati več različnih ocen strojnega učenja in izbrati metodo, pri kateri imajo vse ocene zadovoljivo vrednost. Po testiranju in oceni strojnega učenja sledi zaključek celotnega diplomskega dela.

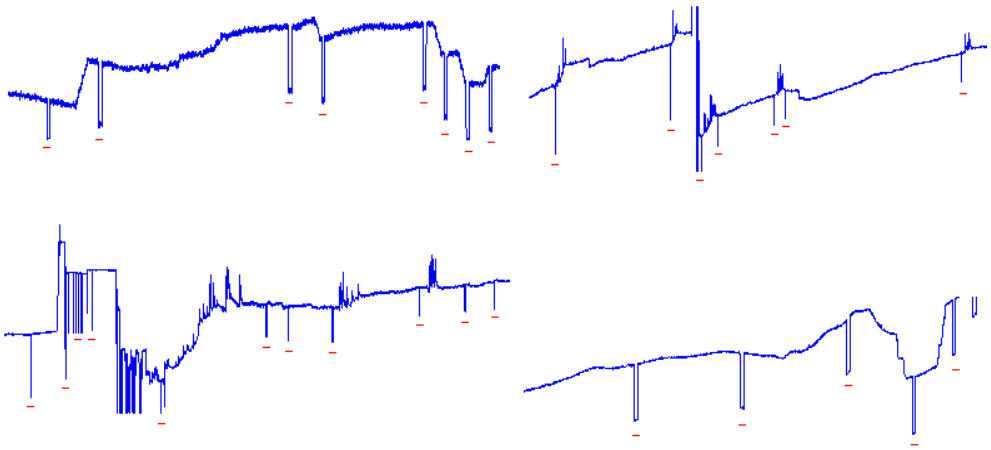
Poglavje 2

Predobdelava podatkov meritve membranske kapacitivnosti

Zaradi posledic elektrofizioloških metod in naključnih dogajanj znotraj ter zunaj celice je meritev Cm zelo šumna in težko pregledna. Meritev poleg vseskozi prisotnega šuma vsebuje predele, kjer je šum zelo obsežen in popolnoma prekrije meritev. Ti predeli se pojavijo naključno in imajo naključno velikost ter naključen čas trajanja. Skokovite spremembe, ki se iščejo, pa so tako majhne, da se popolnoma skrijejo v šumu, zaradi česar je glajenje signala neučinkovito. Zaradi izredno majhnega velikostnega reda iskanih sprememb pa je moteča tudi vseskozi spreminjajoča se bazna linija signala. Da bi učni algoritem čim bolje zgradil model, ki ga za klasifikacijo potrebuje izvajalni algoritem, morajo biti atributi, ki pripadajo učnim primerom, indiferentni za vse spremembe v meritvi Cm, ki ne odražajo realnega dogajanja v celici.
Predobdelava podatkov je implementirana v programskem jeziku Matlab. Kjer je bilo možno, sem uporabila vgrajene Matlabove funkcije, glavne funkcije in nekaj pomožnih pa sem razvila sama.

2.1 Odstranitev kalibracijskih pulzov

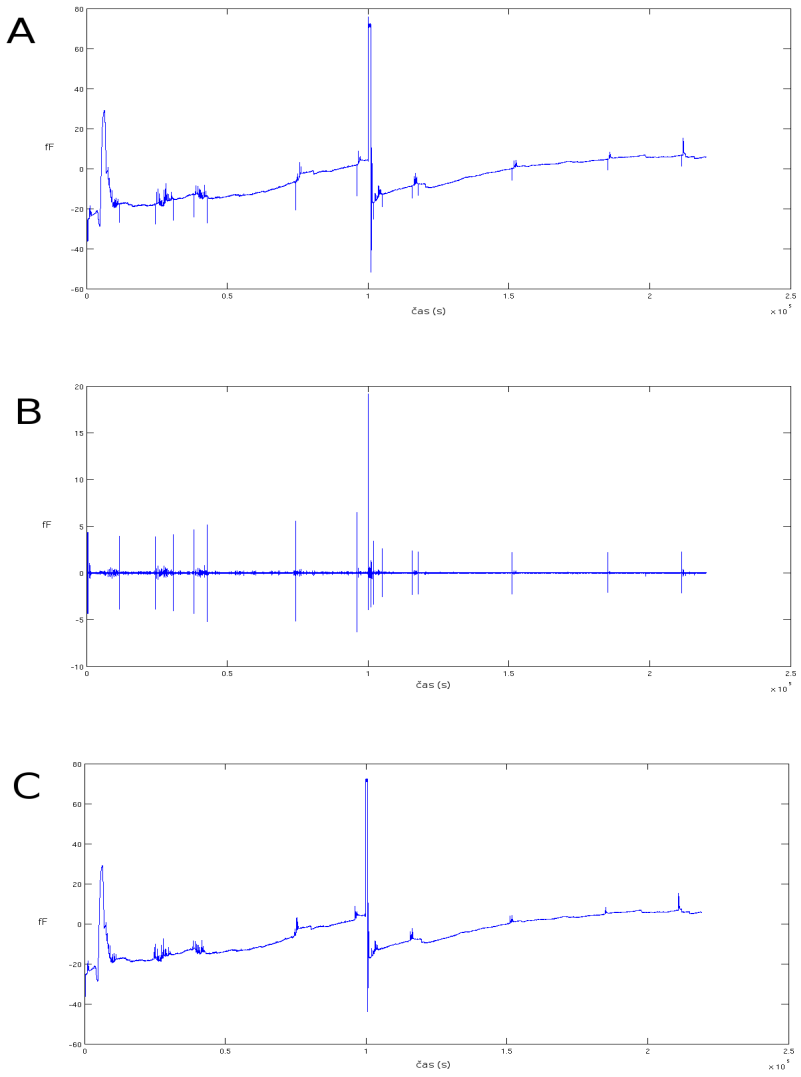
Kalibracijski pulzi so ponavljajoče se navpične spremembe v velikostnem redu femtofaradov. Generirane so ročno, za potrebe po kalibraciji ojačevalnika in za preverjanje faznega kota ojačevalnika. Velikost in dolžina kalibracijskih pulzov rahlo variirata znotraj meritve ter med posameznimi različnimi meritvami. Slika 2.1 prikazuje primere različnih kalibracijskih pulzov v različnih skalah. Čeprav pulzi ne predstavljajo velike ovire pri ekstrakciji učnih primerov in gradnji modela, jih je vseeno priporočljivo odstraniti za boljšo preglednost in nadaljnjo predobdelavo podatkov.



Slika 2.1: Kalibracijski pulzi v različnih meritvah Cm.

Odstranitev kalibracijskih pulzov se opravi v funkciji *remove_calibration_pulses*, ki za vhod prejme meritev Cm, meje intervala tolerance velikosti in dolžino kalibracijskega pulza, za izhod pa vrne nov signal brez kalibracijskih pulzov. Slika 2.2 prikazuje meritev Cm, numerični odvod meritve Cm in meritev Cm brez kalibracijskih pulzov. Posamezne meritve Cm se med seboj razlikujejo, prav tako se med seboj razlikujejo kalibracijski pulzi med različnimi meritvami in v meritvi sami. Posledično mora biti funkcija, ki odstranjuje kalibracijske pulze, kar se da invariantna na velikost in dolžino kalibracijskega pulza ter velikost različnih razmikov med posameznimi pulzi. V ta namen sta na voljo vhodna parametra za meje intervala

tolerance velikosti in dolžino kalibracijskega pulza, katerih privzeta vrednost je [1,7] femtofaradov ter 125 enot. Ti dve vrednosti v kombinaciji z invariantnostjo funkcije *remove_calibration_pulses* zadostujeta za odstranitev vseh pulzov v vseh meritvah učne množice.

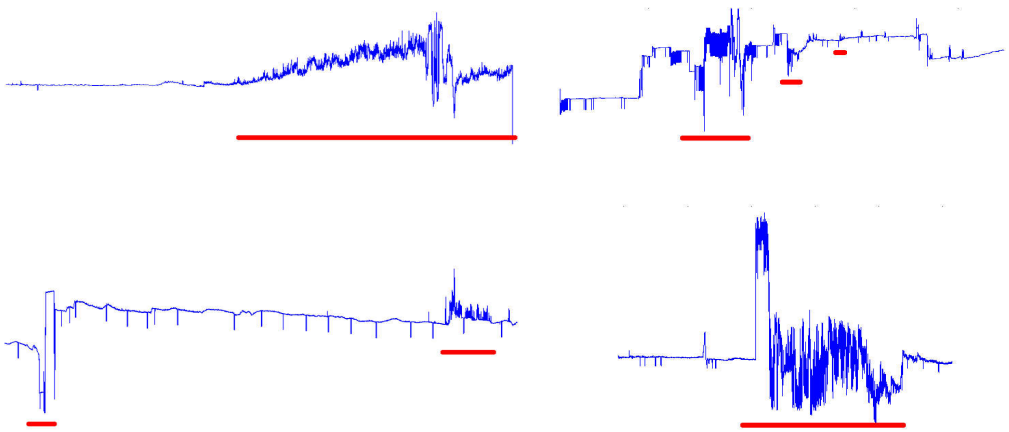


Slika 2.2: Slika A predstavlja meritev Cm, slika B predstavlja njen numerični odvod, slika C pa predstavlja meritev Cm brez kalibracijskih pulzov.

V primeru drastične spremembe prihodnjih meritev lahko uporabnik parametra nastavi na bolj ustrezno vrednost.
Vhodni signal je v Matlabu predstavljen kot vektor realnih števil. Za alokacijo vseh sprememb, katerih velikosti so v intervalu tolerance, je treba izračunati diference med zaporednimi vrednostmi. To opravi Matlabova vgrajena funkcija *diff*, ki od leve proti desni odšteje sosednje vrednosti in vrne vektor razlik. Kjer so razlike skokovito velike, se med podatki pojavi vrh. Razlike, ki so večje od spodnje meje intervala tolerance, se poiščejo s pomočjo Matlabove vgrajene funkcije *findpeaks*, ki v signalu poišče vrhove. Funkcija za vhod prejme vektor razlik, opcijo *'MINPEAKHEIGHT'* in parameter za to opcijo, ki zavzema vrednost spodnje meje intervala tolerance. Funkcija poišče vse vrhove, večje od spodnje meje intervala tolerance, tako da s pomočjo predznaka diferenc vhodnega signala poišče ravno območje oziroma območje, kjer se spremeni zaporedje enakih predznakov diferenc. Funkcija išče samo pozitivne vrhove, ki v vektorju diferenc predstavljajo pozitivno skokovito spremembo. Za alokacijo negativnih sprememb se vektor razlik množi z -1 in ponovno kliče funkcija *findpeaks*. Naknadno se odstranijo še spremembe, večje od zgornje meje intervala tolerance. Nato se v zanki *for*, ki gre čez vse lokacije x negativnih sprememb, vsaka posamezna lokacija odšteje od vektorja x lokacij pozitivnih sprememb in se najde tista, ki je manjša od vrednosti parametra za dolžino kalibracijskega pulza. Na koncu se v vhodnem signalu odstranijo vsa območja med takima dvema spremembama. Rezultat je izhodni signal funkcije *remove_calibration_pulses*, ki predstavlja meritev Cm brez kalibracijskih pulzov. Za meritve, ki spadajo pod učno množico, je treba upoštevati klasifikacije, ki jim pripadajo.
V ta namen sem modificirala zgoraj omenjeno funkcijo v funkcijo *remove_calibration_pulses_with_classifications*, ki ustrezno prilagodi klasifikacije po odstranitvi kalibracijskih pulzov.

2.2 Odstranjevanje šumnih delov posnetka

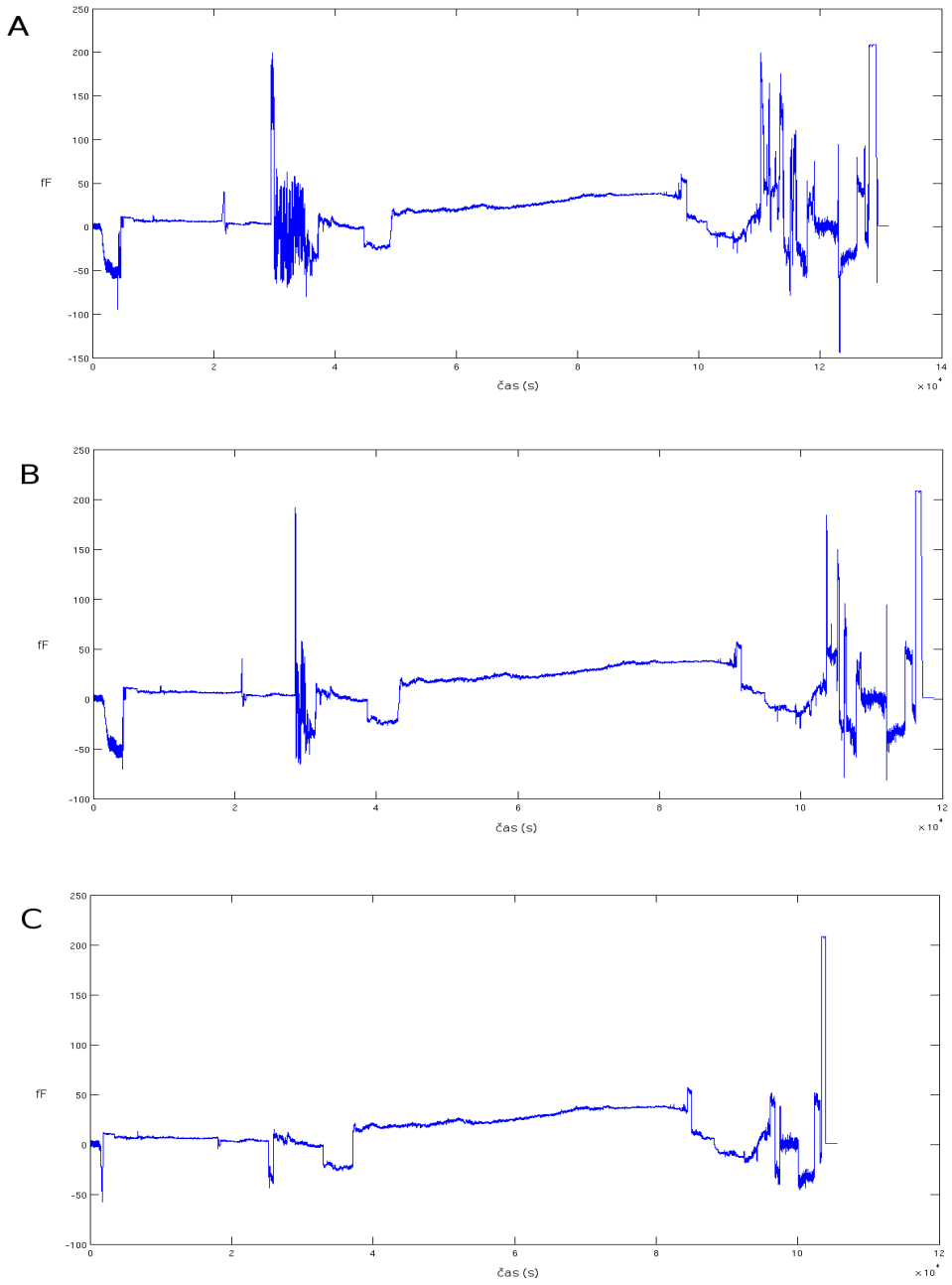
Obsežno šumni predeli posnetka se pojavijo kot posledica elektrofizioloških metod in še neraziskanih dogajanj znotraj ter zunaj celice. Spremembe, ki so znotraj takega predela, bi bile tako neveljavne, kljub temu da bi teoretično lahko ustrezale pogojem za ekstrakcijo primerov ali celo pozitivno klasifikacijo. V meritvi Cm so obsežno šumni predeli prisotni kot nenavadne skokovite spremembe velikosti reda femtofaradov, med katerimi so tudi spremembe reda atofaradov. Pogosto se znotraj teh predelov pojavijo tudi spremembe bazne linije signala. Nasploh imajo ti predeli zelo naključne lastnosti. Naključni so trenutek pojava, čas trajanja in velikosti sprememb meritve Cm znotraj predela. Slika 2.3 prikazuje različne primere obsežno šumnih delov posnetka v posameznih meritvah Cm.
Funkcija, ki bi odstranila vse take predele, mora biti invariantna na zgoraj navedene vrednosti, zaradi česar je šumne predele težko popolnoma odstraniti. Kljub temu pa je meritev možno izboljšati s preprostimi vgrajenimi Matlabovimi funkcijami, kar je implementirano v funkciji *remove_noisy_parts*. Funkcija *remove_noisy_parts* za vhodni parameter prejme meritev Cm, kot izhod pa vrne meritev Cm z manj in manjšimi šumnimi predeli. Funkcija se po potrebi lahko kliče večkrat.



Slika 2.3: Različni primeri šumnih predelov v posameznih meritvah Cm.

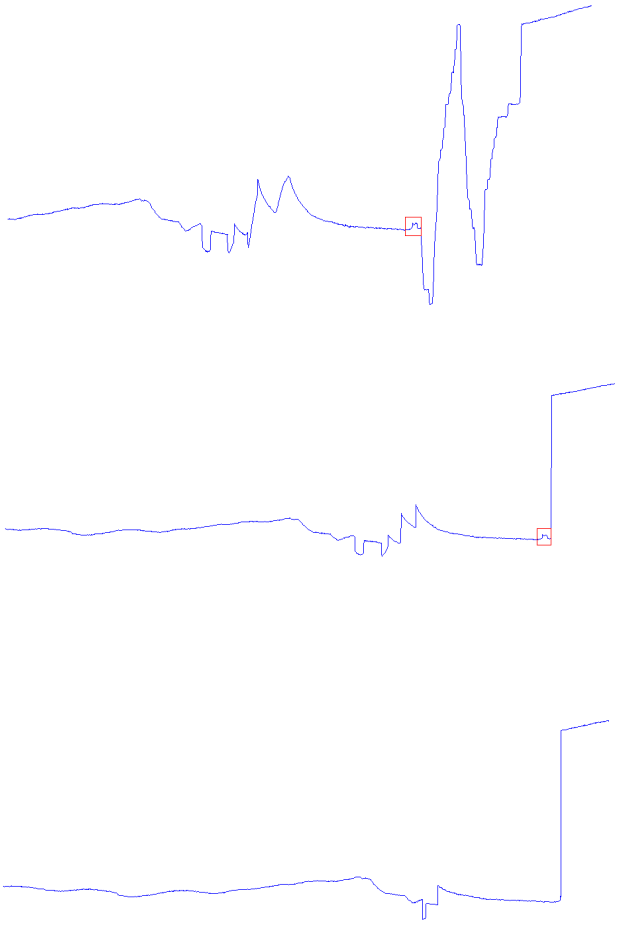
Zaradi izredno velike variacije znotraj meritve Cm je potrebna segmentacija, s pomočjo katere je možno pridobiti lokalne značilnosti podatkov. Z ustrezno nastavitvijo velikosti posameznega segmenta je možno pridobiti tiste lokalne značilnosti podatkov, ki bi razkrile lokacijo šumnih predelov. Na podlagi eksperimentalnih ugotovitev je velikost enega segmenta nastavljena na eno petstotino meritve. Za določitev tega parametra je treba upoštevati čas trajanja funkcije in kakovost končnega rezultata. Pri pregrobi segmentaciji bi bila variacija znotraj segmenta še vedno prevelika, pri predrobni segmentaciji pa bi bile značilnosti posameznega segmenta preveč lokalne, predolg pa bi bil tudi čas trajanja izvedbe funkcije *remove_noisy_parts*. Funkcija namreč deluje s pomočjo vgrajenih Matlabovih funkcij *mean* in *std*. Funkcija *mean* nad podatki izračuna povprečno vrednost, funkcija *std* pa standardno deviacijo podatkov. Zaradi velike variacije znotraj meritve pa sta ti dve vrednosti lokalno popolnoma drugačni kot pa nad celotnimi podatki. S pomočjo njunih lokalnih vrednosti je v meritvi možno poiskati vse segmente, kjer je standardna deviacija močno povečana, se pravi, kjer se podatki močno razlikujejo od svojega povprečja, kar je ena izmed glavnih in invariantnih lastnosti vseh obsežno šumnih predelov. Meja za preveliko standardno deviacijo je na podlagi eksperimentalnih ugotovitev nastavljena na dvakratno vrednost povprečja standardnih deviacij vseh segmentov. Za boljšo točnost alokacije šumnih predelov se za vse segmente, katerih standardna deviacija presega določeno mejo, poiščeta bolj natančen začetek in konec posameznega predela. Postopek je enak, kot je opisano zgoraj, le da se spremenita vrednost parametra za velikost posameznega segmenta, in sicer na eno petino, ter meja za preveliko standardno deviacijo, in sicer na povprečno vrednost standardnih deviacij vseh nadaljnjih segmentov. Vsi segmenti, ki presežejo to mejo, se izločijo iz meritve. Končni rezultat je meritev Cm z manj in manjšimi šumnimi predeli.

Z večkratno uporabo funkcije *remove_noisy_parts* se lahko šumni predeli popolnoma odstranijo. Slika 2.4 prikazuje meritev Cm po različnem številu uporabe funkcije.



Slika 2.4: Primer A prikazuje meritev Cm z vsemi obsežno šumnimi predeli, primer B prikazuje meritev po treh uporabah, primer C pa po osmih uporabah funkcije *remove_noisy_parts*.

Z večkratno uporabo se šumni predeli lahko popolnoma odstranijo, vendar pa se s tem posledično odstrani tudi manjša okolica posameznega predela, ki je veljavna meritev in lahko vsebuje pozitivno klasificirane spremembe. Slika 2.5 prikazuje primer kjer se pozitivno klasificirana sprememba odstrani skupaj s šumnim predelom.



Slika 2.5: Slika prikazuje pozitivno klasificirano spremembo, ki se ohrani pri trikratni uporabi funkcije *remove_noisy_parts*, pri šestkratni pa se odstrani skupaj s šumnim predelom.

Za meritve, ki spadajo pod učno množico, je treba upoštevati klasifikacije, ki jim pripadajo. V ta namen sem modificirala zgoraj omenjeno funkcijo v funkcijo *remove_noisy_parts_with_classifications*, ki ustrezno prilagodi klasifikacije po odstranitvi šumnih predelov.

2.3 Korekcija bazne linije signala

Bazna linija signala meritve Cm se konstantno spreminja kot posledica naraščanja in padanja Cm ter kot posledica motenj elektrofizioloških metod. Stopnja in način variacije se med meritvami ter v meritvi sami razlikujeta. Za lažjo in učinkovitejšo ekstrakcijo učnih primerov ter klasifikacijo novih skokovitih sprememb velikosti reda atofaradov je treba popraviti bazno linijo signala, in sicer tako, da bo imela vsaka meritev, oziroma signal, svojo bazno linijo enako nič. Korekcijo bazne linije lahko opravi v Matlabu vgrajena funkcija *msbackadj*. Funkcija *msbackadj* deluje s pomočjo premikajočega se okna vzdolž osi x in s pomočjo polinomske aproksimacije. Ker pa so meritve Cm različno dolge, se pojavi problem nastavitve velikosti premikajočega se okna, katerega privzeta vrednost je 200 enot in ne ustreza vsem meritvam. Poleg tega pa je v širši okolici šumnih predelov zaradi polinomske aproksimacije končna bazna linija močno popačena. Ker mora biti funkcija za korekcijo bazne linije invarianta na različne velikosti meritev Cm in na pojav naključno velikih šumnih predelov, sem se odločila za metodo korekcije bazne linije, predstavljeno v članku *Baseline Correction with Asymmetric Least Squares Smoothing* [4] , ki vsebuje implementacijo v Matlabu. Zaradi celovitosti sledi opis metode, povzet po zgoraj omenjenem članku.

Zgoraj omenjena metoda za korekcijo bazne linije signala deluje na podlagi Whittakerjeve metode za glajenje, ki je naknadno modificirana s principom asimetričnih najmanjših kvadratov. Whittakerjeva metoda za glajenje skuša signal zgladiti tako, da minimizira vrednost naslednje vsote:

$$S = \sum_i (y_i - z_i)^2 + \lambda \sum_i (z_i - 2z_{i-1} + z_{i-2})^2 \ , \tag{2.1}$$

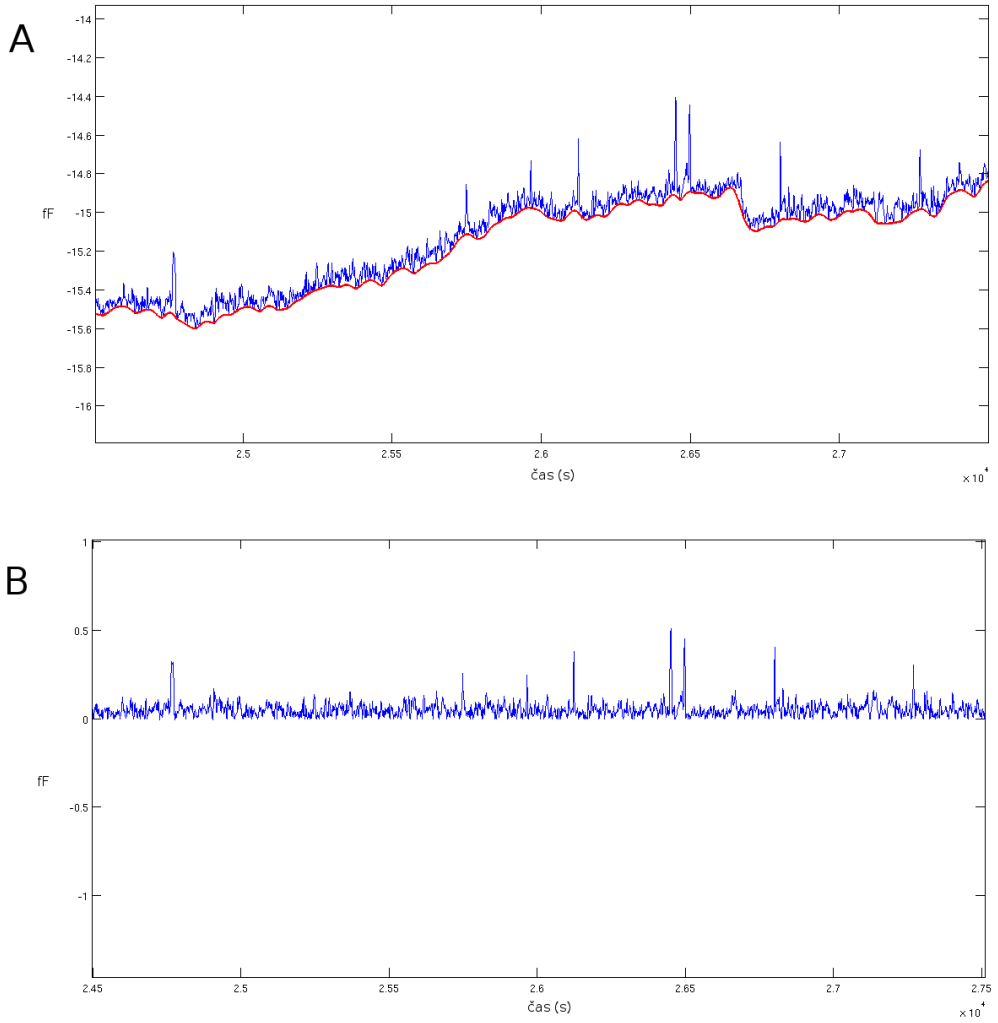
kjer je y vhodni signal, z je končni zglajeni signal in i teče od začetka do konca vhodnega signala. Prvi sumand zagotavlja čim boljšo aproksimacijo vhodnega signala, drugi sumand pa predstavlja kazen za negladkost. Parameter λ uravnovesi oba sumanda. Njegova vrednosti je odvisna od vhodnih podatkov, v praksi je $10^2 \leq \lambda \leq 10^9$. Za potrebe korekcije bazne linije meritve Cm je λ nastavljen na 10^2.
Generalizacija metode uvede vektor uteži w, kar sledi v minimizacijo naslednje vsote:

$$S = \sum_i w_i (y_i - z_i)^2 + \lambda \sum_i (z_i - 2z_{i-1} + z_{i-2})^2 \tag{2.2}$$

Minimizacija navedenega problema sledi v enačbo:

$$(W + \lambda D'D)z = Wy \ , \tag{2.3}$$

kjer je W ničelna matrika z utežmi na diagonali, D pa je matrika diferenc drugega reda.
Pri glajenju se predznak pri ostanku y - z ne upošteva oziroma negativni ostanki dobijo enako utež kot pozitivni. Če pa se negativnim ostankom pripiše več uteži, je rezultat lahko uporaben za korekcijo bazne linije signala. To je možno doseči z uvedbo novega parametra p, s katerim se uteži nastavijo na naslednji način: $y_i > z_i \implies w_i = p$, drugače $w_i = 1 - p$. Cilj je najti rešitev *z(p)* enačbe (2.3), ki bo ustrezala taki izbiri uteži. To predstavlja princip asimetričnih najmanjših kvadratov, apliciran na glajenje z Whittakerjevo metodo. Vrednost parametra p je odvisna od vhodnih podatkov, v praksi je $0.001 \leq p \leq 0.1$. Za potrebe korekcije bazne linije meritve Cm je p nastavljen na 0.001.
Rezultat ni takoj očiten, potrebna je iteracija, kjer se sprva z uniformnimi utežmi izračuna z, v vsaki naslednji iteraciji pa se na podlagi izračunanega z popravijo uteži in novi z, dokler se vrednost uteži ne spreminja več. V praksi je dovolj že od 5 do 10 iteracij. Slika 2.6 prikazuje del meritve Cm pred korekcijo bazne linije in po njej.



Slika 2.6: Na sliki A je del meritve Cm z označeno bazno linijo pridobljeno s funkcijo *baseline_correction*, na sliki B pa je meritev Cm po korekciji bazne linije.

Za korekcijo bazne linije je potrebno z odšteti od meritve. Rezultat je meritev oziroma signal z bazno linijo, poravnano na nič. Metoda je implementirana v funkciji *baseline_correction*, ki za vhod prejme meritev Cm, parameter p in paramter λ, za izhod pa vrne meritev Cm s popravljeno bazno linijo.

Poglavje 3

Priprava množice podatkov

Osnova strojnega učenja je množica podatkov. Na podlagi njenih lastnosti lahko učni algoritem zgradi model oziroma pravilo za klasifikacijo, ki jo opravi izvajalni algoritem. Množica podatkov oziroma matrika je zgrajena iz primerov oziroma vrstic ter njihovih atributov ali značilk oziroma stolpcev. Pri klasifikaciji zadnji atribut predstavlja razred, ki mu primer pripada, razred, ki ga mora pri novem primeru določiti izvajalni algoritem. Primere je treba določiti iz meritev Cm. Meritve Cm so le zaporedja diskretnih vrednosti, ki jih je mnogo preveč, da bi vsaka zase predstavljala primer, prav tako pa posamezna vrednost sama zase nima ustreznih lastnosti, na podlagi katerih bi lahko učni algoritem zgradil smiselno pravilo za klasifikacijo skokovitih sprememb. Iz diskretnih vrednosti, ki predstavljajo meritev Cm, je treba pridobiti primere tako, da jim bo mogoče pripisati značilnosti, katerih množice vrednosti bi se med pozitivno in negativno klasificiranih primerih čim bolj razlikovale.

3.1 Ekstrakcija primerov

Glede na to, da je treba klasificirati skokovite spremembe v meritvi Cm, jih je smiselno določiti kot primere v množici podatkov. Ker pa so meritve Cm zelo šumne, v poštev ne pridejo vse skokovite spremembe, temveč le tiste,

katerih velikost je reda velikosti pozitivno klasificiranih sprememb. Ekstrakcija takih skokovitih sprememb se lahko implementira s pomočjo preteklih analiz za alokacijo ustreznih skokovitih sprememb v meritvah, kjer je razmerje med signalom in šumom dovolj veliko[1]. Da pa bi bila ekstrakcija še bolj temeljita, se skokovite spremembe poiščejo še s pomočjo korekcije bazne linije signala.

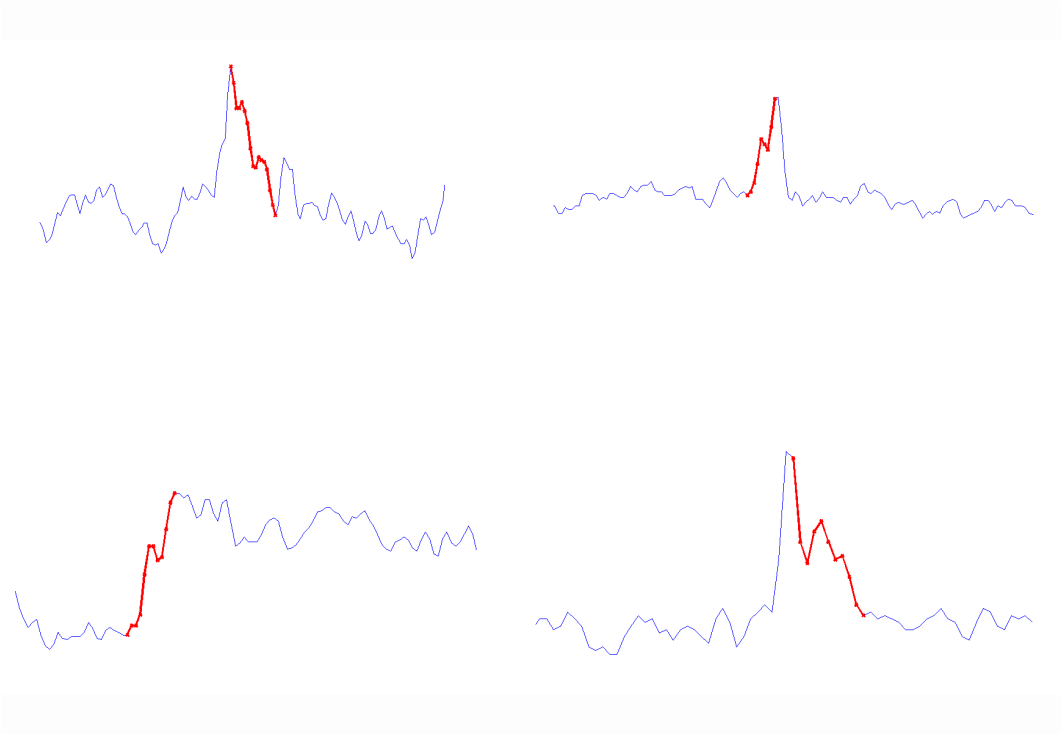
Ekstrakcija primerov je implementirana v funkciji *extract_instances*, ki za vhod prejme meritev Cm in dva parametera za meje velikosti skokovitih sprememb, za izhod pa vrne 2xn matriko, v kateri so shranjene lokacije x vznožij in lokacije x vrhov skokovitih sprememb. Na začetku funkcije *extract_instances* se izračunata vektor diferenc meritve Cm in vektor, ki predstavlja meritev Cm s popravljeno bazno linijo. Prvi vektor poseduje informacijo o velikosti razlik med zaporednimi diskretnimi vrednostmi v meritvi Cm in tako razkriva potencialne lokacije skokovitih sprememb. Drugi vektor predstavlja meritev Cm, kjer so vse skokovite spremembe poravnane na bazno linijo. Bazna linija pa je popravljena na nič, kar omogoča iskanje vrhov ustrezno velikih skokovitih sprememb, ki so posledice prehodnih eksocitoz.

S pomočjo obeh vektorjev in Matlabove vgrajene funkcije *findpeaks* se na podlagi lokalnih značilnosti posameznega segmenta meritve Cm pridobijo primerne skokovite spremembe reda velikosti pozitivno klasificiranih skokovitih sprememb. Na podlagi eksperimentalnih ugotovitev je velikost posameznega segmenta nastavljena na eno tisočino.

V zanki *for* se za vsak posamezni segment izračuna povprečna vrednost z Matlabovo funkcijo *mean*. Povprečje se izračuna za vektor diferenc meritve Cm, negativni vektor diferenc in za meritev Cm s popravljeno bazno linijo. Na podlagi teh vrednosti se za vsak vektor pridobijo vrhovi s pomočjo Matlabove vgrajene funkcije *findpeaks*, in sicer vsi vrhovi, večji od povprečne vrednosti segmenta posameznega vektorja. Zaradi vseprisotnega šuma so nekateri vrhovi neveljavni. Vrhove, ki so preveliki ali preveč oddaljeni od povprečja, odstrani funkcija *cut_peaks_avg*.

Funkcija na podlagi povprečja 100 sosedov lokacije vrha odstrani tiste vrhove,

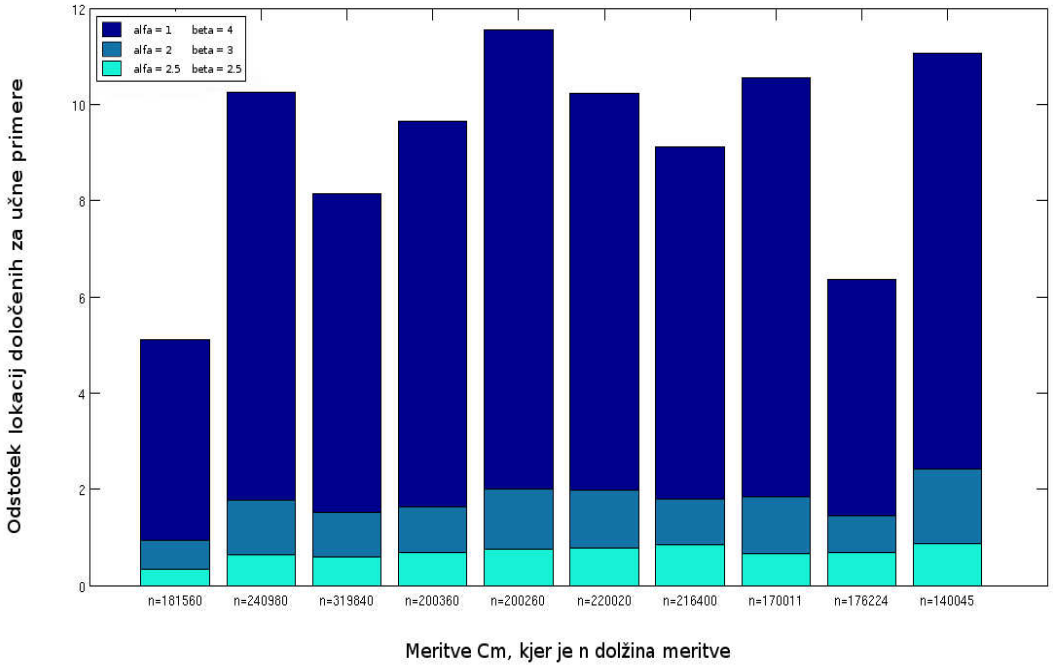
ki so večji od 5 femtofaradov ali so od povprečja oddaljeni za 5 femtofaradov. Vse tri vrednosti je mogoče funkciji podati prek parametrov. Vrednosti parametrov so nastavljene na podlagi eksperimentalnih ugotovitev z dobro mero tolerance, zaradi česar je njihovo vrednost možno spremeniti le znotraj funkcije *extract_instances*. Vseprisotni šum vpliva tudi na obliko skokovitih sprememb oziroma na njihovo izrazitost. Slika 3.1 prikazuje nekaj takih primerov. Pri preprosti analizi diskretnih podatkov so informacije o velikosti takih sprememb lahko zavajajoče, zaradi česar je potrebna dodatna podrobnejša analiza.



Slika 3.1: Slika predstavlja primere skokovitih sprememb, popačenih zaradi šuma.

S pomočjo funkcije, ki poišče lokalni minimum ob strani vrha, se izračunajo

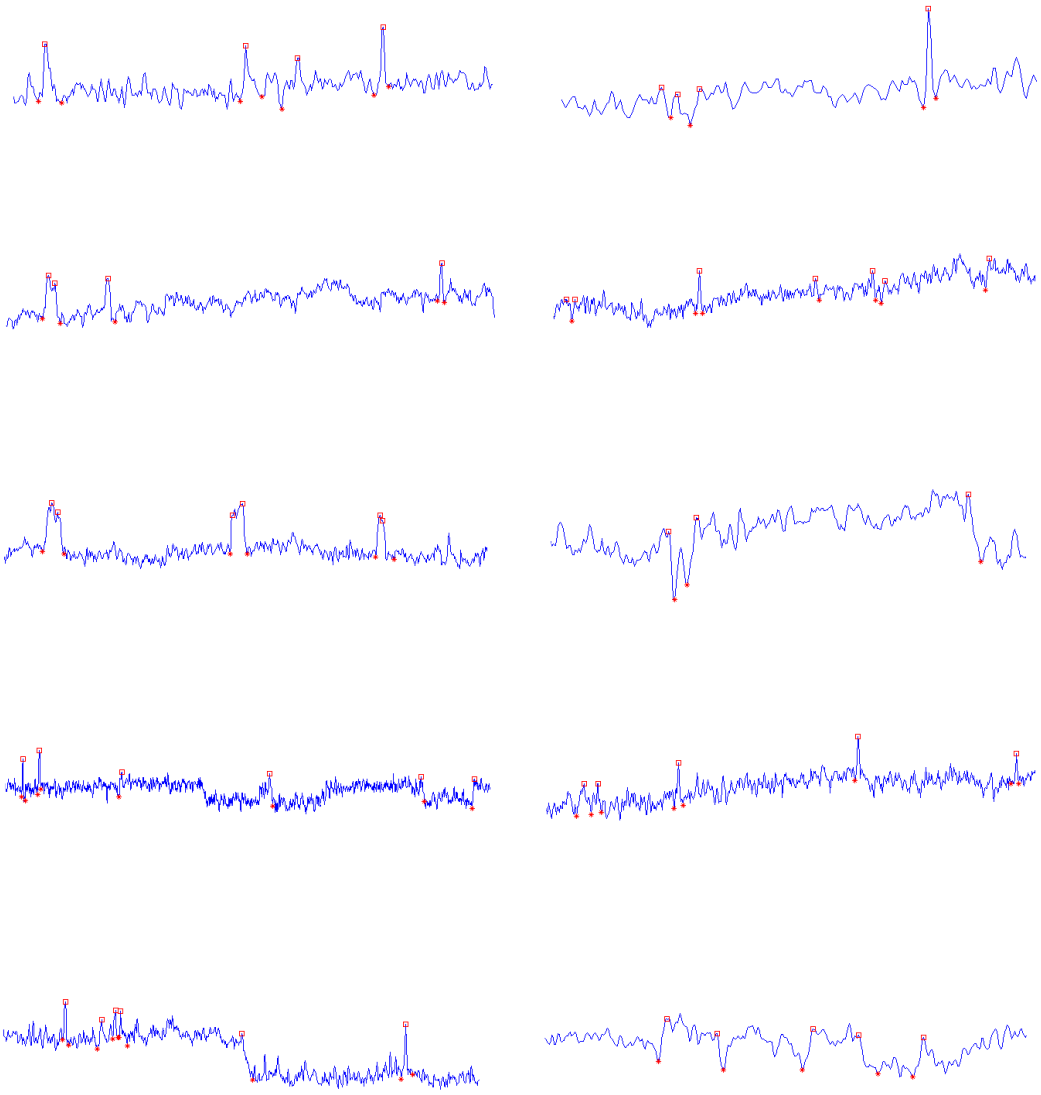
dejanske velikosti skokovitih sprememb. Na podlagi teh velikosti se izločijo neprimerno velike spremembe. Lokalni minimumi se poiščejo v funkciji *find_loc_min*, ki poišče lokalne minimume z neko mero tolerance na šum vzdolž strani do vznožja vrha. Funkcija *find_loc_min* za vhod prejme meritev Cm, lokacije vrhov, vrednosti na lokacijah vrhov in parameter za smer iskanja lokalnega minimuma. Na podlagi parametra za smer iskanja se lokalni minimum lahko poišče v intervalu 100 sosedov levo ali 100 sosedov desno od vrha. Velikost intervala sosedov je ohlapno nastavljena, lokalni minimum se skoraj vedno pojavi mnogo prej. Funkcija *find_loc_min* namreč poišče lokalni minimum tako, da med zaporednimi pozitivnimi spremembami poišče lokacijo, kjer se ali zgodijo 3 zaporedne pozitivne spremembe ali je vsota vseh pozitivnih sprememb večja od četrtine vrednosti v vrhu ali pa je razdalja do naslednje pozitivne spremembe premajhna oziroma dvakrat manjša od trenutne pozitivne spremembe. S pomočjo lokalnih minimumov se izračunajo dejanske velikosti skokovitih sprememb, in sicer kot razdalje od lokalnega minimuma do vrha spremembe. Na podlagi teh velikosti in vhodnih parametrov za meje velikosti skokovitih sprememb se izločijo neprimerno velike spremembe Zadnja dva vhodna parametra močno vplivata na končno število primerov. Histogram na sliki 3.2 predstavlja različno število primerov pri različnih vrednostih parametrov alfa in beta. Prvi parameter alfa določa minimalno velikost primerno velike skokovite spremembe, in sicer tako, da se pomnoži z vrednostjo, ki predstavlja povprečje posameznega segmenta. Drugi parameter beta sam zase določa maksimalno velikost primerno velike spremembe. S pomočjo zgornjih parametrov je možno bolje uskladiti število ustreznih skokovitih sprememb s številom skokovitih sprememb, ki so posledica šuma. V vsakem primeru je drugih mnogo več, lahko pa se njihova obsežnost drastično zmanjša na račun nekaj izgubljenih ustreznih skokovitih sprememb.
Celoten postopek pridobivanja primerno velikih skokovitih sprememb se ponovi za vektor diferenc meritve Cm in za negativni vektor diferenc.



Slika 3.2: Histogram prikazuje odstotek lokacij, določenih za primere množice podatkov, pri treh različnih parih vrednosti parametrov za meje velikosti skokovitih sprememb za deset različnih meritev, kjer n predstavlja dolžino meritve.

Odstranijo se vrhovi, oziroma diference, ki so večji od 2 femtofaradov ali so njihove lokacije od povprečja oddaljene za 2 femtofarada. Dejanska velikost skokovite spremembe se izračuna tako, da se za posamezno diferenco izračunata lokalni minimum in lokalni maksimum s funkcijo *find_loc_min*. Neprimerne dejanske velikosti sprememb se izločijo tako, kot je opisano v prejšnjem odstavku. Na koncu se iz vektorja vseh pridobljenih primerno velikih skokovitih sprememb, oziroma njihovih lokacij, odstranijo tiste lokacije, ki so preblizu skupaj. V takem primeru gre namreč za dve lokaciji iste skokovite spremembe, ki sta bili pridobljeni iz različnih vektorjev. Končni rezultat je 2xn matrika, ki v prvi vrstici vsebuje vznožja, v drugi vrstici pa vrhove skokovitih sprememb. Te skokovite spremembe predstavljajo primere v množici podatkov.

Slika 3.3 na različnih meritvah Cm prikazuje primerno velike skokovite spremembe, ki predstavljajo primere v množici podatkov.
Da bo množica podatkov celovita, je posameznim primerom treba pripisati njihove značilnosti. Opis postopka sledi v naslednjem poglavju.



Slika 3.3: Različne meritve Cm, kjer so lokacije vznožij primerno velikih skokovitih sprememb označene z zvezdico, vrhovi pa s kvadratkom.

3.2 Atributi

Atribut je spremenljivka, ki za posamezni primer zavzema potencialno različne vrednosti, s katerimi je primer opisan. Množica atributov določa prostor, v katerem so primeri, vrednosti atributov za dani primer pa določajo točko v tem prostoru. Učni algoritem v tem prostoru zgradi model za klasifikacijo. Pri nekaterih metodah, kot je metoda k-najbližjih sosedov, pa prostor atributov oziroma njihove vrednosti že določajo model za klasifikacijo. Domena atributa je določena z vrsto atributa, ki označuje vrednosti, ki jih lahko atribut sprejme. V glavnem se atributi delijo med diskretne in zvezne atribute. Rezultat strojnega učenja oziroma iskana spremenljivka se lahko obravnava kot eden izmed atributov, katerega vrednost je treba ugotoviti pri novem primeru. V osnovi je pri klasifikacijskem problemu iskana vrednost diskretna, in sicer predstavlja razred, v katerega je klasificiran nov primer, pogosto pa vrednost razred nadomestimo z verjetnostno distribucijo po vseh razredih. V množici skokovitih sprememb so vsi atributi zvezni. Iskana spremenljivka je zaradi preprostosti binarna, in sicer vrednost nič označuje negativno klasificiran primer, kar pomeni, da je skokovita sprememba posledica šuma, vrednost ena pa označuje pozitivno klasificiran primer, kar pomeni, da je skokovita sprememba posledica raziskovanega dogodka.
Izračun atributov je implementiran v funkciji *generate_attributes*, ki za vhod prejme meritev Cm ter 2xn matriko lokacij skokovitih sprememb, za izhod pa vrne nxm matriko, kjer n predstavlja število primerov in m predstavlja število atributov.

1. Prvi atribut predstavlja velikosti skokovitih sprememb, ki so normirane po formuli:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \ , \tag{3.1}$$

   kjer je x originalna vrednost, x' pa normirana vrednost.

2. Drugi atribut je naklon od vznožja do vrha spremembe. Naklon se izračuna s pomočjo Matlabove vgrajene funkcije *polyfit*, ki poišče koeficiente polinoma *P(x)* stopnje *N*, ki se vhodnim podatkom najbolje prilega po metodi najmanjših kvadratov. Za informacijo o naklonu je treba izračunati polinom prve stopnje oziroma linearni polinom, katerega naklon je koeficient na prvem mestu, oziroma c_1:

$$P(x) = c_1 x + c_2 \ . \tag{3.2}$$

3. V sklopu opisa tipičnih lastnosti skokovitih sprememb, ki so posledice prehodnih eksocitoz, tretji atribut predstavlja čas, ki je minil od zadnjega primera oziroma skokovite spremembe.

4. V sklopu opisa tipičnih lastnosti skokovitih sprememb, ki so posledice prehodnih eksocitoz, četrti atribut predstavlja razmerje med velikostjo trenutnega in velikostjo prejšnjega primera oziroma skokovite spremembe.

5. V sklopu opisa tipičnih lastnosti skokovitih sprememb, ki so posledice prehodnih eksocitoz, peti atribut predstavlja čas, ki je minil do naslednjega primera.

6. V sklopu opisa tipičnih lastnosti skokovitih sprememb, ki so posledice prehodnih eksocitoz, šesti atribut predstavlja razmerje med velikostjo trenutnega in velikostjo naslednjega primera oziroma skokovite spremembe.

7. V sklopu opisa lastnosti bližnje okolice skokovite spremembe sedmi atribut predstavlja razmerje med velikostjo skokovite spremembe in povprečjem stotih sosedov ob vznožju skokovite spremembe.

8. V sklopu opisa lastnosti bližnje okolice skokovite spremembe osmi atribut predstavlja razmerje med velikostjo skokovite spremembe in povprečjem dvajsetih sosedov ob vrhu skokovite spremembe.

9. V sklopu opisa lastnosti bližnje okolice skokovite spremembe deveti atribut predstavlja standardno deviacijo stotih sosedov ob vznožju skokovite spremembe.

10. V sklopu opisa lastnosti bližnje okolice skokovite spremembe deseti atribut predstavlja standardno deviacijo dvajsetih sosedov ob vrhu skokovite spremembe.

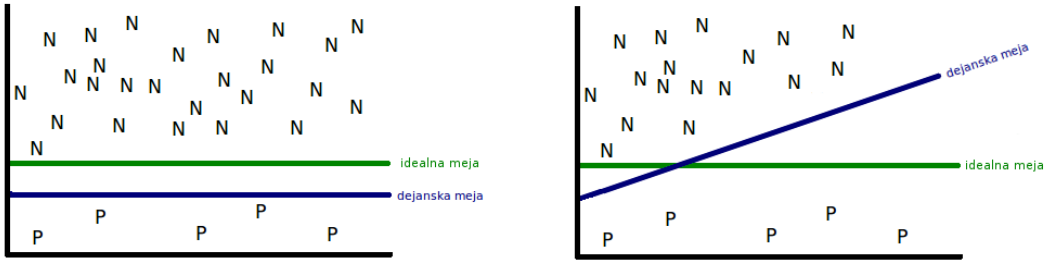
Pri gradnji učne množice, kjer so klasifikacije znane, zadnji atribut predstavlja vektor ničel in enk, ki označuje negativno, oziroma pozitivno klasificirane primere. Vektor klasifikacij je treba pridobiti na podlagi ročno pregledanih meritev Cm. Pri ročnem pregledovanju se lokacije ustrezno velikih skokovitih sprememb shranijo v tekstovno datoteko. Vektor klasifikacij se generira v funkciji *get_classifications*, ki za vhod prejme dolžino meritve Cm ter tekstovno datoteko z lokacijami ustreznih skokovitih sprememb, za izhod pa vrne vektor ničel in enk, ki predstavlja negativno oziroma pozitivno klasificirane lokacije za dano meritev Cm. Po ekstrakciji primerov množice podatkov, oziroma primerno velikih skokovitih sprememb, je potrebno med njimi poiskati tiste, ki so bile pri ročno pregledanih meritvah Cm označene pozitivno, oziroma označene kot posledice raziskovanega dogodka. Namen je ustrezno modificirati vektor klasifikacij, namreč lokacije, ki so bile pridobljene pri avtomatski ekstrakciji skokovitih sprememb, so skoraj vedno za nekaj enot oddaljene od ročno označenih lokacij. Za to poskrbi funkcija *match_classification*, ki za vhod prejme lokacije vznožij in vrhov skokovitih sprememb, ki predstavljajo primere v množici podatkov, ter lokacije vznožij in vrhov skokovitih sprememb, ki so bile ročno označene kot pozitivne. Za izhod funkcija vrne popravljen vektor ničel in enk, ki klasificira vse skokovite spremembe med pozitivne spremembe oziroma spremembe, ki so posledica raziskovanih dogodkov, ter negativne spremembe oziroma spremembe, ki so posledica šuma. S tem je množica podatkov celovita. Za namen ocenjevanja rezultatov strojnega učenja jo je treba razdeliti na učno in testno množico. Da bo učna množica ustrezna za modeliranje, jo je treba prevzorčiti, saj je negativni razred prekomerno zastopan kljub ekstrakciji le tistih skokovitih sprememb,

katerih velikost je reda velikosti ustreznih skokovitih sprememb. Cilj je pripraviti množico, ki bo čim bolj uravnotežena, ne da bi se s tem izkrivila njena slika meritve membranske kapacitivnosti.

3.3 Prevzorčenje učne množice

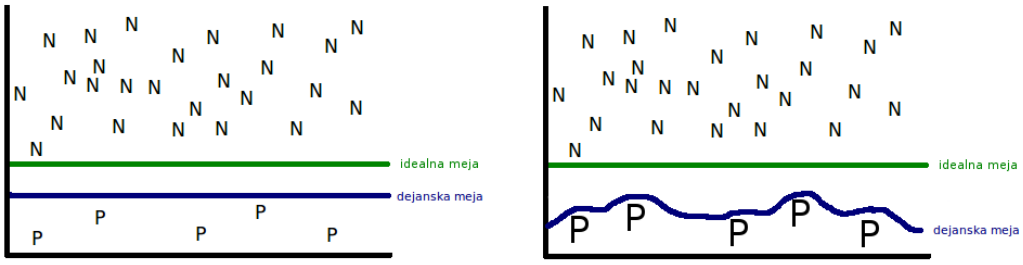
Pri ekstrakciji primerov množice podatkov za modeliranje sta bila parametra alfa in beta nastavljena na 2 oziroma 3. Pri taki nastavitvi ima končna množica 78467, oziroma 98,02 %, negativno klasificiranih primerov ter 1583, oziroma 1,98 %, pozitivno klasificiranih primerov. Model take učne množice bi bil neuporaben za klasifikacijo novih primerov. Zaradi tako velike verjetnosti večinskega razreda bi bil manjšinski razred ignoriran. Rezultat bi sicer sprva bil videti uspešen, namreč v primerjavi z relativno majhnim številom lažnih negativnih bi rezultat vseboval veliko število pravilnih negativnih in malo ali nič lažnih pozitivnih. Ker pa bi rezultat vseboval zelo majhno število pozitivno klasificiranih primerov, bi dejansko tak izid pomenil neuspeh, saj problem detekcije ustreznih skokovitih sprememb ne bi bil rešen. Da bo učni algoritem zgradil ustrezen model, je treba učno množico prevzorčiti tako, da bo odstotek negativno klasificiranih primerov primerljiv z odstotkom pozitivno klasificiranih primerov. Glede na to, da je pri učni množici znano, kateri primeri pripadajo večinskemu in kateri primeri pripadajo manjšinskemu razredu, je mogoče odstraniti delež primerov večinskega razreda ter razmnožiti primere manjšinskega razreda. Vendar bi se s poseganjem v množico podatkov potencialno lahko odstranila pomembna informacija, brez katere bi podatki prikazovali drugačno situacijo. Način izbire primerov za odstranitev iz večinskega razreda tako lahko močno vpliva na končni rezultat strojnega učenja. Če bi se pri podvzorčenju odstranili robni primeri, kot to prikazuje slika 3.4, bi bili podobni novi negativni primeri napačno klasificirani kot pozitivni. Vendar če že učna množica vsebuje napačno klasificirane primere, bi bili odstranjeni ravno tisti primeri, ki bi morali biti v resnici klasificirani pozitivno, in tako bi lahko klasifikator še natančneje opravil svoje delo. Na

podlagi ocen strojnega učenja in ročnega pregleda rezultatov je treba ugotoviti najbolj primerno izbiro primerov za odstranitev iz večinskega razreda.



Slika 3.4: Primer prikazuje, kako se pri neuravnoteženi učni množici po podvzorčenju večinskega razreda lahko napačno spremeni meja med manjšinskim in večinskim razredom.

Prav tako je pomembno, na kakšen način se razmnožijo primeri manjšinskega razreda. Preprosto podvajanje bi lahko privedlo do prevelikega prileganja manjšinskemu razredu, kar prikazuje slika 3.5.



Slika 3.5: Primer prikazuje, kako se pri neuravnoteženi učni množici po podvajanju primerov manjšinskega razreda lahko napačno spremeni meja med manjšinskim in večinskim razredom.

Sledi opis algoritmov za podvzorčenje in nadvzorčenje, ki z upoštevanjem nekaterih lastnosti množice skušajo čim bolje prevzorčiti množico podatkov.

Podvzorčenje večinskega razreda

Najbolj preprost način podvzorčenja je naključna izbira deleža večinskega razreda, ki lahko privede do popolnoma zadovoljivih rezultatov. Kljub temu pa je vredno poskusiti rezultat izboljšati s pomočjo algoritma, ki delež za odstranitev izbere na podlagi določenih lastnosti učne množice. Učna množica vsebuje veliko različnih primerov, razporejenih v 11-dimenzionalnem prostoru. Da se razporeditev množice ne bi preveč spremenila, se lahko učna množica v prostoru razdeli na skupine med seboj podobnih primerov in se izbira deleža za odstranitev opravi znotraj vsake skupine. Na podlagi tega principa deluje podvzorčenje na podlagi gručenja *k-means* [10]. Metoda gručenja *k-means* iterativno razdeli podatke v k centroidov tako, da sprva izbere k naključnih primerov in jih določi za centroide, nato za vsak primer izračuna razdaljo do vsakega centroida in ga pripiše najbližjemu, nato pa v zadnjem koraku za vsako gručo izračuna nov centroid na podlagi povprečja vseh primerov znotraj gruče. Postopek gručenja se ponavlja, dokler se spreminja razporeditev primerov v gruče ali ko je doseženo maksimalno število iteracij. Podvzorčenje na podlagi gručenja *k-means* je implementirano v funkciji *kmeans_undersample*, ki za vhod dobi učno množico, parameter k, ki določa število gruč, in parameter, ki določa eno izmed štirih možnosti za izbiro primerov za odstranitev. Na začetku se z vgrajeno Matlabovo funkcijo *kmeans* opravi gručenje *k-means*. Zatem se za vsako gručo, ki je večja od 50 primerov, gruča razdeli na večinski in manjšinski razred ter nastavi velikost deleža za odstranitev, in sicer na polovico razlike med velikostjo večinskega ter manjšinskega razreda. Sledi izračun povprečja evklidskih razdalj od vsakega primera večinskega razreda do vseh primerov manjšinskega razreda. Na podlagi teh povprečij so na voljo štiri možnosti za izbiro primerov za odstranitev. Pri prvi možnosti se izberejo tisti, ki so najbližje manjšinskim primerom, pri drugi možnosti se izberejo tisti, ki so najdlje od manjšinskih primerov, pri tretji možnosti se izberejo tisti katerih razdalje so v večinskem razredu razdalj, pri zadnji možnost pa se primeri za odstranitev izberejo naključno. Rezultat je učna množica, ki vsebuje približno pol manj negativno

klasificiranih primerov. Pri kateri izmed štirih možnosti je bila učna množica najbolje podvzorčena, bodo pokazale ocene strojnega učenja.

Nadvzorčenje manjšinskega razreda

Pri nadvzorčenju s preprostim podvajanjem lahko pride do prevelikega prileganja učni množici. Učni algoritem, ki bi večkrat videl popolnoma enake primere, bi posledično zgradil preveč točen model, ki bi ustrezal le točno določenim primerom. Kljub temu je treba povečati manjšinski razred in to prav s primeri manjšinskega razreda. Preveliko prileganje se lahko odpravi s tem, da se namesto identičnih podvojenih primerov generirajo sintetični primeri manjšinskega razreda. Metoda, kjer se sintetični primeri generirajo na podlagi najbližjih sosedov vsakega primera manjšinskega razreda, se imenuje SMOTE (angleško *Synthetic Minority Over-sampling Technique*) in je podrobneje opisana v članku [9]. Metoda za vsak primer manjšinskega razreda sprva izračuna k-najbližjih sosedov. Število k je odvisno od želene velikosti končnega nadvzorčenega manjšinskega razreda. Nato za vsakega soseda generira nov primer tako, da v atributnem prostoru izračuna razliko med sosedom in trenutnim primerom manjšinskega razreda ter to razliko, pomnoženo z naključnim številom, prišteje k trenutnemu primeru manjšinskega razreda. Končna nadvzorčena množica je večja za N x k primerov, kjer je N število začetnih primerov manjšinskega razreda. Nadvzorčenje po metodi SMOTE je implementirano v Matlabovi funkciji *SMOTE_oversample*, ki za vhod prejme učno množico in parameter k, za izhod pa vrne učno množico z nadvzorčenim manjšinskim razredom. Pri nadvzorčenju manjšinskega razreda originalne učne množice je bil parameter k nastavljen na 20, pri nadvzorčenju manjšinskega razreda učne množice s podvzorčenim večinskim razredom pa je bil parameter k nastavljen na 10.

Poglavje 4

Strojno učenje na signalu membranske kapacitivnosti

V strojnem učenju se prepletajo računalništvo, informatika in statistika. V računalništvu in informatiki strojno učenje spada v podpodročje umetne inteligence, ki preučuje računske procese, ki omogočajo zaznavanje, sklepanje in ukrepanje [11]. Strojno učenje temelji na vhodnih podatkih in algoritmih, ki iz njih povzamejo znanje oziroma pravila. To znanje je predstavljeno v obliki opisa ali modela podatkov. Cilj strojnega učenja je lažje in natančnejše procesiranje novih podatkov s pomočjo znanja, pridobljenega iz vhodnih podatkov. Glede na potrebe procesiranja podatkov se strojno učenje deli na tri pristope:

1. **Nadzorovano učenje**, kjer je pri vhodnih podatkih točno določeno, kaj se mora algoritem naučiti in kaj je iskani rezultat. Glavna naloga nadzorovanega učenja je generirati splošno pravilo, ki preslika vhode na izhode.

2. **Nenadzarovano učenje**, ki je, kot kaže ime, prepuščeno sebi, da preuči strukturo vhodnih podatkov in ugotovi, ali podatki vsebujejo skrita pravila oziroma vzorce.

3. **Spodbujevano učenje**, ki temelji na povratni informaciji, ki lahko

predstavlja nagrado ali kazen. Nagrada ali kazen sta spodbuda za učenje in gradnjo čim boljše strategije za dosego cilja.

Najbolj pogosto uporabljeno je nadzorovano učenje, in sicer podpodročje nadzorovanega učenja, ki se ukvarja s klasifikacijo ali uvrščanjem.
Na podlagi množice primerov, ki jim je že pripisan razred, mora klasifikator določiti razred novemu primeru. Glede na to, da je cilj diplomskega dela (pol)avtomatska detekcija ustreznih skokovitih sprememb v signalu membranske kapacitivnosti, je klasifikacija primerno področje strojnega učenja. Na podlagi vhodne množice skokovitih sprememb, ki imajo že določen razred, mora klasifikator določiti razred novi skokoviti spremembi. Možna sta le dva razreda, in sicer negativni razred, oziroma skokovita sprememba je posledica šuma, in pozitivni razred, oziroma skokovita sprememba je posledica raziskovanega dogodka. Klasifikator za določitev razreda potrebuje funkcijo, ki preslika prostor atributov v razred. To nalogo opravi učni algoritem, ki na podlagi vhodne množice klasificiranih primerov izpelje klasifikacijsko funkcijo ali pravilo. Obstaja mnogo različnih metod klasifikacije, med seboj se ločijo glede na način predstavitve klasifikacijske funkcije. Sledi kratek opis nekaterih najbolj preprostih metod, izbranih za klasifikacijo skokovitih sprememb v signalu membranske kapacitivnosti.

4.1 Opis izbranih metod strojnega učenja

Naslednje izbrane metode so najbolj osnovne in preproste metode strojnega učenja na področju klasifikacije, ki kljub svoji preprostosti lahko dosežejo popolnoma zadovoljive rezultate.

Odločitveno drevo

Odločitvena drevesa spadajo pod simbolično strojno učenje. Listi drevesa predstavljajo razrede, notranja vozlišča predstavljajo atribute in veje predstavljajo podmnožice vrednosti atributov. Gradnja odločitvenega drevesa poteka z razdelitvijo vhodne množice primerov glede na vrednosti atributov.

Postopek razdelitve se rekurzivno nadaljuje na vsaki nadaljnji podmnožici, dokler ni izpolnjen ustavitveni pogoj. Pri izpolnjenem ustavitvenem pogoju se gradnja ustavi, ustvari se list, pod katerega spada trenutna podmnožica primerov. Ustavitveni pogoj je lahko dejstvo, da vsi primeri ali večina primerov v podmnožici pripadajo istemu razredu ali pa podmnožica vsebuje tako malo primerov, da bi bila nadaljnja gradnja nesmiselna. Lahko se pa tudi zgodi, da za nadaljnjo delitev ne obstaja ustrezen atribut, namreč algoritem pri delitvi množice primerov skuša poiskati atribut, ki bi množico primerov razdelil na čim bolj "čiste"podmnožice. Z zgrajenim odločitvenim drevesom se lahko klasificira nov primer. Odločitveno pravilo za klasifikacijo novega primera je pot od korena drevesa do lista.

Naivni Bayesov klasifikator

Naivni Bayesov klasifikator je verjetnostni klasifikator, saj pri klasifikaciji novega primera ne napove le določenega razreda, ampak verjetnostno porazdelitev po vseh možnih razredih. Posteriorna verjetnostna porazdelitev po razredih se napove s pomočjo Bayesovega pravila:

$$P(r_i|A) = P(r_i) * \frac{P(A|r_i)}{P(A)} \tag{4.1}$$

i = 1... število razredov
A = množica danih atributov
P(r_i|A)... posteriorna verjetnost razreda r_i pri danih atributih
P(r_i)... apriorna verjetnost razreda r_i
P(A|r_i)... pogojna verjetnost razreda r_i pri danih atributih
P(A)... apriorna verjetnost primera pri danih atributih.

Ker pa naivni Bayesov klasifikator predpostavlja neodvisnost vrednosti atributov, se po formuli za pogojno verjetnost neodvisnih dogodkov in izpustitvi

neodvisnih spremenljivk formula poenostavi na:

$$P(r_i|A) = P(r_i) * \prod_{k=1}^{|A|} \frac{P(r_i|a_k)}{P(r_i)} \tag{4.2}$$

$\mathrm{a_k} \in A, k = 1 \ldots |\mathrm{A}|$

Ker pa točne verjetnosti na desni strani enačbe niso znane, mora učni algoritem naivnega Bayesa te verjetnosti aproksimirati na podlagi učne množice. Za ocenjevanje apriornih verjetnosti razredov se uporablja Laplaceov zakon zaporednosti [12]. Za ocenjevanje pogojnih verjetnosti razreda pri dani vrednosti atributa pa se uporablja m-ocena [5]. Ti dve aproksimaciji predstavljata znanje, ki se ga učni algoritem naivnega Bayesa nauči iz učne množice.

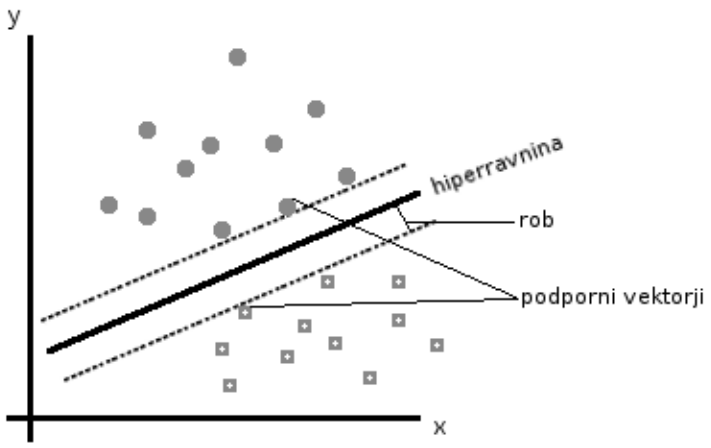
K-najbližjih sosedov

Metoda k-najbližjih sosedov spada pod numerične metode strojnega učenja, zaradi načina učenja pa spada tudi pod *leno učenje*. Učni algoritem si pri metodi k-najbližjih sosedov namreč samo zapomni celotno množico ali podmnožico učnih primerov. Zahtevno delo opravi izvajalni algoritem, ki mora nov primer klasificirati. Izvajalni algoritem nov primer klasificira na podlagi večinskega razreda množice k-najbližjih sosedov v atributnem prostoru, ki si ga je zapomnil učni algoritem. Pri računanju razdalj se uporabljajo različne metrike, najbolj pogosta je evklidska razdalja.

Metoda podpornih vektorjev

Metoda podpornih vektorjev ali SVM (angleško *support vector machine*) prav tako spada pod numerične metode strojnega učenja in je v osnovni obliki namenjena za binarno klasifikacijo oziroma regresijo. Temeljni cilj metode podpornih vektorjev je postavitev optimalne hiperravnine v prostoru atributov, kot kaže slika 4.1. Hiperravnina naj bi bila optimalna v smislu, da med seboj loči primere iz različnih razredov tako, da se najbolj in enako oddaljuje

od robnih primerov obeh razredov. Robni primeri se imenujejo *podporni vektorji*, njihova razdalja do hiperravnine pa *rob*. Optimiziranje hiperravnine je maksimizacija roba. V praksi se redko zgodi, da so primeri v atributnem prostoru linearno ločljivi, posledično ima klasifikacija z linearno hiperravnino slabe rezultate.



Slika 4.1: Prikaz osnovnega principa metode podpornih vektorjev.

V tem primeru metoda podpornih vektorjev implicitno transformira originalni atributni prostor v kompleksnejši atributni prostor, ki je primernejši za postavitev linearne hiperravnine. Transformacija je mogoča z različnimi jedrnimi funkcijami. Osnovna jedrna funkcija je linearna, ki ohrani originalni atributni prostor, najbolj pogosto uporabljena pa je polinomska jedrna funkcija s poljubno polinomsko stopnjo.

Kljub temu, da vse naštete preproste metode na neki način vsebujejo možnost optimizacije klasificiranja, je klasifikacijsko točnost moč izboljšati tudi s splošnimi algoritmi, kot sta *bagging* in *boosting*, ali pa z nadgradnimi algoritmi, kot je metoda naključnega gozda, ki je namenjena izboljšanju klasifikacije odločitvenih dreves. Sledi kratek opis zadnjih dveh metod, uporabljenih za izboljšanje klasifikacijske točnosti metode z najboljšimi rezultati.

Boosting

Cilj metode *boosting* je pretvorba slabih klasifikatorjev, ki le za malo presegajo naključno klasifikacijo, v dobre klasifikatorje. Ideja metode temelji na pripisovanju uteži učnim primerom v odvisnosti od uspeha njihove klasifikacije. Metoda je iterativna z začetno nastavitvijo vseh uteži, enako 1.0. V vsaki iteraciji se napove vrednost odvisne spremenljivke in se na podlagi pravilnosti napovedi posameznemu primeru zmanjša utež ob pravilni napovedi ter poveča pri napačni. V vsaki naslednji iteraciji se glede na uteži učni algoritem osredotoči na napačno klasificirane primere. Ustavitveni pogoj iteracije je dovolj majhna klasifikacijska napaka, razen v primerih izredno velike ali izredno majhne napake. V primerih izredno velike napake se celotna hipoteza zavrže, saj ne prispeva nobene koristne informacije, prav tako se zavrže hipoteza z izredno majhno napako, ki potencialno pomeni preveliko prileganje učni množici. Rezultat iteracije je zaporedje hipotez, ki se skupno uporabi za napovedovanje odvisne spremenljivke novega primera na podlagi glasovanja, ki je uteženo s točnostjo posamezne hipoteze.

Naključni gozd

Metoda naključnega gozda je bila razvita z namenom izboljšanja klasifikacijske točnosti odločitvenih dreves, ki se pogosto preveč prilegajo učni množici. Rezultat učnega algoritma je zaporedje odločitvenih dreves, v praksi je velikost zaporedja enaka 100. Odločitvena drevesa v zaporedju se zgradijo z izbiro najboljšega atributa iz naključne podmnožice atributov za vsako vozlišče. Priporočljiva velikost naključne podmnožice atributov je logaritem števila vseh atributov. Za napoved odvisne spremenljivke novega primera se uporabijo vsa drevesa, ki vsako zase glasujejo za svojo napovedano vrednost. Pri klasifikaciji je rezultat metode verjetnostna distribucija po vseh razredih.

S tem je izbira metod zaključena. Najboljša metoda za klasifikacijo skokovitih sprememb signala membranske kapacitvnosti bo izbrana s pomočjo ocen strojnega učenja. Čeprav se nenehno razvijajo in nadgrajujejo napre-

dnejše metode, s katerimi je mogoče izboljšati rezultate najbolj preprostih in osnovnih metod ter rešiti probleme, ki jih osnovne in preproste metode ne zmorejo, sta implementacija in uporaba teh metod izven obsega tega diplomskega dela.

4.2 Rezultati izbranih metod strojnega učenja

Rezultati strojnega učenja pri klasifikaciji skokovitih sprememb signala membranske kapacitivnosti se pregledajo in ocenijo s pomočjo najbolj pogostih ocen strojnega učenja. Sledi kratek opis vsake izmed uporabljenih ocen.

- **Matrika zmot** grafično predstavi rezultate klasifikacije, kot prikazuje slika 4.2. Vsak stolpec matrike predstavlja dejansko razporeditev primerov po razredih, vsaka vrstica pa predstavlja napovedano razporeditev po razredih. Taka vizualizacija omogoča jasen in točen pregled nad napačnimi klasifikacijami.

	NAPOVEDANI negativni razred	NAPOVEDANI pozitivni razred
DEJANSKI negativni razred	RN	NP
DEJANSKI pozitivni razred	NN	RP

Slika 4.2: Primer matrike zmot za binarno klasifikacijo. Oznaka RN (resnični negativni) pomeni število primerov, pravilno klasificiranih v negativni razred, oznaka NP (napačni pozitivni) pomeni število primerov, napačno klasificiranih v pozitivni razred, oznaka NN (napačni negativni) pomeni število primerov, napačno klasificiranih v negativni razred ter oznaka RP (resnični pozitivni) pa pomeni število primerov, pravilno klasificiranih v pozitivni razred.

- **Klasifikacijska točnost** predstavlja delež pravilno klasificiranih primerov glede na skupno število primerov. Pogosto se ta ocena primerja z odstotkom večinskega razreda. Klasifikacijska točnost, ki je manjša od odstotka večinskega razreda, teoretično pomeni, da je ocenjeni klasifikator slabši od naključne klasifikacije. Pri binarni klasfikaciji je formula za klasifikacijsko točnost naslednja:

$$klasifikacijska\ točnost = \frac{RP + RN}{RP + RN + NP + NN} , \tag{4.3}$$

kjer se vrednosti preberejo iz matrike zmot.

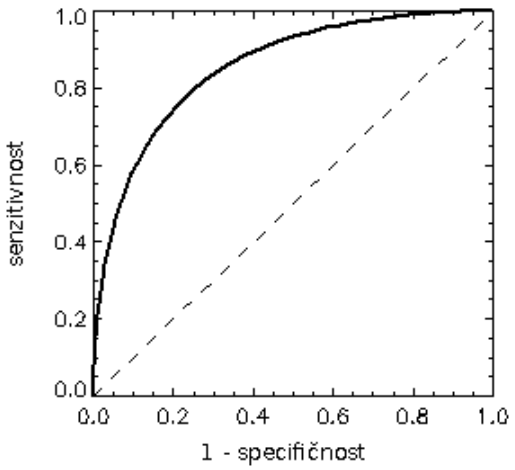
- **Senzitivnost in specifičnost** sta statistični meri uspešnosti binarne klasifikacije. Senzitivnost meri odstotek pravilno klasificiranih pozitivnih primerov, specifičnost meri odstotek pravilno klasificiranih negativnih primerov. Formuli sta naslednji:

$$senzitivnost = \frac{RP}{RP + NN} \qquad specifičnost = \frac{RN}{RN + NP} , \tag{4.4}$$

kjer se vrednosti preberejo iz matrike zmot.
V praksi je z nastavitvijo pragu verjetnosti pozitivno klasificiranega primera treba poiskati kompromis med obema ocenama. Grafično to prikazuje krivulja ROC, opisana v naslednji alineji.

- **Krivulja ROC** (angleško *receiver operating characteristic*) grafično prikazuje razmerje med senzitivnostjo in specifičnostjo. Primer krivulje ROC prikazuje slika 4.3. Običajni prag verjetnosti pozitivno klasificiranega primera je 50 %. S premikanjem tega pragu se spreminjajo klasifikacije primerov in posledično specifičnosti ter senzitivnost. Vsakemu pragu ustreza točka na krivulji ROC. Točka (0,0) na krivulji pomeni, da so vsi primeri klasificirani negativno, posledično so vsi negativni primeri klasificirani pravilno, vsi pozitivni pa napačno. Točka (1,1) na krivulji pomeni, da so vsi primeri klasificirani pozitivno, posledično so vsi pozitivni primeri klasificirani pravilno, vsi negativni primeri pa napačno. Diagonala, ki povezuje ti dve ekstremni točki,

predstavlja naključno klasifikacijo z različnimi pragovi. Krivulja ROC uspešnih klasifikatorjev mora biti levo zgoraj nad diagonalo.



Slika 4.3: Primer krivulje ROC. Črtkana diagonala predstavlja naključno klasifikacijo.

Točka (1,0) predstavlja senzitivnost in specifičnost idealnega klasifikatorja, ki bi pravilno klasificiral vse primere. Praktično se klasifikatorji le približajo tej točki. Čim bolj se klasifikator približa tej točki, bolj uspešen je. S pomočjo krivulje ROC je moč meriti uspešnost klasifikatorja z oceno AUC (angleško *area under the ROC curve*), ki meri ploščino pod krivuljo ROC.

- **Priklic in preciznost** sta pomembni oceni binarne klasifikacije, kjer klasifikacija pomeni detekcijo pomembnih primerov. V takem primeru se s klasifikacijo primerov v negativni ali pozitivni razred ločijo nepomembni primeri od pomembnih. Priklic je enak senzitivnosti in ocenjuje odstotek detektiranih pomembnih primerov glede na vse pomembne primere. Preciznost pa ocenjuje odstotek pomembnih primerov glede na vse primere, klasificirane v razred pomembnih primerov.

Formula za preciznost je naslednja:

$$preciznost = \frac{RP}{RP + NP} \ , \tag{4.5}$$

kjer se vrednosti preberejo iz matrike zmot.
Preciznost v primeru klasifikacije skokovitih sprememb meritve membranske kapacitivnosti ustrezno ocenjuje uspešnost klasifikacije. Cilj klasifikacije je namreč detekcija skokovitih sprememb, ki so posledica endocitoze ali eksocitoze.

- **Brierjeva mera** pri klasifikaciji ocenjuje verjetnostno distribucijo po razredih. Formula za Brierjevo mero je naslednja:

$$Brierjeva\ mera = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}(P_i(r_j) - P_i'(r_j))^2}{N} \ , \tag{4.6}$$

kjer je
N. . . število primerov
M. . . število razredov
$P_i(r_j)$. . . dejanska napoved za j-ti razred pri i-tem primeru
$P'_i(r_j)$. . . napovedana verjetnost za j-ti razred pri i-tem primeru

Čim manjša je Brierjeva mera, boljši je klasifikator.

Z naštetimi ocenami je moč pregledati uspešnost klasifikacije izbranih metod in različnih učnih množic glede na uravnoteženost negativnega ter pozitivnega razreda.
Modeliranje in testiranje je opravljeno v programskem jeziku R s pomočjo knjižnic CORElearn, e1071, ada in pROC ter v programskem jeziku Matlab s funkcijo *TreeBagger*.
Algoritem najbolj uspešne metode je v programskem jeziku Matlab ustrezno povezan z ostalimi funkcijami, ki služijo namenu detekcije skokovitih sprememb signala membranske kapacitivnosti.

Za namen natančnejšega testiranja se opravi 10-kratno prečno preverjanje. Pri 10-kratnem prečnem preverjanju se celotna množica podatkov razdeli na 10 delov, nato se 9 delov uporabi za modeliranje, 1 del pa se uporabi za testiranje uspešnosti modeliranja. Modeliranje s testiranjem se opravi 10-krat, vsakič z drugo izbiro dela množice podatkov za testiranje.

Oznake posameznih metod:
DT... odločitveno drevo
NB... naivni Bayes
KNN... k-najbližji sosedi
SVM1... metoda podpornih vektorjev z linearno jedrsko funkcijo
SVM2... metoda podpornih vektorjev s polinomsko jedrsko funkcijo 2. reda
SVM3... metoda podpornih vektorjev s polinomsko jedrsko funkcijo 3. reda
RF... naključni gozd
ADAB... boosting z metodo AdaBoost [13]

Oznake posameznih ocen:
CA... klasifikacijska točnost
SNS... senzitivnost
SPC... specifičnost
PRC... preciznost
AUC... ploščina pod krivuljo ROC
BRS... Brierjeva mera

Originalna učna množica

Tabela 4.1 predstavlja ocene strojnega učenja pri originalni učni množici, kjer je 98,02 % negativno klasificiranih primerov in 1,98 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.980	0.915	0.977	0.979	0.979	0.979	0.979	0.979
SNS	0.135	0.264	0.006	0	0	0	0.007	0.003
SPC	0.997	0.929	0.997	0.999	0.999	0.999	0.999	0.999
PRC	0.431	0.068	0.028	0	0	0	0.185	0.130
AUC	0.707	0.526	0.357	0.048	0.245	0	0.503	0.260
BRS	0.035	0.106	0.047	0.039	0.038	0.038	0.039	0.036

Tabela 4.1: Ocene strojnega učenja z originalno učno množico.

Iz ocen je razvidno, da učenje na originalni učni množici ni uspešno. Klasifikacijska točnost pri nobeni metodi ne preseže večinskega klasifikatorja. Razmerje senzitivnost specifičnost prikazuje dejstvo, da je bil pozitivni razred skoraj popolnoma ignoriran, kar je bilo pričakovano pri tako neuravnoteženi učni množici. Sledi pregled ocen učenja pri različno prevzorčenih učnih množicah.

Učna množica s podvzorčenim večinskim razredom

Pri prvem načinu podvzorčenja se odstrani tisti delež primerov večinskega razreda, ki je najbližje manjšinskim primerom. Tabela 4.2 predstavlja ocene strojnega učenja pri učni množici s prvim načinom podvzorčenja, ki vsebuje 96,20 % negativno klasificiranih primerov in 3,80 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.795	0.863	0.978	0.979	0.979	0.979	0.965	0.949
SNS	0.423	0.490	0.001	0	0	0	0.161	0.250
SPC	0.804	0.872	0.998	0.999	0.999	0.999	0.981	0.964
PRC	0.068	0.078	0.030	0.016	0	0	0.159	0.127
AUC	0.526	0.533	0.358	0.155	0.244	0.048	0.571	0.555
BRS	0.381	0.184	0.056	0.040	0.039	0.040	0.074	0.083

Tabela 4.2: Ocene strojnega učenja pri učni množici s prvim načinom podvzorčenja večinskega razreda.

Pri drugem načinu podvzorčenja se odstrani tisti delež primerov večinskega razreda, ki je najdlje od manjšinskih primerov. Tabela 4.3 predstavlja ocene strojnega učenja pri učni množici z drugim načinom podvzorčenja, ki vsebuje 96,19 % negativno klasificiranih primerov in 3,81% pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.786	0.889	0.974	0.978	0.975	0.979	0.882	0.970
SNS	0.363	0.408	0.054	0.024	0.034	0	0.138	0.109
SPC	0.795	0.900	0.993	0.998	0.994	0.999	0.898	0.988
PRC	0.038	0.080	0.175	0.262	0.099	0	0.036	0.155
AUC	0.511	0.533	0.578	0.621	0.539	0.049	0.518	0.568
BRS	0.411	0.147	0.049	0.041	0.044	0.040	0.161	0.051

Tabela 4.3: Ocene strojnega učenja pri učni množici z drugim načinom podvzorčenja večinskega razreda.

Pri tretjem načinu podvzorčenja se odstrani tisti delež primerov večinskega razreda, katerih razdalje do manjšinskih primerov so v večinskem razredu razdalj. Tabela 4.4 predstavlja ocene strojnega učenja pri učni množici s tretjim načinom podvzorčenja, ki vsebuje 96,91 % negativno klasificiranih primerov in 3,09 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.977	0.889	0.979	0.979	0.979	0.979	0.979	0.979
SNS	0.181	0.396	0	0.0007	0	0	0.024	0.015
SPC	0.993	0.899	0.999	0.999	0.999	1	0.998	0.998
PRC	0.334	0.080	0	0.014	0	0	0.237	0.371
AUC	0.659	0.533	0.343	0.056	0.195	0	0.511	0.675
BRS	0.040	0.142	0.055	0.039	0.039	0.039	0.042	0.037

Tabela 4.4: Ocene strojnega učenja pri učni množici s tretjim načinom podvzorčenja večinskega razreda.

Pri četrtem načinu podvzorčenja se naključno odstrani delež primerov večinskega razreda. Tabela 4.5 predstavlja ocene strojnega učenja pri učni množici s četrtim načinom podvzorčenja, ki vsebuje 96,81 % negativno klasificiranih primerov in 3,19 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.977	0.891	0.979	0.979	0.979	0.979	0.978	0.948
SNS	0.181	0.390	0	0	0	0	0.025	0.252
SPC	0.994	0.902	0.999	0.999	0.999	0.999	0.998	0.963
PRC	0.345	0.080	0	0.100	0	0	0.164	0.126
AUC	0.664	0.533	0.097	0.246	0.294	0.049	0.511	0.555
BRS	0.039	0.138	0.040	0.039	0.039	0.039	0.042	0.084

Tabela 4.5: Ocene strojnega učenja pri učni množici s četrtim načinom podvzorčenja večinskega razreda.

Učna množica z nadvzorčenim manjšinskim razredom

Pri nadvzorčenju manjšinskega razreda originalne učne množice se za vsak primer manjšinskega razreda generira 20 sintetičnih primerov pozitivnega razreda. Tabela 4.6 predstavlja ocene strojnega učenja pri učni množici z nadvzorčenim manjšinskim razredom, ki vsebuje 70,24 % negativno klasificiranih primerov in 29,76 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.942	0.815	0.977	0.979	0.979	0.979	0.978	0.903
SNS	0.984	0.629	0.008	0.0007	0	0	0.999	0.586
SPC	0.942	0.820	0.997	0.999	0.999	1	0.978	0.909
PRC	0.266	0.071	0.096	0.016	0	0	0.498	0.116
AUC	0.633	0.531	0.489	0.204	0.244	0	0.879	0.553
BRS	0.095	0.308	0.050	0.040	0.039	0.039	0.051	0.164

Tabela 4.6: Ocene strojnega učenja pri učni množici z nadvzorčenim manjšinskim razredom.

Učna množica s podvzorčenim večinskim razredom in nadvzorčenim manjšinskim razredom

Pri nadvzorčenju manjšinskega razreda učne množice s prvim načinom podvzorčenja večinskega razreda se za vsak primer manjšinskega razreda generira 10 sintetičnih primerov pozitivnega razreda. Tabela 4.7 predstavlja ocene strojnega učenja pri učni množici s prvim načinom podvzorčenja in nadvzorčenim manjšinskim razredom. Učna množica vsebuje 69,69 % negativno klasificiranih primerov in 30,31 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.700	0.789	0.979	0.901	0.976	0.979	0.839	0.715
SNS	0.971	0.668	0.003	0.560	0.025	0	0.997	0.698
SPC	0.695	0.792	0.999	0.909	0.996	1	0.836	0.715
PRC	0.068	0.0.067	0.029	0.116	0.049	0	0.138	0.054
AUC	0.533	0.529	0.504	0.553	0.417	0	0.916	0.523
BRS	0.574	0.346	0.053	0.182	0.207	0.211	0.216	0.390

Tabela 4.7: Ocene strojnega učenja pri učni množici s prvim načinom podvzorčenja večinskega razreda in nadvzorčenim manjšinskim razredom.

Pri nadvzorčenju manjšinskega razreda učne množice z drugim načinom podvzorčenja večinskega razreda se za vsak primer manjšinskega razreda generira 10 sintetičnih primerov pozitivnega razreda. Tabela 4.8 predstavlja ocene strojnega učenja pri učni množici z drugim načinom podvzorčenja in nadvzorčenim manjšinskim razredom. Učna množica vsebuje 69,74 % negativno klasificiranih primerov in 30,26 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.701	0.816	0.970	0.896	0.972	0.978	0.818	0.794
SNS	0.967	0.642	0.045	0.551	0.047	0.005	0.996	0.669
SPC	0.696	0.820	0.989	0.904	0.991	0.998	0.815	0.797
PRC	0.065	0.073	0.105	0.106	0.090	0.021	0.119	0.071
AUC	0.532	0.532	0.542	0.547	0.535	0.157	0.905	0.531
BRS	0.569	0.304	0.054	0.188	0.206	0.207	0.227	0.268

Tabela 4.8: Ocene strojnega učenja pri učni množici z drugim načinom podvzorčenja večinskega razreda in nadvzorčenim manjšinskim razredom.

Pri nadvzorčenju manjšinskega razreda učne množice s tretjim načinom podvzorčenja večinskega razreda se za vsak primer manjšinskega razreda generira 10 sintetičnih primerov pozitivnega razreda. Tabela 4.9 predstavlja ocene strojnega učenja pri učni množici s tretjim načinom podvzorčenja in nadvzorčenim manjšinskim razredom. Učna množica vsebuje 74,00 % negativno klasificiranih primerov in 26,00 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.923	0.820	0.951	0.937	0.976	0.979	0.971	0.917
SNS	0.965	0.626	0.093	0.455	0.019	0	0.997	0.543
SPC	0.922	0.824	0.968	0.947	0.996	1	0.971	0.925
PRC	0.209	0.071	0.091	0.151	0.061	0	0.443	0.128
AUC	0.604	0.531	0.487	0.569	0.423	0	0.984	0.559
BRS	0.121	0.305	0.077	0.135	0.161	0.163	0.062	0.139

Tabela 4.9: Ocene strojnega učenja pri učni množici s tretjim načinom podvzorčenja večinskega razreda in nadvzorčenim manjšinskim razredom.

Pri nadvzorčenju manjšinskega razreda učne množice s četrtim načinom podvzorčenja večinskega razreda se za vsak primer manjšinskega razreda generira 10 sintetičnih primerov pozitivnega razreda. Tabela 4.10 predstavlja ocene strojnega učenja pri učni množici s četrtim načinom podvzorčenja in nadvzorčenim manjšinskim razredom. Učna množica vsebuje 69,74 % negativno klasificiranih primerov in 26,55 % pozitivno klasificiranih primerov.

	DT	NB	KNN	SVM1	SVM2	SVM3	RF	ADAB
CA	0.918	0.822	0.970	0.934	0.971	0.978	0.971	0.915
SNS	0.957	0.622	0.037	0.476	0.046	0.006	0.997	0.554
SPC	0.917	0.827	0.989	0.944	0.990	0.998	0.970	0.922
PRC	0.199	0.072	0.097	0.147	0.086	0.016	0.437	0.126
AUC	0.599	0.531	0.539	0.567	0.533	0.106	0.984	0.558
BRS	0.129	0.301	0.055	0.135	0.166	0.165	0.063	0.142

Tabela 4.10: Ocene strojnega učenja pri učni množici s četrtim načinom podvzorčenja večinskega razreda in nadvzorčenim manjšinskim razredom.

Pri pregledu vseh tabel se izkaže, da je strojno učenje najbolj uspešno z uporabo metode naključnega gozda, in sicer pri učni množici, kjer je nadvzorčen manjšinski razred. S premikanjem pragu verjetnosti pozitivno klasificiranega primera se lahko ocene še izboljšajo, kar prikazuje tabela 4.11.

	meja 50 %	meja 60 %	meja 70 %	meja 80 %	meja 90 %
CA	0.9784	0.9884	0.9943	0.9974	0.9977
SNS	0.9996	0.9973	0.9936	0.9751	0.9006
SPC	0.9780	0.9883	0.9944	0.9978	0.9997
PRC	0.4986	0.6404	0.7725	0.8841	0.9739

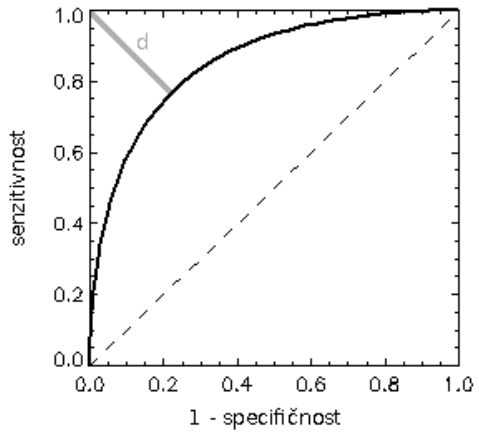
Tabela 4.11: Ocene metode naključnega gozda pri učni množici z nadvzorčenim manjšinskim razredom in različnih pragovih verjetnosti pozitivno klasificiranega razreda.

Za določitev pragu je treba je poiskati kompromis med senzitivnostjo in specifičnostjo. Kompromis se lahko poišče s pomočjo pregleda matrike zmot, ki jo prikazuje tabela 4.12.

	napovedano									
	meja 50 %		meja 60 %		meja 70 %		meja 80 %		meja 90 %	
dejansko	7665.5	181.2	7748.9	97.8	7799.2	47.5	7828.4	18.3	7843.8	2.9
	0.1	158.2	0.4	157.9	1.0	157.3	3.7	154.6	15.1	143.2

Tabela 4.12: Matrika zmot metode naključnega gozda pri učni množici z nadvzorčenim manjšinskim razredom in različnih pragovih verjetnosti pozitivno klasificiranega razreda .

Lahko pa se tudi analitično poišče točka na krivulji ROC, ki je najmanj oddaljena od točke (1,0) v prostoru krivulje ROC, kot to prikazuje slika 4.4. Točka (1,0) namreč predstavlja senzitivnost in specifičnost idealnega klasifikatorja, ki bi pravilno klasificiral vse primere.



Slika 4.4: Tipična krivulja ROC, kjer razdalja d prikazuje minimalno razdaljo med točko na krivulji ROC in točko (1,0).

Točka na krivulji ROC, ki bi bila najbližje tej točki, se lahko poišče s pomočjo minimizacije evklidske razdalje med točko (1,0) in točko na krivulji s formulo:

$$min(d) = min(\sqrt{(0-x)^2 + (1-y)^2}) \ , \tag{4.7}$$

kjer sta x in y koordinati točke na krivulji ROC.
V primeru metode naključni gozd s učno množico, ki ima nadvzorčen manjšinski razred, je prag verjetnosti pozitivno klasificiranega primera pri optimalni točki na krivulji ROC enak 83.35 %. Ocene pri tem pragu so prikazane v tabeli 4.13, rezultati pa v matriki zmot, ki jo prikazuje tabela 4.14.

	meja 83.35 %
CA	0.9977
SNS	0.9663
SPC	0.9984
PRC	0.9056

Tabela 4.13: Ocene metode naključnega gozda pri učni množici z nadvzorčenim manjšinskim razredom in pragom verjetnosti pozitivno klasificiranega primera enakem 83.35 %.

	napovedano	
	meja 83.35 %	
dejansko	7832.9	13.8
	5.0	153.3

Tabela 4.14: Matrika zmot metode naključnega gozda pri učni množici z nadvzorčenim manjšinskim razredom in pragom verjetnosti pozitivno klasificiranega primera enakem 83.35 %.

Izmed različno strukturiranih učnih množic, različnih metod klasifikacije in različnih pragov verjetnosti pozitivno klasificiranega primera se je izkazalo, da je strojno učenje najbolj uspešno pri:

- učni množici, kjer je le nadvzorčen manjšinski razred,
- pri uporabi metode naključni gozd,
- s pragom verjetnosti pozitivno klasificiranega primera nad 80 %.

Zgoraj naštete komponente in nastavitve sestavljajo sistem, ki je na podlagi meritev signala membranske kapacitivnosti in njihovih lastnosti zmožen z ustreznimi algoritmi in nastavitvami z 99,77-odstotno verjetnostjo pravilno klasificirati skokovite spremembe pri novi meritvi membranske kapacitivnosti.
Najboljši model podatkov se skupaj s ustreznimi nastavitvami in algoritmi najboljše metode integrira med obstoječe funkcije za analizo signala membranske kapacitivnosti v programskem jeziku Matlab. Metoda naključnega gozda je implementirana v obstoječi Matlabovi funkciji *TreeBagger* [14]. Funkcija se uporabi z učno množico in parametrom, ki določa število odločitvenih dreves. Parameter se nastavi na 100 odločitvenih dreves, vsi dodatni parametri funkcije niso nastavljeni oziroma se uporabi privzeta vrednost.

Poglavje 5

Zaključek

5.1 Povzetek

Cilj diplomskega dela je analiza signala membranske kapacitivnosti živih celic. S pomočjo strojnega učenja je treba detektirati tiste skokovite spremembe v signalu, ki so posledica eksocitoze ali endocitoze. Algoritem strojnega učenja naj bi znanje pridobil iz množice meritev, kjer so ustrezne skokovite spremembe že ročno označene. Pred modeliranjem je bilo podatke meritev treba predobdelati. Označene meritve so namreč relativno na velikost ustreznih skokovitih sprememb zelo šumne. Pri predobdelavi podatkov sem odstranila kalibracijske pulze, obsežno šumne dele signala in popravila bazno linijo signala. Po predobdelavi podatkov sem pripravila učno množico potrebno za strojno učenje. Moja prva naloga je bila ekstrakcija primerov. Glede na to, da je naloga strojnega učenja detekcija ustreznih skokovitih sprememb signala membranske kapacitivnosti, sem množico primerov zgradila iz skokovitih sprememb množice označenih meritev. Ideja je, da bi se detekcija opravila s principom klasifikacije, ki bi ločila skokovite spremembe v dva razreda, in sicer v negativni razred, kjer so spremembe posledica šuma, ter v pozitivni razred, kjer so spremembe posledica endocitoze ali eksocitoze. Da bi algoritem strojnega učenja lahko zgradil klasifikacijsko pravilo, sem primerom priredila njihove lastnosti, oziroma atribute. Množica podatkov bi bila

tako praviloma že pripravljena na modeliranje. Učni algoritem klasifikacijske metode bi na podlagi primerov zgradil klasifikacijsko pravilo, ki preslika množico lastnosti primerov v ustrezen izhod oziroma razred. Vendar bi bili rezultati takega pravila pri originalni množici primerov neuporabni. Ker je izredna šumnost meritve prvoten problem, je originalna množica primerov vsebovala 98,02 % negativno klasificiranih in 1,98 % pozitivno klasificiranih primerov. Ker na tako neuravnoteženi učni množici ni možno zgraditi uspešnega klasifikacijskega pravila, sem poskusila več različnih metod prevzorčenja. Poskusila sem s podvzorčenjem večinskega razreda, in sicer na štiri različne načine, z nadvzorčenjem manjšinskega razreda in s kombinacijami obeh. Katero prevzorčenje je prispevalo k najboljšim rezultatom, so pokazale ocene strojnega učenja. Pri vsakem načinu prevzorčenja sem ocenila rezultate osmih klasifikacijskih metod. Izkazalo se je, da so rezultati najboljši pri učni množici, kjer je nadvzorčen manjšinski razred, in sicer s klasifikacijsko metodo naključnega gozda. Ocene sem skušala izboljšati še s premikanjem pragu verjetnosti pozitivnega razreda. Pri pregledu matrike zmot in izračunu optimalne točke na krivulji ROC je očitno, da so ocene najboljše s pragom verjetnosti pozitivno klasificiranega razreda nad 80 %. Cilj diplomskega dela je s tem dosežen. Na podlagi množice že označnih meritev membranske kapacitivnosti celic in najbolj ustrezne klasifikacijske metode strojnega učenja sem razvila sistem, ki je zmožen v novi meritvi avtomatsko detektirati skokovite spremembe, ki so posledice eksocitoze ali endocitoze. Glede na želeno točnost sistema je detekcija lahko avtomatska ali polavtomatska. Klasifikacijska točnost sistema je 99,77 %, kjer se z 96,63-odstotno verjetnostjo detektirajo ustrezne skokovite spremembe. Na podlagi matrike zmot, sistem na 1000 dogodkov odkrije 968.4 dogodkov in hkrati dodatno napačno označi 87.1 šumnih sprememb kot iskani dogodek. Pri potrebi po boljši točnosti mora uporabnik s pomočjo ročnega pregleda odstraniti tiste skokovite spremembe, ki so bile napačno klasificirane pozitivno, in dodati tiste spremembe, ki so bile napačno klasificirane negativno. Za namen uporabe razvitega sistema in nadaljnega ročnega pregleda rezultatov sem razvila

preprost uporabniški vmesnik, ki pa ni del tega diplomskega dela.

5.2 Nadaljnje delo

Diplomsko delo je zajemalo celoten razvoj sistema za klasifikacijo skokovitih sprememb signala membranske kapacitivnosti živih celic, od predobdelave podatkov, priprave učne množice do modeliranja učne množice in testiranja rezultatov strojnega učenja. S tem razlogom je pri vsaki podnalogi veliko možnosti za izboljšavo. Najbolj preprosta izboljšava je možna z dodajanjem novih atributov, ki bi prispevali pomembno informacijo o lastnostih, ki so unikatne za pozitivno klasificirane primere. Največ možnosti za izboljšavo klasifikacije se ponuja pri izbiri metode za prevzorčenje učne množice in izbiri klasifikacijske metode. Rešitev problema neuravnotežene učne množice se je in se še vedno veliko raziskuje. Poleg drugih metod z drugačnim pristopom obstajajo nadgrajene in izboljšane verzije uporabljenih metod za prevzorčenje in nadvzorčenje, katerih kompleksnost presega diplomsko delo. Pri izbiri klasifikacijskih metod sem upoštevala le osnovne in najbolj preproste metode oziroma verzije osnovnih metod. Nekatere osnovne metode, kot je metoda podpornih vektorjev, se lahko znantno izboljšajo že samo s pravilnejšo nastavitvijo nekaterih parametrov. Za nekatere metode je bilo razvitih veliko uspešnih nadgradenj in kombinacij z drugimi metodami, prav tako pa obstaja še mnogo drugih uspešnih metod, popolnoma ločenih od uporabljenih, kot je metoda umetnih nevronskih mrež, ki je prav tako ena izmed osnovnih metod strojnega učenja. Poleg nadgradenj in izboljšav specifičnih metod je rezultate strojnega učenja možno izboljšati z bolj generičnimi metodami, ki skušajo rezultate izboljšati na podlagi večih različnih modelov posamezne metode. Uporabljeni metodi tega tipa sta metoda naključnega gozda in *boosting*, obstaja pa jih veliko več.

Nasploh izboljšanje rezultatov preprostih metod pogosto zahteva dolgotrajno eksperimentalno ugotavljanje, kompleksne implementacije in razumevanje zahtevnih principov, ki presegajo to diplomsko delo in so lahko predmet nadaljnjega dela.

Literatura

[1] G. Zupančič, L. Kocmur, P. Veranič, S. Grilc, M. Kordaš, R. Zorec "The separation of exocytosis from endocytosis in rat melanotroph membrane capacitance records.", J Physiol Lond 480.3: 539-552 , 1994.

[2] R. Zorec, M. Kreft. "Cell-attached measurments of attofarad capacitance steps in rat melanotrophs", Eur J Physiol 434:212-214 , 1997.

[3] D. Kabaso, J. Jorgačevski, A. I. Calejo, A. Flašker, A. Guček, M. Kreft, R. Zorec "Comparison of unitary exocytic events in pituitary lactotrophs and in astrocytes: modeling the discrete fusion-pore states.", Frontiers in Cellular Neuroscience, Article 33, 2013

[4] P. H. C. Eilers, H. F. M. Boelens, "Baseline Correction with Asymmetric Least Squares Smoothing.", 2005

[5] I. Kononenko , M. R. Šikonja "Inteligenti sistemi", Založba FE in FRI, 2010

[6] Endocitoza, http://sl.wikipedia.org/wiki/Endocitoza, datum dostopa 2.11.2014

[7] Eksocitoza, http://sl.wikipedia.org/wiki/Eksocitoza, datum dostopa 2.11.2014

[8] Feature scaling, http://en.wikipedia.org/wiki/Feature_scaling, datum dostopa 28.11.2014

[9] N. v. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique.", Journal of Artificial Intelligence Research, 16, 321–357, 2002.

[10] S. J. Yen, Y. S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions.", Expert Systems with Applications, vol. 36, no. 3, 5718–5727, 2009.

[11] P. H. Winston, Artificial Intelligence, Second edition. Reading, MA: Addison-Wesley, (1992).

[12] Laplaceov zakon zaporednosti, http://en.wikipedia.org/wiki/Rule_of_succession, datum dostopa 8.1.2015

[13] Metoda AdaBoost, http://en.wikipedia.org/wiki/AdaBoost, datum dostopa 18.1.2015

[14] Tree Bagger, http://se.mathworks.com/help/stats/treebagger.html, datum dostopa 18.1.2014