Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Maja Žbogar

Vodenje združevalnih metod s pomočjo naborov genov

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: izr. prof. dr. Janez Demšar

Ljubljana 2014

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00493 / 2013
Datum: 12.4.2013

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

Kandidat: **MAJA ŽBOGAR**

Naslov: **VODENJE ZDRUŽEVALNIH METOD S POMOČJO NABOROV GENOV**
GUIDING ENSEMBLE METHODS BY USING GENESETS

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Za podatke s področja genetike so tipični nabori podatkov, v katerih število spremenljivk, na primer meritev izražanj genov, bistveno presega velikost vzorcev. Takšni situaciji pravimo tudi "prekletstvo dimenzionalnosti". Kadar želimo uporabljati podatke za gradnjo predikcijskih modelov, se s prekletstvom navadno spopadamo na enega od dveh načinov. Prvi je, da uporabimo vse spremenljivke in gradimo model z združevalnimi postopki (ensemble method), kot na primer naključnimi gozdovi. Drugi je, da spremenljivke združujemo v skupine, kakršne določajo vnaprej določeni nabori genov (genesets). V diplomskem delu preverite, ali je mogoče pristopa združiti: lahko združevalne metode, ki delujejo po načelu naključnega izbora spremenljivk, vodimo tako, da izbirajo spremenljivke (gene) glede na njihovo pripadnost naborom genov? V teoretičnem delu opišite razloge za delovanje združevalnih metod, v empiričnem pa uspešnost njihovega kombiniranja z vnaprej določenimi podmnožicami spremenljivk.

Mentor:
izr. prof. dr. Janez Demšar

Dekan:
prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Izjava o avtorstvu diplomskega dela

Spodaj podpisana Maja Žbogar, z vpisno številko **63090411**, sem avtorica diplomskega dela z naslovom:

Vodenje združevalnih metod s pomočjo naborov genov

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelala samostojno pod mentorstvom izr. prof. dr. Janeza Demšarja,
- 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 5. marca 2014 Podpis avtorice:

Želela bi se zahvaliti profesorju dr. Janezu Demšarju za ves trud, ki ga vlaga v svoje poučevanje in za vse kar sem se imela od njega priložnost naučiti tekom svojega študija. Seveda bi se mu rada zahvalila tudi za pomoč in mentorstvo pri izdelavi diplomske naloge. Hvaležna sem tudi vsem ostalim izjemnim profesorjem in profesoricam, ki v svojem delu vidijo poslanstvo in so nam postavili vznemirljiv poligon za pridobivanje novega znanja ter v nas prebudili željo po učenju. Svojim sošolcem in sošolkam za iskreno prijateljstvo, dragocene spodbude, skupno delo in podporo v času študija. Po njihovi zaslugi bom to ekipno preizkušnjo (in vse skupinske treninge v sklopu priprav) brez dvoma ohranila v lepem spominu.

Posebnemu prijatelju, ki mi že dolga leta potrpežljivo nudi zatočišče, nikoli ne pozabi opomniti, naj nastavim »getterje« in »setterje« in me je v procesu izdelave diplomske naloge vztrajno motiviral z modro mislijo »Črka na črko, Vojna in mir«. Vsem drugim izjemnim ljudem, ki sem jih imela priložnost pobližje spoznati in z njimi deliti najgloblje skrivnosti (Rada vas imam, čeprav sem včasih poplen antitalent, da bi vam to pokazala). In seveda tudi vsem tistim ljudem, ki sem jih v zahvali morebiti pozabila omeniti in bi mi to utegnili zameriti.

"I have not failed. I've just found 10,000 ways that won't work."

Thomas Alva Edison

Kazalo

Povzetek

Diplomsko delo obravnava področje skupinskih (*ensemble*) metod strojnega učenja, ki za razliko od klasičnih metod strojnega učenja, temeljijo na povezovanju večjega števila osnovnih modelov v zbirno shemo. S teoretičnega vidika so predstavljeni morebitni razlogi za dobre dosežke strategije združevanja in opisani glavni pristopi h gradnji skupinskih modelov, skupaj z najbolj znanimi predstavniki posameznih usmeritev. V okviru praktičnega dela se diplomsko delo primarno osredotoča na filozofijo izbire podmnožic atributov, ki je značilna za metodo naključnih podprostorov (*random subspace*) ter poskuša raziskati možno modifikacijo te metode, ki temelji na ideji informirane gradnje osnovnih modelov na smiselno povezanih skupinah atributov. Smiselnost skupin je definirana s pomočjo uporabe genskih setov (*gene sets*). Modificirani modeli so zgrajeni na podatkovnih zbirkah, ki vsebujejo podatke mikročipov DNA. Predstavljene so uporabljene genetske podatkovne zbirke in orodje GSEA za analizo podatkov mikromrež DNA. Uspešnost posameznih modifikacij je testirana na različnih podatkovnih zbirkah. Predstavljeni so doseženi rezultati in primerjava z ostalimi metodami strojnega učenja, ki vodi do zaključka, da se predlagana modifikacija ni izkazala za uspešnejšo od osnovne variante. Identificirani so morebitni razlogi za neuspeh in izpostavljene glavne v postopku raziskovanja pridobljene ugotovitve.

Ključne besede: skupinske metode strojnega učenja, skladanje klasifikatorjev, genski seti, tehnologija mikromrež DNA, analiza GSEA

Abstract

The following bachelor thesis is mainly focused on the field of ensemble machine learning. In contrast to usual machine learning methods, where one strives to find a single best performing model, ensembles are based on the idea of combining multiple base models. They are generally considered to be more successful, than any of their constituent parts. The theoretical part of the thesis explores possible reasons for this superior performance and presents some of the most influential frameworks on ways of building ensembles. Core focus of the empirical research is building its foundations on the philosophy of selecting subsets of input variable space, proposed by the random subspace theory framework, but instead of using randomly selected variables, it explores the idea of using subsets of features, sharing some meaningful connection. Meaningful sets of features, required in order to build the base models, are obtained by using predefined gene sets. Modified ensemble models are built and tested on several different DNA microarray data sets, analysed by using GSEA analysis. The performance of the suggested modifications is compared to the results achieved by using other learning methods. Results indicate, that suggested approach does not yield the desired performance improvements. Possible reasons for the absence of results are investigated and some main findings of the conducted research are highlighted.

Key words: ensemble learning methods, stacking, gene sets, DNA microarray, GSEA analysis

Poglavje 1

Uvod

Diplomska naloga se osredotoča na področje skupinskih (*ensemble*) metod strojnega učenja. Vse od svoje pojavitve skupinski modeli zbujajo veliko zanimanja v skupnosti strojnega učenja in so pogosto preferenčna izbira orodja (*weapon of choice*) za reševanje praktičnih problemov. Po zaslugi svoje uspešnosti so priljubljeno področje za modifikacije in vir inspiracije mnogih idej za različne izboljšave. Skupinske metode, za razliko od klasičnih modelov strojnega učenja, ki se koncentrirajo na gradnjo enega samega čim bolj uspešnega modela, temeljijo na filozofiji združevanja večjega števila osnovnih modelov. V okviru izdelave diplomskega dela nas je zanimalo, kaj je s teoretičnega vidika recept za uspeh strategije združevanja in kakšne so pravzaprav glavne razlike med posameznimi temeljnimi usmeritvami na področju gradnje skupinskih modelov. Poleg teoretičnega zanimanja za fenomen uspešnosti združevanja smo v okviru empiričnega dela diplomske naloge želeli raziskati drugačen postopek gradnje skupinskih klasifikacijskih modelov. Glavna ideja predlaganega postopka je vodena gradnja osnovnih modelov na smiselno povezanih podmnožicah atributov.

Ena izmed osnovnih zahtev pri gradnji skupinskih modelov je, da na-

mesto enega samega (karseda točnega) modela za združevanje potrebujemo večje število (lahko tudi nekoliko manj točnih) osnovnih modelov. Te osnovne modele moramo na naših učnih podatkih na določen način zgraditi, pri tem pa moramo zagotoviti različnost osnovnih modelov. V odgovor na to nujno potrebo se je razvilo nekaj različnih temeljnih teoretičnih pristopov, ki nam omogočajo gradnjo tovrstnih raznovrstnih modelov. Za učenje posameznega osnovnega modela lahko na primer naključno izberemo različne podmnožice učnih primerov. Na ta način bomo z uporabo primernega algoritma za gradnjo osnovnih modelov, ki zadovoljuje teoretični pogoj nestabilnosti [10], na vsaki učni podmnožici dobili različne klasifikatorje. Tipična predstavnika in hkrati tudi utemeljitelja te kategorije sta *bagging* [3] in naključni gozdovi (*random forest*) [4]. Namesto izbiranja podmnožic učnih primerov lahko v procesu učenja učnim primerom spreminjamo uteži. Slednje je temeljna ideja metode *boosting* [15]. Tretja možnost je gradnja skupinskih modelov na naključno izbranih podmnožicah atributov, saj lahko tudi z izbiro različnih atributov, dobimo različne osnovne modele. Najbolj viden predstavnik te kategorije je metoda naključnih podprostorov (*random subspace*) [20].

V okviru raziskovalnega dela diplomske naloge smo primarno črpali iz filozofije izbiranja podmnožic atributov, ki je značilna za metodo naključnih podprostorov in poskušali raziskati potencialno modifikacijo te metode. Karakteristika obstoječe metode je, da podmnožice atributov, ki jih potrebuje za gradnjo osnovnih modelov, izbira naključno, brez kakršnekoli posebno smiselne povezave med njimi. Za razliko od originalne metode smo namesto tega želeli vpeljati nekoliko drugačno filozofijo izbiranja. Naš cilj je bil raziskati, kako se obnaša skupinska metoda, če izbiramo smiselne in na nek način povezane podmnožice atributov. V proces gradnje osnovnih modelov smo torej želeli vpeljati neke vrste informirano izbiro. S tem smo želeli nasloviti težavo originalne metode, ki jo lahko prinaša tovrstno naključno izbiranje. Nezaželena posledica naključnega izbiranja atributov je namreč, da so lahko osnovni modeli zelo slabi, saj je možno, da je proces gradnje

posameznih osnovnih modelov omejen na uporabo podmnožice popolnoma nepovezanih atributov, ki jih algoritem za gradnjo težko uporabi za konstrukcijo dobrega modela [17]. Zanimalo nas je, ali lahko z informirano izbranimi podmnožicami atributov, zgradimo bolj uspešen skupinski model, kot ga dobimo z uporabo originalne metode naključnega izbiranja. Pri tem smo izhajali iz domneve, da bomo lahko s pomočjo tega dodatnega znanja, zgradili boljše osnovne klasifikatorje in ta način dosegli večjo uspešnost končnega združenega modela.

Ena izmed možnosti za pridobivanje tovrstnih smiselnih podmnožic atributov je uporaba specifičnih zbirk učnih podatkov. Veliko možnosti za črpanje tovrstnega ekspertnega predznanja, ki ga potrebujemo za določitev smiselnih podmnožic atributov, nam ponuja področje genetike. Pri delu na tej domeni imamo možnost uporabe podatkovnih zbirk, ki so pridobljene z uporabo tehnologije mikromrež DNA. Na področju uporabe genetskih podatkovnih zbirk za reševanje klasifikacijskih problemov, lahko izkoristimo zbirke definiranih genskih setov (*gene sets*), ki nam posredujejo to dodatno znanje in omogočajo izbiro smiselnih podmnožic atributov.

Analiza genetskih podatkov je sama po sebi zelo kompleksno področje in prav gotovo obstaja veliko motivacije za reševanje problemov poveznih z genetskimi vprašanji, saj so geni zelo povezani z odgovori na uganko človekovega (ne)zdravja. Ena izmed temeljnih značilnosti genetskih podatkov je visoka dimenzionalnost, saj se število atributov pogosto meri v več tisočih genov. Uporaba manjših genskih setov za gradnjo osnovnih modelov lahko zelo zmanjša dimenzionalnost podatkov na obvladljiv nivo. Poleg tega je človeško telo nadvse kompleksno povezan sistem. Pri reševanju klasifikacijskih problemov na domeni genetskih podatkov, ki opisujejo na primer določeno bolezensko stanje, lahko domnevamo, da razred bolezni vpliva na izražanje večih med seboj povezanih genov, ki so del nekega biološkega procesa. S posameznimi procesi povezane gene opisujejo genski seti. Slednje je temelj za domnevo, da bi lahko morda klasifikator, ki je zgrajen na celotni skupini z določenim

biološkim procesom povezanih genov, dajal bolj točne napovedi kot klasifikator, ki je zgrajen na povsem nepovezanih genih. V okviru diplomskega dela smo želeli tovrstne biološko obarvane osnovne klasifikatorje združiti v skupinski model in ugotoviti, kako uspešen je tak kolektivni model v primerjavi z ostalimi metodami strojnega učenja pri reševanju klasifikacijskih problemov na različnih genetskih podatkovnih zbirkah.

V poglavju 2 smo poskusili na kratko predstaviti področje strojnega učenja s poudarkom na klasifikaciji in se nato lotili področja skupinskega strojnega učenja (poglavje 3). Pri tem nas je najbolj zanimalo, kje tiči ključ uspešnosti združevanja osnovnih modelov in kaj so glavne značilnosti različnih temeljnih usmeritev na področju skupinskih metod. V nadaljevanju smo predstavili uporabljene genetske podatkovne zbirke in njihovo obdelavo s pomočjo analize GSEA (*Gene Set Enrichment Analyisis*) (poglavje: 4). Glavni del diplomskega dela tvori empirična raziskava o možnostih uporabe smiselno povezanih podmnožic atributov pri gradnji skupinskih modelov, kjer so te smiselne skupine atributov definirane z genskimi seti (poglavje 5). V okviru tega nas je naprej zanimala okvirna ocena uspešnosti različnih metod strojnega učenja na genetskih podatkih (podpoglavje: 5.2). Nato smo preučevali vpliv izbiranja atributov na uspešnost modelov (podpoglavje: 5.3). V naslednjem koraku smo se posvetili metodi naključnih podprostorov (podpoglavje: 5.4). Potem smo se lotili modifikacije metode naključnih podprostorov (podpoglavje: 5.5). Najprej smo gradili modele na naključno izbranih genskih setih, nato še na rangiranih genskih setih, ki smo jih razvrščali s pomočjo analize GSEA ter primerjali rezultate obeh pristopov. V naslednji fazi smo metodo informirane gradnje osnovnih modelov na genskih setih nadgradili še z uporabo odločitvenih pravil CN2 na osnovnem nivoju (podpoglavje: 5.6). Možni razlogi, zakaj gradnja skupinskih modelov na genskih setih, kljub vloženemu trudu, ni dosegla želenih rezultatov, so predstavljeni v podpoglavju 5.7). V luči teh ugotovitev smo identificirane probleme želeli nasloviti z uporabo metode skladanja klasifikatorjev (podpo-

glavje: 5.8). V sklepnem delu so povzete nekatere glavne ugotovitve, ki smo jih pridobili tekom našega raziskovanja (podpoglavje: 5.9).

Poglavje 2

Strojno učenje in klasifikacija

2.1 Strojno učenje in klasifikacija

Osnovne pojme pogosto razlagamo s pomočjo metafor. Predstavljajmo si torej, da smo zdravnik in da želimo na primer zaradi novih direktiv o varčevanju v javnem sektorju, razviti sistem za avtomatsko napovedovanje pacientovega zdravstvenega stanja. Zanima nas torej zakonitost, ki uravnava ali je pacient zdrav, je zgolj prehlajen ali pa ima morda gripo. Za vsakega pacienta, ki ga obravnavamo, v ta namen zbiramo podatke o njegovi telesni temperaturi, kašljanju, kapljanju z nosa, glavobolu ipd. Poleg simptomov pa, potem ko smo za posameznega pacienta postavili diagnozo, zabeležimo še, v katerega izmed treh razredov sodi: zdrav, prehlad, gripa.

Meritve in opažanja o pacientovih lastnostih so naši atributi, diagnoza oziroma razred, pa je tisto, kar nas zanima. Izhajamo torej iz predpostavke, da pacientovi simptomi določajo njegovo zdravstveno stanje. V našem primeru poznamo razrede, oziroma možne vrednosti odvisne spremenljivke, zato tej vrsti učenja pravimo nadzorovano učenje. Pri nenadzorovanem učenju pa

bi od našega avtomatskega sistema pričakovali, da samostojno poišče možne razrede. Možne diagnoze za posameznega pacienta so diskretne, zato temu rečemo klasifikacijski problem, saj želimo pacienta glede na vrednosti atributov, razvrstiti v določeno kategorijo.

Formalno je naš učni problem opredeljen z množico učnih primerov (pacientov) oblike $((x_1, y_1), \ldots, (x_n, y_n))$, za katero želimo poiskati funkcijo $y = f(x)$. Vrednost spremenljivke y_i predstavlja vrednost razredne spremenljivke. Vrednosti x_i so vrednosti atributov in so navadno predstavljene z vektorjem oblike $(x_{i,1}, x_{i,2}..., x_{i,n})$, katerega komponente so realne ali diskretne vrednosti atributov kot na primer telesna temperatura, teža, barva oči ipd [10]. V nadzorovanem stojnem učenju torej izhajamo iz predpostavke, da je izid y (zdravstveno stanje), ki jemlje vrednosti iz prostora Y (zdrav, prehlajen, gripa), odvisna (delno ali v celoti) od n vrednosti atributov $X = (X_1, \ldots.., X_n)$, katerih zaloga vrednosti je R.

2.2 Ocenjevanje uspešnosti strojnega učenja

Seveda naš model prav gotovo ni popoln in ni nujno, da bo imel vedno prav. Da bi izvedeli, kako pogosto se bomo v svojih napovedih zmotili, moramo model na nek način oceniti. Napovedi našega modela zato preverimo na ločeni testni množici. Točnost modela moramo zaradi objektivnosti preveriti na drugih podatkih in ne na tistih, na katerih smo se učili. Z ocenjevanjem modela na učnih podatkih bi namreč dobili pretirano optimistično predvidevanje o uspešnosti našega modela. Na ta način pa lahko podcenimo napoved, kako pogosto se bomo zmotili.

Ena izmed priljubljenih metod za oceno uspešnosti učenja, s pomočjo katere lahko dobimo približno predstavo o tem, kako dobre rezultate lahko pričakujemo od našega modela, je klasifikacijska točnost. Pri računanju kla-

sifikacijske točnosti ocenjujemo delež pravilnih odgovorov. Formalno je klasifikacijska točnost definirana kot

$$CA = \frac{N^{(p)}}{N} \times 100\%$$

kjer je N število vseh testnih primerov in $N^{(p)}$ pa število pravilno klasificiranih primerov. Klasifikacijsko točnost lahko razumemo kot verjetnost, da bo naključno izbran primer, pravilno klasificiran [29].

Ta ocena je zelo preprosta in lahko razumljiva, vendar pa je njena uporaba lahko včasih zavajajoča, saj ima kar nekaj slabosti. Rezultat, ki ga dobimo z računanjem klasifikacijske točnosti, je namreč povprečen preko vseh razredov, zato nam ne pove ničesar o tem, kako dobro so klasificirani primeri iz posameznega razreda. Poleg tega ne upošteva števila možnih razredov, distribucije primerov med razredi in podobno.

Poleg klasifikacijske točnosti, obstaja še kar nekaj drugih ocen za ocenjevanje uspešnosti učenja. Za boljšo (bolj objektivno) primerjavo uspešnosti dveh klasifikatorjev na dvorazrednih problemih se pogosto uporablja ocena AUC (ploščina pod krivuljo ROC), ki poenostavljeno rečeno, upošteva še druge aspekte uspešnosti klasifikacije (senzitivnosti in specifičnost). Za boljši klasifikator velja tisti, ki ima večji AUC. Izkaže se, da je AUC enaka verjetnosti, da bo klasifikator, ki zna napovedovati verjetnosti, pravilno razločil med pozitivnim in negativnim primerom (t.j. pozitivnemu bo pripisal večjo verjetnost, da je pozitiven) [29].

2.3 Narava in predpostavke strojnega učenja

Cilj gradnje modela, s katerim bi si pomagali pri reševanju praktičnih problemov (na primer napovedovanje pacientovega zdravstvenega stanja), lahko vidimo kot poskus iskanja neznane funkcije $f(x)$, ki zna na podlagi vrednosti

atributov napovedati vrednost neodvisne spremenljivke (razred). Ena izmed pomembnih predpostavk v teoriji stojnega učenja je hipoteza, da lahko na podlagi napake, ki jo algoritem naredi na učni množici, posredno sklepamo o napaki na testni množici. Cilj učnih algoritmov je torej minimizirati napako na učni množici in na ta način poskušati modelirati optimalno neznano funkcijo F. Tej neznani funkciji bi se radi seveda karseda približali. Edina orientacijska točka, ki jo imamo ponavadi na voljo, je naša učna množica. Učni algoritmi tako med procesom učenja stremijo k minimizaciji napake na učni množici in posplošitvi pridobljenega znanja na nove primere iz testne množice.

Na žalost pa prave oblike neznane ciljne funkcije F ne poznamo. Iščemo torej neko optimalno funkcijo, s katero lahko modeliramo vrednosti odvisnih spremenljivk v naši učni množici. Optimalna funkcija F je lahko linearna, kvadratna ali bolj kompleksna. O tem lahko le špekuliramo, saj prave informacije nimamo. Posledično zato ne moremo vedeti, kakšno funkcijo uporabiti, da zajamemo, oziroma z njo ustrezno modeliramo podatke iz naše učne množice.

Recimo, da je naša optimalna funkcija F kvadratna, mi pa za modeliranje učnih podatkov uporabimo linearno. Naša napaka pri ocenjevanju novih primerov bo velika, saj ima naša hipoteza veliko pristranskost (*bias*) t.j. naša linearna funkcija je daleč od prave oblike neznane funkcije in z njo nikoli ne bomo mogli uspešno modelirati podatkov učne množice, saj rešitve iščemo v prostoru funkcij, ki ne vsebuje neznane ciljne funkcije. Na ta način smo podučili (*underfitting*) učno množico [32]. Koncept pristranskosti je upodobljen na Sliki 2.1.

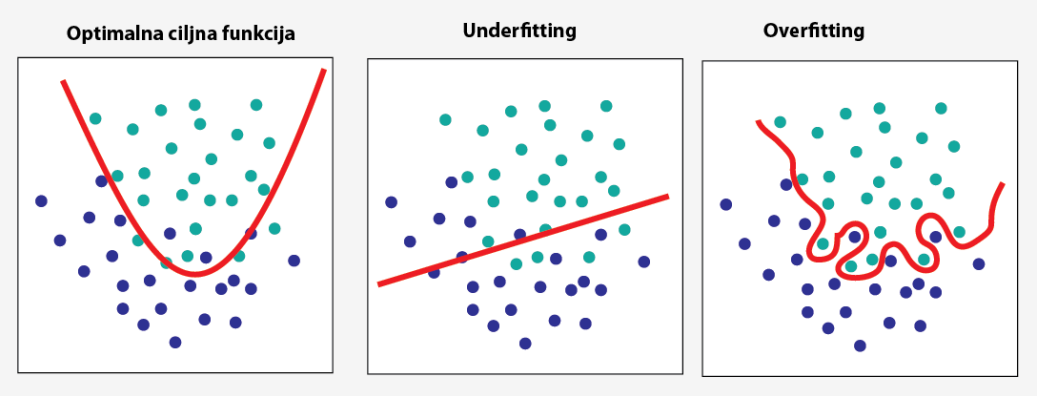
Na prvi pogled obstaja za ta problem zelo preprosta rešitev. Kaj če bi torej uporabili zelo kompleksno funkcijo, s katero bomo gotovo natančno zajeli vse podatke iz naše učne množice? Na ta način bomo vsekakor dosegli popolno točnost naše hipoteze na učni množici. Z drugimi besedami, kaj če

bi se popolnoma izognili problemu iz prejšnjega odstavka in uporabili zelo kompleksno funkcijo. Z uporabo zelo kompleksne funkcije bi namreč skoraj gotovo popolnoma natančno razločili med primeri v naši učni množici. V primeru, da iščemo rešitev v zelo velikem prostoru funkcij, bo naš prostor gotovo zajemal tudi optimalno ciljno funkcijo. Na žalost pa se izkaže, da to ni prava rešitev, saj na ta način naletimo na drug problem.

Poleg tega, da je v praksi težko zasnovati tak postopek, saj je za tovrstno kompleksno funkcijo potrebno izračunati veliko parametrov, lahko za povrhu z veliko verjetnostjo rečemo, da naša (pre)kompleksna funkcija na drugačni učni množici, ne bo več popolna. Obstaja torej resna nevarnost, da bomo na drugačni učni množici dosegli zelo slabe rezultate. Slednje bo še posebej držalo, če se primeri v novi učni množici zelo razlikujejo od prvotno uporabljene učne množice. Na ta način smo torej zasnovali učni proces, ki ima veliko varianco in je preveč odvisen od lastnosti primerov učne množice. S tem smo povzročili problem pretiranega prileganja (*overfitting*) učni množici [32]. Posledica je, da svojega znanja in pričakovane točnosti na testni množici ne moremo posplošiti, saj je ocenjena natančnost relevantna le za trenutno točno določeno učno množico in se bo z veliko verjetnostjo na drugačnih primerih obnašala popolnoma drugače. S tovrstno pretirano specializacijo (prilagoditvijo funkcije učni množici), izgubimo sposobnost generalizacije naučenega znanja in posledično o uspešnosti modela na testni množici, ne moremo sklepati popolnoma ničesar. Koncepti optimalne hipoteze, underfittinga in overfittinga so ilustrirani na Sliki 2.1.

Med željo, da bi bila naša funkcija (hipoteza) dovolj kompleksna, da bi obsegala tudi neznano ciljno funkcijo (majhna pristranskost) in željo, da bi bila hkrati tudi dovolj preprosta (majhna varianca), da ne bi bila preveč odvisna od primerov v učni množici, obstaja kompromis. Če želimo zmanjšati varianco, se bo posledično zmanjšalo število možnih hipotez. S tem tvegamo, da naš prostor možnih hipotez ne vsebuje ciljne. Naš komplet instrumentov je torej premajhen, zato nimamo možnosti, da bi izbrali pravega. In naspro-

tno, če želimo povečati število možnih hipotez (manjša pristranskost), se bo hkrati povečala tudi varianca, česar pa si tudi ne želimo. V tem scenariju imamo na razpolago preveč orodja in nikakor ne moremo poiskati pravega. Problem nadzorovanega učenja lahko torej vidimo kot iskanje optimalnega ravnovesja med medsebojno izključujočimi se cilji, manjše variance in večje pristranskosti. Poiskati želimo torej učni proces, ki je dovolj fleksibilen, da zajame primere učne množice in hkrati naučeno znanje ne variira preveč, če to učno množico spremenimo.



Slika 2.1: Neznana optimalna ciljna funkcija, underfitting in overfitting

Na proces gradnje modelov z algoritmi strojnega učenja na učni množici, morda lahko gledamo kot na problem iskanja optimalnega ravnovesja med konfliktnimi cilji. Model zgradimo na učni množici, kjer želimo, da bi bil kar najbolj točen. Vendar pa, bolj ko je točen na učni množici, bolj se tej učni množici prilega in manj lahko to točnost posplošimo na testno množico. Obenem seveda prav tako želimo, da bi bil natančen tudi na testni množici, saj drugače svojega znanja ne moremo posplošiti. Domnevamo torej, da preveč kompleksen model ne bo natančen na testni množici, če bo preveč enostaven pa tvegamo, da morda ne bomo zajeli ciljne funkcije. To je eden izmed tipičnih problemov običajnih metod strojnega učenja. V kategorijo »običajnih metod« sodijo metode, ki zgradijo en sam model. Obstajajo seveda razni prijemi, ki poskušajo ublažiti ta konflikt. V primeru odločitvenih

dreves, lahko na primer za zmanjševanje efekta prevelike variance uporabimo postopek rezanja dreves. Poleg običajnih metod so se pojavile tudi metode, za katere se zdi, da ta konflikt vsaj minimizirajo, če že ne teoretično odpravijo. V to kategorijo metod sodijo skupinske metode strojnega učenja, ki jih bomo predstavili v naslednjem poglavju

Poglavje 3

Skupinske metode strojnega učenja

Za začetke razvoja področja strojnega učenja bi morda lahko rekli, da je bilo veliko dela vloženega v razvoj novih učnih algoritmov in iskanje njihovih izboljšav. Pri tem je bil razvoj usmerjen v uporabo enega samega klasifikacijskega modela. Želeli smo torej zgraditi na primer eno samo čim bolj uspešno odločitveno drevo. Eden izmed velikih miselnih preskokov je bil začetek uporabe večih posameznih modelov skupaj. Zadnje čase tako čedalje bolj stopajo v ospredje t.i. skupinski (*ensemble*) modeli in smo lahko priča velikemu zanimanju za preučevanje in razvijanje novih metod skupinskega učenja. Eden izmed velikih motivatorjev za uporabo in preučevanje skupinskih učnih metod je, da slovijo po svoji večji uspešnosti in se jih pogosto uporablja za reševanje praktičnih problemov. Eno izmed prvih vprašanj je seveda, kje se skriva razlog njihove uspešnosti. Za njihovo pogosto boljšo uspešnost obstajajo različne razlage. Zaradi velikega zanimanja so se namreč pojavile različne metode in zdi se, da pogosto razlagajo temelje svoje uspešnosti na različen način. V naslednjem poglavju bomo poskusili identificirati morebitne razloge za uspešnost skupinskih metod, na kratko prestaviti

njihove najbolj vidne predstavnike in se trudili med njimi povleči določene vzporednice.

3.1 Zakaj združevanje deluje?

V prejšnjem poglavju predstavljeni konflikt med pristranskostjo in varianco lahko v luči skupinskih metod pogledamo v drugačni svetlobi. Na prvo žogo bi namreč morda lahko domnevali, da z združevanjem modelov teoretično povečujemo varianco in povzročamo pretirano prileganje podatkom. Združevanje osnovnih modelov lahko namreč vidimo kot povečevanje kompleksnosti modela. Z dodajanjem osnovnih klasifikatorjev v skupinski model se neobhodno tudi čedalje bolj prilagajamo učni množici. Vendar bomo v nadaljevanju predstavili možne razlage, zakaj temu morebiti ni tako. V kontekstu skupinskih metod, lahko morda nekatere izmed tradicionalnih predpostavk, kamor sodi na primer kompromis med varianco in pretiranim prileganjem podatkom, vidimo povsem v drugačni luči. Celo več. Zdi se, da je možno doseči, da tradicionalno nezdružljiva cilja nista več medsebojno izključujoča in ju lahko morebiti celo dosežemo skupaj.

Skupinska klasifikacijska metoda je klasifikacijska metoda, ki namesto enega samega klasifikatorja (hipoteze h), zgradi množico klasifikatorjev oziroma hipotez $h_1, h_2, ..., h_L$. Odločitve oziroma napovedi posameznih klasifikatorjev za razred, v katerega sodi neznani primer x, so na koncu združene v končno odločitev. Navadno se v procesu združevanja individualnih odločitev posameznih klasifikatorjev, uporablja utežno ali pa večinsko glasovanje [10].

Bistvo skupinskih metod je torej v številčnosti, saj namesto uporabe individualnega klasifikatorja uporabimo več takih klasifikatorjev. Uporaba večjega števila klasifikatorjev skupaj se v praksi navadno izkaže za bolj uspešno kot uporaba enega samega. Kako pojasniti to uspešnost, ki izvira iz sode-

lovanja? Ena od bolj intuitivnih razlag kolektivne uspešnosti za reševanje klasifikacijskih problemov, je t.i. *Condorcet's jury theorem* [9]. Denimo, da imamo problem z dvema možnima rešitvama, od katerih je ena slaba in druga dobra (*good*, *bad*). Da bi se lažje odločili, katero izbrati, vprašamo M sodnikov, da nam zaupajo svoj glas. Na koncu izberemo tisto rešitev, ki zbere največji odstotek glasov. Verjetnost, da bo večinski glas pravilen lahko formalno definiramo kot

$$PR\left[\left(\sum_{m=1}^{M}(Vote_m = good)\right) > \frac{M}{2}\right] \tag{3.1}$$

Računamo torej verjetnost, da nam več kot polovica sodnikov, ponudi pravilen odgovor. Predpostavimo, 1) da je za vsakega izmed sodnikov verjetnost pravega odgovora p enaka in 2) da vsak izmed sodnikov glasuje neodvisno od drugih. Iz tega sledi da $\left(\sum_{m=1}^{M} I(Vote_m = good)\right)$ ustreza binomski porazdelitvi (M, p). Enačbo (3.1) lahko torej izrazimo kot

$$Pr\left[\sum_{m>M/2}^{M}\binom{M}{m}p^m(1-p)^{M-m} > \frac{M}{2}\right] \tag{3.2}$$

V tem kontekstu je verjetnost pravilne odločitve večinskega glasovanja, odvisna le od M in p. Tabela na Sliki 3.1 prikazuje vrednosti enačbe (3.2) kot funkcijo M in p. Iz tabele je razvidno, da se verjetnost pravilnega odgovora M sodnikov veča v primeru, da je $p > 0.5$ in zmanjšuje v primeru je $p < 0.5$. Če je verjetnost za pravilni glas vsakega sodnika rahlo boljša od naključnega ugibanja, t.j, da vsak sodnik glasuje za pravilni odgovor z verjetnostjo p strogo večjo od 0.5, se verjetnost pravilnosti večinskega glasovanja povečuje s številom sodnikov [32]. V primeru, da število sodnikov narašča v neskončnost, verjetnost, da dobimo pravi odgovor, konvergira k 1. Slednje nas navdaja z upanjem, da lahko ob izpolnitvi dveh teoretičnih predpostavk precej varno

pričakujemo, da bo pravici slej ko prej zadoščeno.

V kontekstu nadzorovanega učenja in klasifikacije to pomeni, da če 1) smo sposobni predvideti pravilno oznako za posamezen primer rahlo boljše kot z naključnim ugibanjem in 2) so posamezni klasifikatorji našega modela med seboj neodvisni, potem se verjetnost, da bo večinski glas za razred pravilen, povečuje s številom osnovnih klasifikatorjev, ki sestavljajo naš skupinski model [32].

		p								
		0.1	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
	1	0.100	0.200	0.300	0.400	0.500	0.600	0.700	0.800	0.900
	3	0.028	0.104	0.216	0.352	0.500	0.648	0.784	0.896	0.972
	5	0.009	0.058	0.163	0.317	0.500	0.683	0.837	0.942	0.991
	7	0.003	0.033	0.126	0.290	0.500	0.710	0.874	0.967	0.997
	9	0.001	0.020	0.099	0.267	0.500	0.733	0.901	0.980	0.999
	11	0.000	0.012	0.078	0.247	0.500	0.753	0.922	0.988	1.000
M	**13**	0.000	0.007	0.062	0.229	0.500	0.771	0.938	0.993	1.000
	15	0.000	0.004	0.050	0.213	0.500	0.787	0.950	0.996	1.000
	17	0.000	0.003	0.040	0.199	0.500	0.801	0.960	0.997	1.000
	19	0.000	0.002	0.033	0.186	0.500	0.814	0.967	0.998	1.000
	21	0.000	0.001	0.026	0.174	0.500	0.826	0.974	0.999	1.000
	23	0.000	0.001	0.021	0.164	0.500	0.836	0.979	0.999	1.000
	25	0.000	0.000	0.017	0.154	0.500	0.846	0.983	1.000	1.000

Slika 3.1: Verjetnost, da pri večinskem glasovanju M sodnikov izbere pravo rešitev, če vsak izmed posameznih sodnikov odgovori pravilno z verjetnostjo p (Vir: [32]).

Na podlagi tega lahko zaključimo, da je za dodano vrednost večinskega glasovanja in kolektiven uspeh individualnih klasifikatorjev, povezanih v skupinski model, potreben in zadosten pogoj, da so njegovi člani natančni in različni. Natančni v smislu tega, da imajo verjetnost pravilnega odgovora

večjo od naključnega ugibanja in različni v smislu tega, da delajo različne napake na novih primerih. Vkolikor so si posamezni klasifikatorji, ki sestavljajo model med seboj zelo podobni oziroma zelo korelirani, potem bo v primeru, ko je glas prvega napačen ($h_1(x)! = y$), po vsej verjetnosti napačen tudi glas vseh ostalih ($h_2(x)! = y$,...,$h_n(x)! = y$) [10].

V duhu črnega scenarija pristranskega glasovanja sodnikov iz prejšnje metafore, ki med seboj ne bi glasovali neodvisno in njihov glas ne bi bil različen (bi bil koreliran), se lahko nadejamo, da bo večinski glas zelo verjetno enak glasu posameznega sodnika, ne glede na to koliko sodnikov sestavlja naše montirano sodišče. Z neodvisnim glasovanjem in različnostjo sodnikov med seboj pa imamo stanje, ko bo glas prvega sodnika morda napačen, vendar imamo vsaj upanje, da se morda drugi in tretji t.j. preostalih več kot polovica, ne bodo zmotili. Na ta način se lahko nadejamo, da bomo na koncu dosegli sinergijske učinke združevanja posameznih glasov in uspeli dobiti pravilno sodbo.

Poleg predstavljene razlage obstajajo še druge. Zanimivo predstavitev razlogov za uspeh delovanja skupinskih modelov lahko najdemo v članku Diettericha [10]. Dietterich identificira tri razloge, ki poskušajo razložiti, zakaj so skupinske metode tako uspešne in dosegajo dobre rezultate.

Prvi razlog je statističen. Statistični razlog se navezuje na že prej opisani problem zmanjševanja problema variance pri iskanju optimalne funkcije F. Če pogledamo na problem učenja z zornega kota iskanja najboljše hipoteze, ki opisuje neznano funkcijo v neskončnem prostoru možnih hipotez $\mathcal{H}$, potem naletimo na problem, da je naša množica učnih primerov premajhna v primerjavi z velikostjo prostora $\mathcal{H}$. Posledično lahko naš učni algoritem najde mnogo različnih dobrih hipotez, ki vse dosegajo enako dobro natančnost na učni množici. Problem torej nastane, ker se je potrebno v naboru na videz enako dobrih možnosti odločiti, katero izmed njih izbrati. Skupinske metode nam omogočajo, da se izognemo problemu prisiljene izbire ene same hipo-

teze, saj jih lahko izberemo več. Na ta način naš algoritem lahko povpreči glasove vseh izbranih hipotez in se posledično izogne tveganju, da bi izbral eno samo, ki utegne biti suboptimalna. S povprečenjem podmnožice izbranih hipotez si odpre možnost, da se bolj približa ciljni hipotezi. Grafičen prikaz statističnega razloga je ponazorjen na Sliki 3.2 levo zgoraj. Notranja krivulja označuje množico hipotez, ki nam dajejo dobro natančnost na učni množici, točka f pa je neznana optimalna hipoteza. S povprečenjem hipotez $(h_1, .., h_4)$ lahko torej dobimo dober približek iskane hipteze f [10].

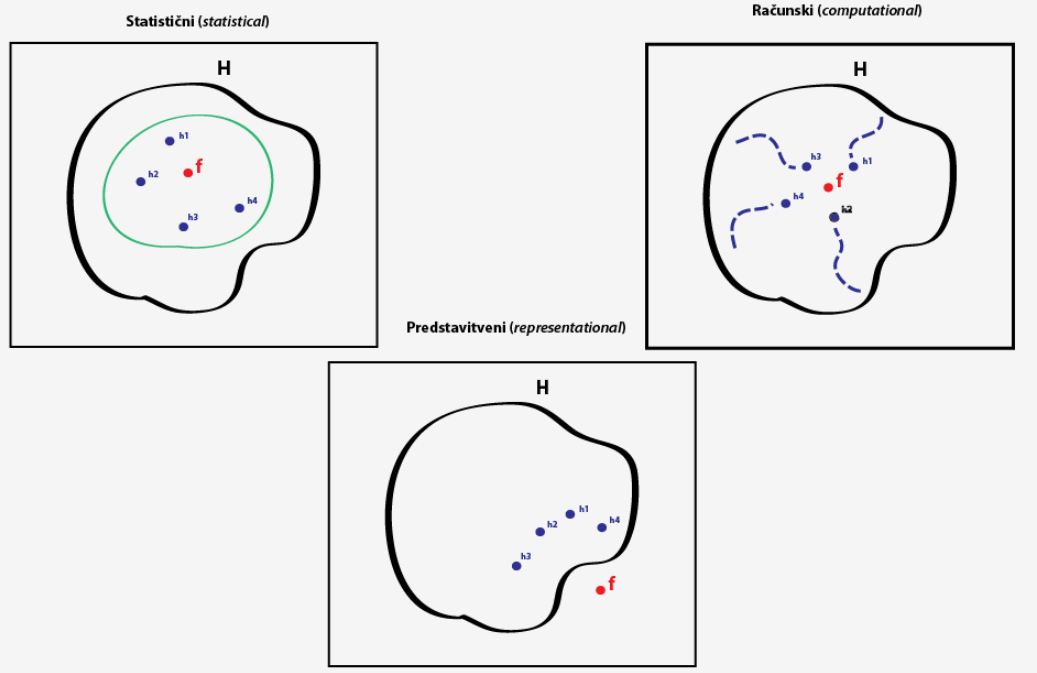
Drugi razlog je računski. Pri reševanju realnih problemov je običajno nemogoče preiskati celoten prostor možnih modelov in minimizirati napako na učni množici. To je vzrok, da mnogo učnih algoritmov deluje na ta način, da išče lokalno. Posledično lahko obtičijo v lokalnem minimumu. Algoritem za gradnjo odločitvenega drevesa na primer pri dodajanju novih vozlišč (iskanju najboljšega atributa) izvaja požrešno iskanje, saj je ekstenzivno preiskovanje prostora po naravi eksponenten NP-poln problem. S taktiko požrešnega iskanja pa brez dvoma tvegamo, da bomo na določeni točki morda izbrali atribut, ki je najboljši le lokalno in ne globalno. Nezaželen stranski učinek te napačne izbire je, da se napaka v nadaljevanju širi po celotni strukturi drevesa. V primeru, da imamo namesto enega klasifikatorja združbo večih odločitvenih dreves, ki svoje iskanje začnejo iz drugačne točke (izberejo denimo drugačen korenski atribut), lahko preiščemo večji del prostora možnih rešitev in lažje aproksimiramo vrednost iskane hipoteze f [10]. Koncept računskega razloga je predstavljen na Sliki 3.2 desno zgoraj.

Tretji izmed razlogov, ki jih navaja Dietterich [10], je predstavitveni (*representional*) razlog. Predstavitveni razlog se navezuje na v prejšnjem delu predstavljen problem pristranskosti (*bias*) pri iskanju optimalne funkcije F. V primeru pristranskosti je naša težava, da imamo pomankljivo orodje, saj iščemo dobro linearno ciljno funkcijo v X, medtem ko je ciljna funkcija v resnici kvadratna. Posledično ima učni algoritem veliko pristranskost, saj prostor možnih hipotez ne vsebuje pravilne optimalne rešitve. Z orodjem, ki

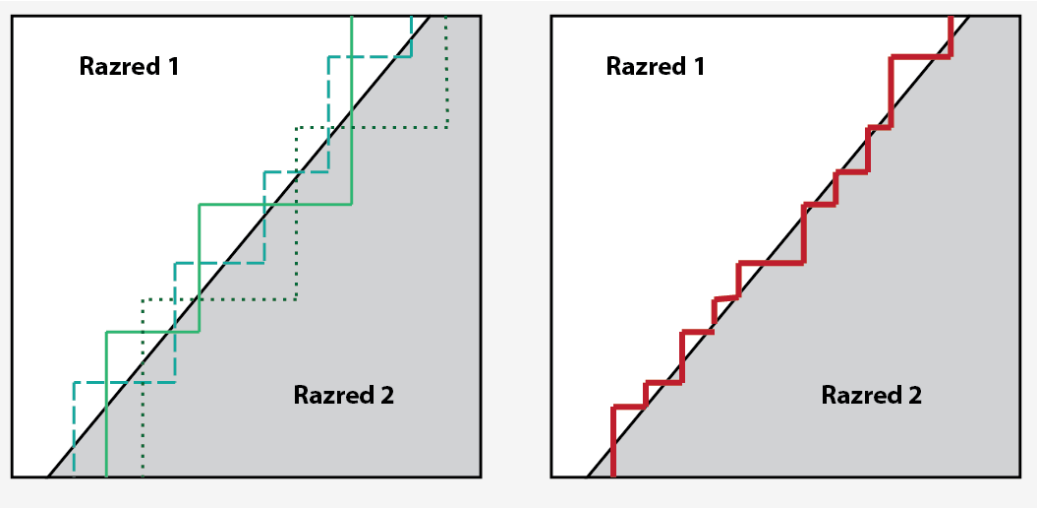
ga imamo na razpolago, torej nikakor ne moremo ustrezno opraviti zadane naloge, saj z eno samo črto, ne glede na to, kako jo postavimo, ne moremo narisati krivulje. Izkaže se, da problem lahko ublažimo s tem, da uporabimo združbo manj izraznih modelov, ki vsi iščejo linearno rešitev. Na ta način lažje aproksimiramo ciljno funkcijo, ki je po naravi kvadratna in se tudi bolje prilegamo učni množici [32]. Narišemo torej več črt, s katerimi vsaj približno ujamemo nekatere lastnosti krivulje in na ta način zmanjšamo napako na učni množici. To lahko vodi v zmanjševanje pristranskosti (*bias*), saj nam uporaba večjega števila istovrstnih funkcij omogoča, da razširimo prostor funkcij, ki jih lahko vsaj približno zajamemo z določenim algoritmom. V primeru odločitvenih dreves ta problem lahko opazujemo v situacijah, ko ciljna odločitvena meja ni ortogonalna v prostoru. Odločitvena drevesa so namreč omejena v particioniranju prostora vhodnih spremenljivk in ga lahko razdelijo le v pravokotne dele. Slika 3.3 ilustrira sposobnost združevanja večjega števila dreves, ki vodi v boljšo aproksimacijo ciljne diagonalne linearne funkcije, ki predstavlja resnično mejo za razločevanje med dvema razredoma.

Konvencionalna modrost v smislu Occamove britve opozarja na to, da je nesmiselno narediti z več, kar lahko storimo z manj. V kontekstu strojnega učenja to načelo posplošimo na t.i princip najkrajšega opisa [29]. Princip najkrajšega opisa navadno razumemo kot stališče, da imajo preprostejši modeli, zgrajeni na velikem številu učnih primerov, boljšo napovedno sposobnost v primerjavi z preveč kompliciranimi modeli. Preprostejši modeli imajo torej sposobnost, da se v večji meri približajo natančnosti, ki jo dosegajo na učni množici, tudi ko so soočeni z novimi primeri iz testne množice. Princip Occamove britve pa si lahko morda lahko razlagamo tudi kot željo po majhni varianci. Slednje velja za »običajne« klasifikacijske modele oziroma za iskanje ene najboljše hipoteze.

Pri skupinskih modelih pride v poštev še drug princip, ki mu pravimo princip večkratne razlage. Že dolgo pred pojavom področja strojnega učenja,



Slika 3.2: Trije morebitni razlogi (statistični, računski, predstavitveni) za večji uspeh skupinskih modelov v primerjavi z uporabo enega samega modela [10].



Slika 3.3: Iskana linearna odločitvena meja in odločitvena meja združenih odločitvenih dreves. Slika na levi prikazuje posamezne odločitvene meje treh različnih odločitvenih dreves. Slika na desni ilustrira odločitveno mejo, ki jo dobimo z vključitvijo teh treh dreves v skupinski model, ki glasuje za končno skupno odločitev [10].

je starogrški filozof Epikurej to načelo opisal z navodilom, da je v primeru, ko smo soočeni z večjim številom z dejstvi konsistentnih teorij, najboljše da obdržimo vse. Princip večkratne razlage od nas zahteva uporabo čim večjega števila hipotez in je navidezno v konfliktu z minimalističnim principom Ocaamove britve (princip najkrajšega opisa), ki narekuje, naj ne kompliciramo preveč in da je več manj. Vendar se izkaže, da je to nasprotje morda zgolj navidezno. Če oba principa postavimo v pravo luč, se lahko medsebojno dopolnjujeta. Princip Occamove britve uporabimo za iskanje najboljše hipoteze in z njim eliminiramo nezanesljive osnovne hipoteze. Princip večkratne razlage uporabimo v naslednji fazi za kombinirananje več najboljših osnovnih hipotez. Na ta način lahko uporabimo vse možne hipoteze [29].

Princip večkratne razlage spominja na centralni limitni teorem, kjer z večanjem števila spremenljivk povprečje konvergira k pravi verjetnosti [29]. V Principu večkratne razlage lahko morda vidimo tudi, da gre na nek način

za izraz istega načela moči številčnosti kot v primeru prej omenjenega in bolj podrobno predstavljenega *Condorcet's Jury theorema* [32]. Tudi v članku Breimana [4] lahko najdemo izraženo podobno stališče. Breiman je v teoretični utemeljitvi natančnosti skupinske metode naključnih gozdov (*random forest*) pokazal, da metoda konvergira in da pretirano prileganje učni množici (*overfitting*) ni problem, kar gre pripisati zakonu velikih števil (*Strong law of large numbers*).

V primeru združevanja osnovnih klasifikacijskih modelov bi morda lahko potegnili vzporednico tudi med principom Occamove britve in majhne variance ter vzporednico med načelom večkratne razlage in željo po majhni pristranskosti (*bias*). Če torej vidimo prej omenjena principa Occamove britve in večkratne razlage kot medsebojno dopolnjujoča, potem jih morda lahko vidimo kot možen način, kako hkrati zmanjšati pristranskost in varianco. Slednje po nekaterih razlagah velja za enega izmed pomembnih mehanizmov, ki lahko pojasni uspeh skupinskih modelov. Druga teorija, ki ima intuitivno mnogo vzporednic s prej omenjeno zmožnostjo doseganja obeh ciljev sočasno, je teorija stohastične diskriminacije (*stohastic discrimination*), ki je podlaga za nastanek skupinskega modela naključnih podprostorov (*random subspace*) [26] in jo bomo bolj podrobno predstavili v nadaljevanju (podpoglavje: 3.2.4). Pod določenimi pogoji v kontekstu stohastične diskriminacije konflikt ne obstaja. Celo več, skupinski modeli v nasprotju s pričakovanji s povečevanjem kompleksnosti še povečujejo natančnost na testni množici, tudi potem ko dosegli 100 odstotno točnost na učni množici.

Glede na do sedaj povedano lahko vidimo, da je področje skupinskih metod precej pestro in teoretično razvejano. Obstaja mnogo smiselnih razlag, med katerimi lahko morda začutimo kar nekaj podobnosti, vendar pa še vedno ni neke krovne skupne enotne teorije. Morda je ravno zato nastalo toliko različnih odgovorov na vprašanje, zakaj združevanje pogosto deluje bolje kot uporaba posameznih modelov. Odgovor na to vprašanje je ponavadi posledica razvoja različnih skupinskih metod. Če bi morda povzeli vse skupaj,

bi lahko rekli, da imamo pravzaprav več različnih teorij, ki so podlaga za različne modele skupinskega učenja, pod črto pa razlagajo morda podobno temeljno zakonitost, vendar z različno terminologijo. Diettrich [10] ugotavlja (oziroma namiguje), da skupinske metode pozitivno vplivajo na odpravljanje konfilkta pristranskosti in variance, kar on poimenuje kot zmožnost skupinskih metod, da hkrati naslavljajo tako reprezentativni problemi kot tudi statistični problem. Do podobne ugotovitve o potencialu združevanja, pride tudi Kleinberg [26] v svojem članku o skupinskem modelu, ki s povečevanjem kompleksnosti, ne izgublja sposobnosti generalizicje in v nasprotju s pričakovanji povečuje svojo uspešnost. Princip dopolnjevanja načela Occamove britve in večkratne razlage tudi lahko morda vidimo kot način ublažitve konflikta med pristranskostjo in varianco. Vendar pa je še vedno možno med vrsticami razbrati, kot bomo videli v nadaljevanju, da ta inherentna konfliktnost morda še vedno ostaja prisotna, saj določene metode združevanja v večji meri naslavljajo en aspekt konflikta, druge pa se osredotočajo na drugega [11]. Slednje lahko morda pojasni razlike v njihovi uspešnosti pri uporabi za reševanje praktičnih problemov na različnih domenah.

3.2 Metode za gradnjo skupinskih modelov

Cilj in hkrati tudi potreben pogoj za uspeh, h kateremu stremimo pri izgradnji skupinskih modelov, je poleg čim večjega nivoja natančnosti tudi doseganje čim večje različnosti osnovnih klasifikatorjev. Želeli bi torej, da nam osnovni klasifikatorji dajejo čim bolj različne odgovore. Želja po čim večji natančnosti in hkrati zagotavljanje čim večje raznovrstnosti, sta na žalost nekoliko protislovna cilja, saj obstaja tendenca, da so si natančni klasifikatorji, navadno med seboj tudi bolj podobni. Zaradi pomembnosti različnosti osnovnih klasifikatorjev, se je razvilo nekaj različnih pristopov zagotavljanja diverzitete, od katerih bomo v nadaljevanju opredeli nekaj najbolj priljubljenih možnosti.

Raznovrstnost zgrajenih klasifikatorjev lahko dosežemo s pomočjo manipuliranja z učnimi primeri, ki so vhod v naš učni algoritem. Ena izmed možnosti, ki jih imamo na voljo, je, da iz celotne učne množice izbiramo različne podmnožice učnih primerov. Na ta način uporabimo manj učnih primerov, kot jih imamo na voljo. V to kategorijo sodi metoda bagging [3] in zelo znana nadgradnja naključnih dreves (*random forest*) [4]. Poleg tega lahko manipuliramo s prostorom vhodnih spremenljivk. V okviru te metode učni algoritem gradimo na različnih podmnožicah atributov t.j naključnih podprostorih vhodnih spremenljivk (*random Subspces*) [20]. Z množico učnih primerov lahko manipuliramo tudi na nivoju uteževanja primerov iz učne množice in jim med postopkom učenja spreminjamo uteži, kar je temelj metode boosting [35]. Poleg iskanja raznovrstnosti z različnimi prijemi na učni množici imamo na voljo tudi možnost vpeljevanja elementov naključnosti v proces gradnje osnovnih klasifikatorjev. Slednje se med drugim denimo uporablja kot del procesa gradnje posameznih dreves v naključnem gozdu. Znan postopek za gradnjo skupinskih modelov je tudi pristop združevanja osnovnih modelov, ki so zgrajeni z uporabo različnih algoritmov na osnovnem nivoju in združevanje teh osnovnih modelov z uporabo klasifikatorja na meta nivoju. V to kategorijo sodi skladanje klasifikatorjev (*stacking*) [41].

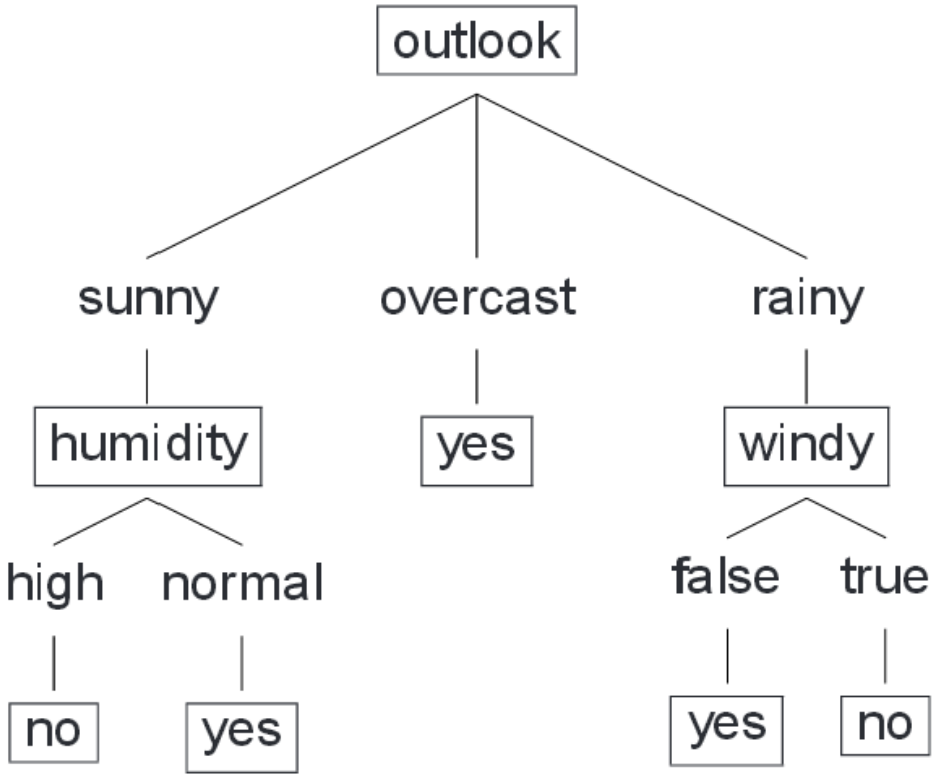
Omenjene metode bomo na kratko predstavili v nadaljevanju tega poglavja. Najprej pa bomo uvodoma predstavili še dva algoritma, ki se pogosto uporabljata v postopku gradnje osnovnih klasifikatorjev. To sta odločitveno drevo in odločitvena pravila. Razlog za pogosto uporabo odločitvenih dreves v skupinskih metodah lahko morda najdemo v enem izmed začetnih del na temo skupinskih metod. Breiman [3] namreč ugotavlja, da je eden izmed pogojev za dobro delovanje skupinskih metod nestabilnost osnovnih algoritmov. Nestabilnost pomeni, da majhne spremembe v učni množici pomenijo velike spremembe v strukturi algoritma. Dva izmed algoritmov s to običajno nezaželeno lastnostjo, vendar v kontekstu združevanja ključno prednostjo, sta odločitvena drevesa in odločitvena pravila. Zahteva po nestabilnosti mo-

rebiti ne velja za vse metode v enaki meri. Morda bi lahko rekli, da je nestabilnost načeloma bolj pomembna za uspeh metod, ki manipulirajo z učnimi primeri in vhodnimi spremenljivkami, saj na primer boosting niti ne postavlja te zahteve. Eden izmed razlogov, zakaj je temu tako, morda tiči v opažanju Diettericha, ki ugotavlja, da boosting in bagging pravzaprav naslavljata različna problema. Bagging v večji meri naslavlja statistični problem, boosting pa predstavitvenega (reprezentacijskega) [10].

3.2.1 Odločitvena drevesa

Odločitvena drevesa (*decision tree*) so eden izmed najbolj priljubljenih in enostavnih klasifikacijskih modelov, saj so relativno enostavna za razumevanje ter implementacijo, imajo veliko izrazno moč in jih je enostavno interpretirati. Odločitveno drevo je hierarhična struktura, kjer notranja vozlišča ustrezajo atributom, posamezne veje vrednostim atributov, listi pa določajo razred. Glede na pogoje, ki jih določajo vrednosti atributov, se pomikamo po drevesu od korena do listov in na ta način dobimo odločitveno pravilo, katerega listi klasificirajo primer v ustrezen razredu [29]. Primer odločitvenega drevesa je prikazan na Sliki 3.4.

Gradnja odločitvenega drevesa se prične s korenskim vozliščem, kjer za primere iz učne množice, poiščemo najboljši atribut. Glede na vrednost tega atributa učne primere razdelimo na manjše podskupine. Cilj gradnje odločitvenega drevesa je zgraditi čim manjše drevo. Ta problem je eksponentne narave, zato uporabimo hevristično rešitev in v vsakem koraku izberemo najboljši atribut. Temu postopku pravimo požrešno iskanje (*best-first search*, *greedy search*). Njegova temeljna hiba je, da kratkovidno izberemo atribut, ki se nam v določenem trenutku zdi najboljša kratkoročna možnost. Seveda izbrani atribut dolgoročno gledano ni nujno najbolj optimalna izbira. Za določitev najboljšega atributa, na podlagi katerega bomo glede na vrednost



Slika 3.4: Primer odločitvenega drevesa (Vir: `http://www.coli.uni-saarland.de/~crocker/Teaching/Connectionist/lecture9_4up.pdf`).

tega atributa razdelili učne primere, se uporabljajo različne ocene med katerimi je na primer informacijski prispevek (*information gain*). Informacijski prispevek meri, koliko informacije o razredu nosi posamezen atribut. Za vsako vrednost najboljšega atributa nato rekurzivno ponavljamo postopek dodajanja novih vozlišč. Izbira ocene za določitev najboljšega atributa je lahko zelo pomembna, saj z izbiro atributov v vozliščih močno vplivamo na strukturo drevesa [32].

Pomembno vprašanje pri gradnji dreves pa je, poleg izbire metode za določitev najpomembnejšega atributa, tudi vprašanje, kdaj ustaviti gradnjo drevesa. Slednje je morda eden izmed ključnih elementov za uspešnost odločitvenih dreves, saj z dodajanjem vozlišč in posledičnim hitrim zmanjševanjem števila učnih primerov v posameznih vejah drevo postaja čedalje bolj občutljivo na šum v podatkih. Povečuje se tudi nevarnost, da se pretirano prilagodimo množici učnih primerov (*overfitting*). Z majhnim številom primerov v vozliščih se namreč zelo hitro povečuje varianca, ki degradira performanse drevesa [32]. Za rešitev te težave so se razvile metode za ustavljanje gradnje drevesa in naknadno rezanje drevesa, ki poskušajo iz drevesne strukture odstraniti nezanesljiva vozlišča. Na področju skupinskih metod so odločitvena drevesa ravno po zaslugi nestabilnosti strukture, ki je posledica požrešnega izbiranja najboljšega atributa v posameznih vozliščih, pogosto uporabljene kot osnovni algoritem za gradnjo skupinskega modela.

3.2.2 Odločitvena pravila

Druga metoda za pridobivanje novega znanja o množici učnih primerov je gradnja odločitvenih pravil. Tudi odločitvena pravila so znana po svoji nestabilnosti in jih prav tako lahko uporabimo kot osnovni klasifikacijski algoritem v okviru skupinskih metod. Poleg tega imajo kar nekaj vzporednic z odločitvenimi drevesi. S plezanjem po strukturi drevesa, od korena

vse do listov, lahko dobimo odločitveno pravilo. Obratno pa ne drži, saj iz množice odločitvenih pravil, navadno ne moremo zgraditi drevesa. Nadzorovano učenje odločitvenih pravil je torej usmerjeno k iskanju množice pravil, ki jih lahko uporabljamo za klasifikacijo novih primerov. Temu pristopu pravimo tudi napovedna indukcija (*predictive induction*) [25]. Algoritem za generiranje učnih pravil iz množice učnih primerov generira pravila oblike

$$IF < Pogoji > THEN razred = c_i$$

Konjunkcijo pravil imenujemo tudi selektor. Selektor vsebuje enega ali več pogojev nad atributi oblike $A_i = v_{ij}$ za diskretne in $A_i < v$ ali $A_i > v$ za zvezne atribute. Sklepni del pravila novemu primeru določi vrednost razreda [14]. Tipičen predstavnik algoritma za gradnjo odločitvenih pravil je CN2 [6, 5], ki smo ga implementirali v procesu izdelave naše diplomske naloge. Delovanje tega algoritma in njegovo implementacijo bomo bolj podrobno predstavili v podpoglavju 5.6.

3.2.3 Izbiranje učnih primerov: bagging in naključni gozdovi

Ena izmed najbolj priljubljenih metod za gradnjo med seboj karseda različnih osnovnih klasifikatorjev je izbiranje različnih podmnožic učnih primerov. Ta strategija se včasih imenuje tudi razmnoževanje učnih primerov [29]. Na teh različnih podmnožicah učnih primerov zgradimo osnovne klasifikacijske modele in jih na koncu sestavimo v končni skupinski model. Tehnika razmnoževanja učnih primerov še posebej poudarja pomembnost nestabilnosti osnovnih klasifikatorjev. Nestabilni klasifikatorji, ki so odzivni na spremembe podatkov v učni množici, nam omogočajo, da na izbranih manjših podmnožicah učnih primerov, zgradimo klasifikatorje, ki so med seboj dovolj različni [10]. Značilna predstavnika te kategorije skupinskih modelov sta Breimanov bagging [3] in njegova nadgradnja v obliki naključnega gozda

(*random forest*) [4].

Bagging

Osnovna značilnost bagginga je, da v vsaki iteraciji iz učne množice velikosti m z vračanjem izberemo m primerov. Vsaka izmed na ta način dobljenih učnih množic, na katerih uporabimo učni algoritem, v poprečju vsebuje 63,3% različnih primerov iz originalne množice. Zaradi izbiranja z vračanjem se namreč kar nekaj učnih primerov v novi učni množici pojavi večkrat [3]. Dejstvo, da v povprečju približno $\frac{1}{3}$ učnih primerov nismo uporabili, poleg tega da prispeva k večji odpornosti algoritma za šum v podatkih [4], v sebi nosi še dodaten velik potencial. Te neizbrane učne primere lahko namreč vidimo kot novo testno (validacijsko) množico (*out-of-bag*), s pomočjo katere lahko med drugim na primer dobimo dokaj objektivno oceno generalizacijske napake (*out-of-bag error*), brez potrebe po uporabi postopka prečnega preverjanja na ločenih testnih množicah [4]. Poleg teh dodatnih koristi, nam uporaba strategije bagginga v prvi vrsti omogoča, da dobimo med seboj različne učne množice, na katerih lahko z nestabilnimi algoritmi zgradimo različne osnovne klasifikatorje.

Naključni gozdovi

Dobro poznana metoda združevanja osnovnih modelov so naključni gozdovi (*random forest*) [4]. Breimanovo idejo naključnega gozda lahko morda vidimo kot nadgradnjo ideje bagginga [3]. Praktična implementacija je naključni gozd *RF-1*, ki deluje na ta način, da za gradnjo vsakega posameznega drevesa uporabi z baggingom dobljeno podmnožico učnih primerov. Na tako dobljenih podmnožicah zgradi raznovrstna odločitvena drevesa. Druga pomembna lastnost naključnega gozda je sama tehnika gradnje posameznega drevesa. V gradnjo posameznih dreves vpeljemo nekaj elementov naključnosti, saj za razliko od običajnih odločitvenih dreves ne izbiramo enega samega najboljšega atributa. Pri dodajanju posameznih vozlič v strukturo

drevesa algoritem naključno izbere manjšo podmnožico atributov, med katerimi nato izbere najboljšega [4]. Posledično bi lahko rekli, da ima ta naključnost izbire podmnožice atributov v vozliščih nekaj vzporednic s teorijo stohastične diskriminacije (*stohastic discrimination*) [26] in njeno praktično implementacijo naključnih podprostorov (*random subspace*) [19], ki ju bomo predstavili v nadaljevanju. Ta pristop h gradnji skupinskih modelov je bil namreč prav tako razvit v približno istem času kot Breimanova ideja bagginga in ravno tako deluje na nivoju randomizacije prostora vhodnih spremenljivk.

3.2.4 Izbiranje vhodnih spremenljivk: naključni podprostori

Kleinbergova teorija stohastične diskriminacije [26] (*stohastic discrimination*) ponuja zelo splošno teoretično ogrodje, kako zasnovati proces gradnje zbirnega klasifikatorja, katerega skupna natančnost narašča s številom posameznih šibkih modelov (*weak models*), vključenih v združbo. Šibki modeli so v kontekstu stohastične diskriminacije široko definirani kot podmnožice vhodnih spremenljivk (*subset of feature space*). To pomeni, da takšen šibek model morda ni sposben povedati ničesar o primerih (oziroma delu prostora), ki jih ne pokriva. Posamezen šibek model si morda lahko predstavljamo kot odločitveno pravilo oblike $IF < Pogoj > THEN < Razred >$. Tak model (oz. odločitveno pravilo) nam ne more dati nobene informacije o pripadnosti razredu posameznega primera, ki ga pogoj pravila ne pokriva.

Teoretični pogoji, ki jih narekuje teorija stohastične diskriminacije za uspeh takega združenega klasifikatorja, so obogatitev (*enrichment*), uniformnost (*uniformity*) in projektabilnost (*projectability*). Obogatitev zahteva, da je vsak šibek model sposoben zajeti več primerov iz enega razreda kot iz drugega. Vsak naključen podprostor mora torej zajeti večje število pripadnikov iz enega razreda. Potrebno je torej zagotoviti, da je posamezen šibek

model uspešnejši od naključnega ugibanja. Naslednji pogoj je uniformnost. Uniformnost zahteva, da so vsi učni primeri pokriti enakomerno. Slednje lahko razumemo kot zahtevo, da je vsak primer iz učne množice, pokrit s približno enakim številom šibkih modelov. V primeru, da smo torej sposobni 1) generirati šibke modele naključno, 2) posamezni šibki modeli se v razločevanjem med razredoma odrežejo boljše kot ugibanje, 3) in šibki modeli uniformno pokrivajo prostor učnih primerov, potem bo naš model projektabilen in bo imel na populaciji enako sposobnost razločevanja med razredi kot na učni množici. Kleinberg in kasneje Ho, sta se zelo promovirala to teorijo in v kar nekaj člankih lahko najdemo dokaj preproste ilustracije teh teoretičnih izhodišč skozi poenostavljene matematične primere [27, 21].

Idejo gradnje velikega števila odločitvenih dreves na naključno izbranih vhodnih spremenljivkah (atributih) je Ho [19] prvič predlagala v delu na temo naključnih odločitvenih gozdov (*random decision forests*) in jo kasneje nadgradila v bolj znano metodo naključnih podprostorov (*random subspace*) [20]. Metoda naključnih podprostorov pogosto velja za eno izmed praktičnih implementacij teorije stohastične diskriminacije. Uspešnost metode združevanja odločitvenih dreves, je opredeljena kot 100 odstotna pravilnost na učni množici in padanje generalizacijske napake na testni množici z dodajanjem posameznih dreves. Ključna karakteristika metode je, da zagotavlja različnost dreves v gozdu s projekcijo učnih primerov v zmanjšan prostor vhodnih spremenljivk in z gradnjo posameznih neporezanih dreves z čistimi listi, ki zagotovijo obogatitev na tem zmanjšanem podprostoru atributov. Za vsak prostor binarnih vhodnih spremenljivk velikosti n imamo tako na voljo 2^n možnih podprostorov [20]. Na vsakem izmed njih lahko zgradimo drugačno odločitveno drevo. To nam nudi več možnosti, kot jih pri velikem številu atributov v določenih domenah potrebujemo v praksi in omogoča, da prekletstvo dimenzionalnosti spremenimo v blagoslov in uporabimo v svojo korist.

Random subspace in random forest (brez bagginga) lahko morda vidimo

kot implementacijo teorije stohastične diskriminacije. Obe metodi skupinskih modelov gradita odločitvena drevesa z randomizacijo na nivoju izbiranja atributov v vozliščih. Za obe metodi bi lahko rekli, da poiščeta podprostor vhodnih spremenljivk, na katerem gradita osnovne šibke modele. Osnovni šibki modeli so neporezana odločitvena drevesa, ki zagotavljajo obogatitev, saj so listi drevesa čisti t.j. vsebujejo le primere iz enega razreda. Poleg tega gradnja odločitvenih dreves zagotavlja uniformnost, saj odločitveno drevo enakomerno pokrije vse učne primere. Problem je le generalizacijska sposobnost oziroma projektabilnost posameznega drevesa. Sposobnost generalizacije posameznega lista v odločitvenem drevesu je namreč odvisna od števila primerov v listih. Nesposobnost generalizacije listov lahko vodi v veliko razliko v obogatitvi (*enrichment*), ki jo dosežemo na učni množici in tisto, ki jo bomo dosegli na testni množici. Obstaja torej kompromis med projektabilnostjo (sposobnostjo generalizacije) in številom modelov. Če dovolimo visoko stopnjo randomizacije, lahko ustvarimo veliko število različnih osnovnih modelov. Po drugi strani pa je tako zelo naključno odločitveno drevo, ki ga dobimo s pomočjo randomizacije, lahko zelo šibko obogateno, saj vsak list v takih modelih pokriva malo primerov in posledično izgubi sposobnost generalizacije. Nasprotno pa z generiranjem plitkih (naknadno porezanih) in močnih dreves z listi, ki vsebujejo mnogo primerov, drevesa postanejo preveč podobna [32].

3.2.5 Uteževanje učnih primerov: boosting

Ideja boostinga je transformirati oziroma ojačati (*boost*) slabo učno strategijo, ki ima uspešnost le rahlo večjo od naključnega ugibanja, v uspešen učni algoritem. Zasnovati močno učno metodo, ki zanesljivo daje dobre rezultate, je namreč bistveno težje, kot zasnovati metodo, za katero zahtevamo le to, da se mora odrezati vsaj rahlo boljše, kot če bi naključno napovedovala izzid [32]. Odgovor na vprašanje, kako to storiti, je ponudil Schapire in

kmalu zatem je prišlo do prve konkretne implementacije boostinga *AdaBoost* (*Adaptive boosting*) [16].

Temeljna ideja algoritma *AdaBoost* je v iterativnem uteževanju primerov iz učne množice glede na njihovo težavnost. V prvi iteraciji imajo vsi primeri enako utež ($\frac{1}{n}$) in osnovni klasifikator jih klasificira. V vsaki naslednji iteraciji zopet uporabimo enak šibek učni algoritem, vendar na drugače utežni množici učnih primerov. Uteži določimo glede na pomembnost posameznega primera v trenutni iteraciji. Primerom, ki smo jih v prejšnji iteraciji klasificirali pravilno, zmanjšamo utež. Primerom, za katere smo se zmotili, utež povečamo. Temeljna ideja je torej, da se algoritem s pomočjo zmanjševanja in povečevanja uteži v vsaki iteraciji posledično osredotoča na težje primere, t.j. primere, ki jih v prejšnji iteraciji ni klasificiral pravilno [15].

Zanimiva karakteristika metode boostinga je lastnost, da lahko s povečevanjem števila iteracij, zmanjšujemo napako na testni množici, še dolgo potem, ko smo dosegli nično napako na učni [32]. Tovrstno povečevanje kompleksnosti, ki je rezultat povečevanje števila iteracij, navadno vodi v pretirano prileganje učni množici (*overfitting*), na katerega pa je boosting teoretično relativno imun. Celo nasprotno. Napaka na učni množici po določenem številu iteracij, pade na nič. S povečevanjem števila iteracij, na testni množici napaka nadaljuje padanje, kljub temu, da je model na učni množici dosegel nično napako.

Na prvi pogled ta nenavadna lastnost izgleda kot lep primer paradoksa. Schapire to neobičajno obnašanje boostinga empirično razišče z izvedbo praktičnih poskusov na različnih podatkovnih zbirkah. Kot algoritem za gradnjo osnovnih modelov uporabi odločitveno drevo. Napaka na učni množici po parih iteracijah boostinga hitro pade na nič, kar pomeni, da vsi primeri pravilno klasificirani. Napaka na testni množici je v tem konkretnem trenutku 13.8%. Na tej točki ni ovire, ki bi nas ustavila, da ne bi mogli več dodajati novih odločitvenih dreves. S povečevanjem števila iteracij (dodaj-

nem dreves), napaka na testni množici nadaljuje svoje padanje in pri 1000 iteracijah znaša le še 3.1% [35]. Zastavi se povsem legitimno vprašanje, kako je možno, da se tak prekomerno kompleksen klasifikator z ogromnim številom vozlišč, ki ga lahko morda vidimo kot utelešenje pojma pretiranega prileganja (*overfitting*), na testni množici obnaša toliko boljše. Schapire ponudi pojasnilo za ta fenomen s pomočjo razlage o stopnji zaupanja (*margins explanation*) [35]. Napaka na učni množici je le del cele zgodbe. Napaka na učni množici nam pove samo število pravilno klasificiranih primerov. Ne pove pa nam ničesar o tem, kako prepričan je model v svoje odločitve. Zaupanje v odločitev je ta t.i. *margin*. Napaka na učni množici v trenutku, ko smo vse primere pravilno klasificirali, doseže nič in se od tega trenutka naprej s povečevanjem števila iteracij ne more spreminjati. Spreminja pa se lahko zaupanje modela v svoje odločitve. S povečevanjem števila vključenih osnovnih modelov, lahko povečujemo nivo zaupanja (*margin*) in posledično zmanjšujemo generalizacijsko napako na testni množici. Slednja lastnost je tudi eden izmed temeljnih kamnov prej omenjene teorije stohastične diskriminacije. Tudi stohatistična diskriminacija namreč identificira to presenetljivo lastnost šibkih modelov.

Po drugi strani nekatere raziskave kažejo, da pri reševanju praktičnih problemov lahko naletimo na problem pretiranega prileganja (*overfitting*), še posebno v primerih, ko je v naši učni množici prisotnega veliko šuma. V primeru, da imamo v učni množici veliko šuma, potem lahko boosting z iterativnim uteževanjem učne množice, povečuje uteži ravno tistim primerom, ki smo jih je zaradi šuma napačno klasificirali v prejšnjih iteracijah [11]. Povečevanje uteži šumnim primerom lahko vodi v pretirano prileganje. Pretirano prileganje je v tem kontekstu nezaželeno, saj smo z uteževanjem v združeni model zajeli preveč šuma. Po drugi strani je ta lastnost lahko koristna v primeru, da želimo identificirati osamelce (*outlinerje*) t.j. primere, ki na nek način odstopajo. Poleg določenih pasti, na katere lahko naletimo pri uporabi boostinga, je boosting priljubljena metoda združevanja klasifika-

torjev, znana prav zaradi svoje uspešnosti. Praktično ima namreč AdaBoost kar nekaj prednosti. Algoritem je relativno lahko implementirati, ne zahteva nastavljanja številnih parametrov in ga lahko kombiniramo s poljubnim osnovnim algoritmom [35].

3.2.6 Skladanje klasifikatorjev: stacking

Do sedaj predstavljeni skupinski klasifikacijski modeli na osnovnem nivoju gradijo posamezne klasifikatorje z enim samim učnim algoritmom. Priljubljena izbira algoritma so odločitvena drevesa. Odločitve posameznih osnovnih klasifikatorjev nato tipično združijo z večinskim ali utežnim glasovanjem. Različnost osnovnih klasifikatorjev poskušajo zagotoviti s prijemi, kot so manipuliranje z učno množico, prostorom vhodnih spremenljivk in vpeljevanjem elementov naključnosti.

Za razliko od do sedaj opisanih skupinskih metod, ki se osredotočajo na gradnjo osnovnih klasifikatorjev z enakim algoritmom, obstaja še drug pristop, ki temelji na uporabi in kombiniranju različnih učnih algoritmov na osnovnem nivoju. Skladanje klasifikatorjev (*stacking*) [41] sodi v to kategorijo skupinskih algoritmov, ki se poskušajo naučiti, kako uspešno združiti heterogene nizkonivojske klasifikacijske algoritme [13]. Na skladanje klasifikatorjev lahko v širšem kontekstu gledamo kot na meta-učenje. Meta-učenje je učenje o učenju samem, kjer kot vhod vzamemo izhod prve stopnje učenja in ga poskušamo generalizirati na višjem nivoju. Ena izmed nalog meta učenja, je učenje o tem, kako uspešno združiti napovedi heterogenih osnovnih klasifikatorjev. Za pridobivanje napovedi osnovnih algoritmov v procesu gradnje modela, izvajamo interno prečno preverjanje na učni množici. Učno množico torej razdelimo na dve novi podmnožici. Prva vsebuje učne, druga testne primere. Napovedi osnovnih klasifikatorjev za posamezni testni primer iz te sekundarne testne množice, skupaj s pravilnim razredom tega testnega

primera, predstavljajo vhod v klasifikator na meta nivoju [13].

Formalno lahko skladanje klasifikatorjev definiramo kot uporabo različnih učnih algoritmov $(L_1, \ldots, L_N)$ na isti učni množici S, kjer je vsak primer iz učne množice definiran kot $(s_i = (x_i, y_i))$. Vektor atributov učnega primera je x_i, pripadajoči razred pa je y_i. V prvi fazi zgradimo množico osnovnih klasifikatorjev $(C_1, C_2, \ldots, C_N)$, kjer je $C_i = L_i(S)$. V drugi fazi pa zgradimo meta-klasifikator, ki kot vhod v učni algoritem (nove učne primere), uporabi izhod klasifikatorjev na prvem nivoju [40]. V procesu izdelave diplomskega dela smo implementirali to metodo združevanja klasifikatorjev, zato bomo postopek delovanja te metode, bolj podrobno opisali in natančno predstavili v drugem delu diplomskega dela (podpoglavje: 5.8).

3.2.7 Kratek povzetek poglavja

Področje skupinskih algoritmov v strojnem učenju je po zaslugi doseganja dobrih praktičnih rezultatov deležno velikega zanimanja. Posledica tega je razvoj različnih metod. Posamezne metode navadno gradijo svojo utemeljitev na različnih teoretičnih temeljih in predpisujejo različne predpogoje za uspeh svojega delovanja. Kljub temu, da med njimi lahko začutimo določene podobnosti, se pogosto zdi, da do sedaj še vedno ni razvite neke enotne teorije skupinskih algoritmov. Nekatere podobnosti smo poskušali identificirati in izpostaviti v tem poglavju. Poleg najbolj popularnih metod, kot so bagging, naključni gozdovi, naključni podprostori, boosting in skladanje klasifikatorjev, ki smo jih na kratko predstavili, seveda obstaja še kar nekaj drugih skupinskih metod. Med njimi lahko najdemo na primer Bayesovsko povprečenje (*Bayesian averaging*), manipuliranje izhodnih spremenljivk (*error correcting output codes*), vbrizgavanje naključnosti (*random trees*) ipd. V tem diplomskem delu smo se osredotočili le na najbolj znane, saj si ne moremo privoščiti, da bi šli v prevelike podrobnosti. Tudi pri metodah, opi-

sanih v tem poglavju, je zaradi velikega zanimanja, posledično nastalo ogromno število različnih bolj ali manj (ne)uspešnih modifikacij osnovnih oblik in različnih načinov poskusov kombiniranja med njimi, vendar je zaradi velike razvejanosti področja skupinskih metod na žalost težko na kratko predstaviti celotno sliko.

Poglavje 4

Podatki mikromrež DNA

Za primerjanje uspešnosti posameznih modelov strojnega učenja in njihovih modifikacij pri reševanju praktičnih problemov zgrajene modele pogosto testiramo na različnih podatkovnih zbirkah. Naša modifikacija metode naključnih podprostorov je dokaj specifična, saj predpostavlja uporabo definiranih podskupin prostora vhodnih spremenljivk, zato smo uporabili podatke mikročipov DNA. Ti podatki namreč omogočajo uporabo smiselnih podskupin atributov, ki predstavljajo ključen sestavni del pri izdelavi diplomskega dela. V tem poglavju bomo na kratko predstavili uporabljene podatkovne zbirke in analitično orodje GSEA, ki smo ga uporabili za njihovo analizo.

4.1 Predstavitev podatkovnih zbirk mikromrež DNA

Spomnimo se zdravniške metafore z začetka, kjer smo želeli zasnovati preprost sistem za avtomatsko ugotavljanje pacientovega zdravstvenega stanja.

V ta namen smo beležili atribute za posameznega pacienta, kot so telesna temperatura, kašljanje ipd. S pojavom in napredkom genetike ter z razvojem tehnologije mikromrež DNA pa naši pacienti niso več opisani z majhnim številom dokaj razumljivih atributov. Na področju modeliranja genetskih podatkov je povsem običajno, da se število atributov meri v več tisočih. Poleg tega je brez obsežnega ekspertnega znanja pomen teh atributov težko razumljiv. Za iskanje novih zakonitosti pri obravnavanju genetskih podatkov se pogosto uporabljajo različne metode stojnega učenja in računalniško podprti postopki za pridobivanje novega znanja o človekovem genetskem ustroju. V tako velikem številu atributov je namreč z ročnim preiskovanjem brez pomoči računalnika praktično nemogoče karkoli odkriti.

Na področju raziskovanja skrivnosti človeškega telesa obstaja veliko odprtih vprašanja in na medsebojno povezanost bioloških procesov še vedno lahko gledamo kot na neke vrste črno škatlo. Z razvojem možnosti preučevanja genov smo se morda za korak bolj približali vpogledu v to črno škatlo. Gene lahko najdemo v celicah živih bitij in so osnovni nosilci dednih lastnosti. Geni so navodila za zgradbo in z medsebojno interakcijo vplivajo na procese v človekovem telesu, njegovo telesno zgradbo ipd. Organizem je torej kompleksen sistem, kjer tisoči genov in njihovih produktov (RNA in proteini) ustvarjajo čudežnost življenja. Posamezen gen si lahko predstavljamo kot besedo v slovarju človekovega genoma. Pristope, ki nam omogočajo celostno razumevanje izražanja genoma, si lahko predstavljamo kot orodja, ki nam pomagajo razumeti, kako se besede genoma povezujejo v smiselno besedilo [24].

V preteklosti so znanstveniki preučevali posamezne gene neodvisno enega od drugega, vendar pa podobno, kot ne moremo razumeti jezika, če gledamo samo s stališča posameznih besed, tudi življenje celice ni sestavljeno iz izoliranih enot, temveč iz spleta različnih procesov, ki so v nenehnem stiku z svojim okoljem. Vsaka sprememba, ki jo povzročimo v celici, tudi če ciljamo eno samo podenoto (na primer spremembo izražanja enega samega gena), se odraža v desetinah sprememb vzdolž različnih poti [24]. Za spoznavanje

tega zapletenega spleta bioloških interakcij moramo torej uporabljati celostne pristope, ki lahko sledijo več lastnostim hkrati. Temu se je preteklih letih najbolj približala tehnologija mikromrež DNA, ki je eno izmed najboljših orodij za globalne študije izražanja genoma.

V okviru izdelave diplomskega dela smo za modificiranje združevalnih skupinskih algoritmov in primerjavo njihove uspešnosti raziskavo izvajali na genetskih podatkih, pridobljenih s pomočjo uporabe tehnologije mikromrež DNA. Za raziskovanje in primerjanje različnih klasifikacijskih algoritmov smo uporabili 10 različnih podatkovnih setov (podatkovnih zbirk) genetskih podatkov. Pri tem je potrebno poudariti, da nismo želeli odgovoriti na vprašanje, kateri algoritem je najuspešnejši. Primarno nas je gnala radovednost in želja po raziskovanju, zato nismo izvajali obsežnih eksperimentov, po pravilni in sistematični metodologiji, s katerimi bi lahko dokazovali statistično pomembnost ugotovitev. Poleg tega tudi uporabljenih genetskih podatkovnih zbirk, nismo izbirali po kakšnem posebnem ključu, pač pa smo jih bolj ali manj naključno izbrali 10. Dodaten problem je, da področje bioinformatike od raziskovalcev zahteva hkratno poznavanje metod bioloških, statističnih in računalniških znanosti [24]. V našem primeru smo se s tovrstnimi podatki srečali prvič in poleg tega, da z njimi nimamo nobenih izkušenj, vsekakor nimamo dovolj potrebnega znanja, ki ga terja resnejša analiza takih podatkov.

4.2 Opis uporabljenih podatkovnih zbirk

Podatki mikromrež DNA, ki smo jih uporabili pri našem delu, so opisani z atributi, ki predstavljajo izraženost genov. V posameznimi vrstici se nahajajo podatki za posamezen učni primer. Imena posameznih atributov so po določenih standardih poimenovani geni. Razred posameznega primera je dvorazredna diskretna spremenljivka.

Data set	#Razredov	#Atributov	#Primerov	Porazdelitev
bleomycin	2	16246	28	14/14
breastColon	2	14164	52	31/21
dlbcl	2	6220	77	58/19
leukemia	2	4681	70	45/25
luminalBasal	2	14903	43	27/16
metastasis	2	14904	27	19/8
prostata	2	9581	102	52/50
responder	2	9552	46	11/35
tumorAdjacent	2	9552	39	19/20
tumorRecurrence	2	14346	59	31/28

Tabela 4.1: Osnovni podatki o uporabljenih podatkovnih zbirkah: število razredov, število različnih atributov, število primerov in porazdelitev primerov po posameznih razredih.

V diplomskem delu smo uporabili 10 različnih podatkovnih zbirk, ki vsebujejo podatke, pridobljene z uporabo tehnologije mikromrež DNA. V nadaljevanju bomo vsako izmed podatkovnih zbirk in problem, ki ga obravnava, na kratko predstavili. Glavne lastnosti posameznih zbirk so predstavljene v tabeli 4.1. Različne podatkovne zbirke vsebujejo različno število atributov in različno število učnih primerov. Skupna lastnost podatkovnih zbirk je veliko število atributov in relativno majhno število učnih primerov. Majhno število primerov je lahko posledica visokih cen meritev ali pomanjkanja primernih bioloških vzorcev [24]. Podatkovne zbirke naslavljajo različne klasifikacijske probleme, predvsem s področja rakastih obolenj, vendar jim je skupno to, da gre pri vseh za dvorazredne klasifikacijske probleme. Za posamezne zbirke lahko vidimo, da imajo različno razmerje med številom primerov, ki pripadajo posameznemu razredu. Za nekatere izmed njih tako velja, da je apriorna verjetnost enega izmed razredov zelo velika (npr: dlbcl [58/19] glej tabela 4.1).

Podatkovna zbirka **bleyomicyn** [DataSet Record GDS1714] vsebuje odzive določenih skupin celic (*mutagen-sensitive lymphoblastoid cell lines*) na zdravilo bleyomycin. Mutagenost (*mutagen sensitivity*) odraža posameznikovo dojemljivost za sporadične oblike raka (*sporadic cancers*) [7]. Raziskave mikromrež DNA so namreč nepogrešljive tudi pri preučevanju, kako dedni (genetski zapis posameznika) vpliva na odziv posameznika na določeno zdravilo, saj lahko na ta način potencialno zmanjšamo nezaželene stranske učinke [24].

Ostale podatkovne zbirke se ukvarjajo z različnimi vrstami rakastih obolenj. Podatkovna zbirka **breastColon** [DataSet Record GSE3726] vsebuje podatke o ekspresiji genov pri pacientih z dvema različnima oblikam raka in sicer raka na prsih (*breast*) ter raka na debelem črevesju (*colon*).

Podatkovna zbirka **dlbcl** [37] vsebuje podatke za klasifikacijo dveh različnih vrst limfomov. Prva vrsta je difuzni velikocelični limfom B (*diffuse large B-cell lymphoma*), druga kategorija je folikularni limfom (*follicular lymphoma*). Pogosto se folikularni limfom sčasoma razvije in postane podoben prvi vrsti limfoma (*diffuse large B-cell lymphoma*).

Podatkovna zbirka **leukemia** [18] obravnava paciente z dvema vrstama levkemije: akutna limfoblastna levkemija (ALL), ki je značilna predvsem za otroke in akutna mieloična levkemija (AML), ki se bolj pogosto pojavlja pri odraslih.

Podatkovna zbirka **luminalBasal** [DataSet Record GDS1329] vsebuje podatke o pacientkah obolelih za rakom na prsih. Rakasta obolenja dojk so kategorizirana v dva razreda tumorjev. Prvi razred je luminalni (*luminal*) drugi pa bazalni (*basal*). Tip rakaste tvorbe vpliva na izbor oblike zdravljenja. Pri zdravljenju bazalnega tipa je potrebna kemoterapija, medtem ko pri lažji obliki luminalnega tipa lahko zadošča že hormonska terapija [22].

Podatkovna zbirka **metastasis** [DataSetRecord GDS1062] vsebuje podatke o ekspresiji genov v primeru metastaze rakavih obolenj, kjer se bolezenska tvorba prenese v druge dele telesa. Pacienti so razvrščeni v dve skupini in sicer v skupino, pri kateri se je rakasto obolenje razvilo v metastatčino obliko in tiste, pri katerih se to ni zgodilo.

Podatkovna zbirka **prostata** [38] vsebuje podatke o ekspresiji genov pri pacientih z rakom na prostati in pri zdravih osebah.

Podatkovna zbirka **responder** [DataSetRecord GDS1887] vsebuje podatke o pacientih, pri katerih se je rakasto obolenje odzvalo na radioterapijo (zdravljenje z obsevanjem) in tistih, pri katerih do tega ni prišlo.

Podatkovna zbirka **tumorAdjacent** [DataSetRecord GDS1650] vsebuje podatke o ekspresiji genov v primerih tkiv pljučnega adenokarcinoma in v primerih normalnih pljučnih tkiv.

Podatkovna zbirka **tumorRecurrence** [DataSetRecord GDS2415] vsebuje podatke o tkivih pacientk, pri katerih se je rakasto obolenje na dojkah ponovilo in tistih, pri katerih se ni.

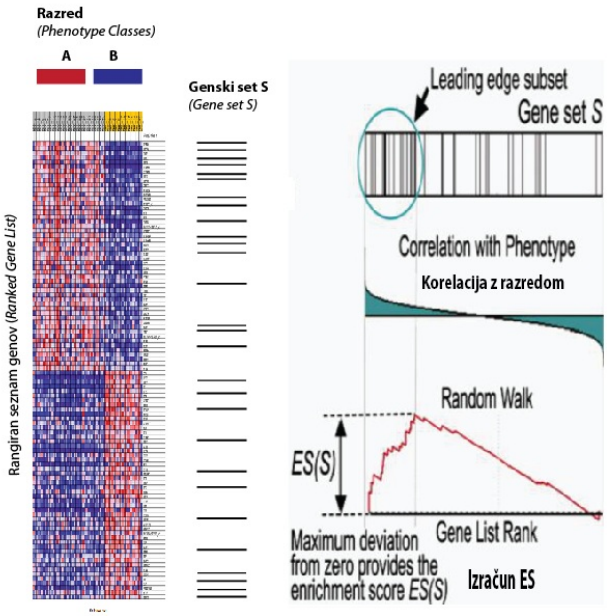
4.3 Predstavitev analize GSEA

GSEA (*Gene Set Enrichment Analysis*) (`http://www.broadinstitute.org/gsea/index.jsp`) je eno izmed analitičnih orodij za analizo in interpretacijo podatkov mikromrež DNA. Metoda se osredotoča na skupine genov (*gene sets*). Geni, ki sodijo v posamezno skupino, so med seboj na določen način povezani. Ta povezava je lahko opravljanje iste biološke funkcije, delitev lokacije v kromosomu, ali kakšna druga smiselna biološka povezava.

Gene, ki so pomembni za razločevanje med dvema različnima razredoma, lahko razvrstimo v urejen seznam genov L. Tak urejen seznam genov L odraža pomembnost posameznega gena glede na drugačno izražanje v odvisnosti od ciljnega razreda. Rangiran seznam lahko na primer dobimo tako, da uporabimo neko metriko, s pomočjo katere ocenimo sposobnost ločevanja posameznega gena med ciljnima razredoma. Prave izzive preučevanja mikromrež DNA, v večji meri kot samo razvrščanje genov po pomembnosti, predstavljata interpretacija in pomen tega seznama.

Pristopi, ki se osredotočajo na analizo nekaj genov pri vrhu in na koncu takega seznama, imajo kar nekaj pomanjkljivosti. Z osredotočanjem na posamezne najbolje rangirane gene, lahko dobimo kopico statistično pomembnih genov, brez kakšne posebne skupne biološke rdeče niti. Poleg tega lahko tovrstna analiza posameznega gena v izolaciji od širšega biološkega konteksta, spregleda pomembne učinke na celotne biološke procese, saj celični procesi navadno vplivajo na množico genov in je denimo 20% povečana vrednost ekspresije določenih genov, ki pripadajo posameznemu biološkemu procesu, veliko bolj pomembna kot 20-kratno povečanje enega samega gena [39].

GSEA poskuša premostiti to kratkovidnost iskanja najpomembnejših genov, saj se osredotoča na ocenjevanje podatkov mikromrež DNA na nivoju skupin genov oziroma genskih setov (*gene sets*). Genski seti so skupine genov, sestavljene na podlagi biološkega znanja in raziskav o določenih procesih. Skupaj s kratkim opisom so funkcionalne skupine genov predstavljene v tabeli 4.2.



Slika 4.1: Faze analize GSEA (Vir: [32]).

Postopek analize GSEA je predstavljen na Sliki 4.1. Cilj analize GSEA je ugotoviti, ali se člani genskega seta S pojavljajo proti vrhu razvrščenega seznama genov L. V tem primeru je genski set koreliran z razlikovanjem med fenotipoma (razredoma). GSEA z določeno metriko rangira gene, na podlagi korelacije ekspresije posameznih genov in sposobnostjo razlikovanja med dvema razredoma. Za diskretne (kategorične) spremenljivke se uporabljajo metrike, kot je na primer *Signal2Noise*, ki smo jo uporabili tudi mi.

Zanimivo bi bilo morda poskusiti tudi s kakšno drugo obliko razvrščanja genov, glede na sposobnost razločevanja med posameznimi razredi. Ena izmed zanimivih izbir bi bila morda ocena ReliefF [28]. ReliefF je namreč metoda za ocenjevanje atributov, ki se pogosto uporablja za iskanje pomembnih atributov na področju strojnega učenja. Z uporabo dugačnega načina rangiranja genov bi posledično dobili tudi drugačno razvrstitev genskih setov. Vendar tega v procesu izdelave diplomskega dela na žalost nismo uspeli preskusiti.

Skupina	Opis
C1 (positional gene sets)	Skupine povezane z vsakim izmed krmosomov.
C2 (currated gene sets)	Skupine genov sestavljene iz različnih baz in na podlagi znanja ekspertov.
C3 (motif gene sets)	Geni glede na cis-regulatorne motive.
C4 (computational gene sets)	Skupine dobljene z rudarjenjem podatkov v bazah DNA mikročipov
C5 (GO gene sets)	Skupine sestavljene na podlagi GO (Gene Ontology).
C6 (Oncogenic signatures)	Skupine povezane z cličnimi potmi, ki so pogosto deregulirane pri obolelih za rakom.
C7 (immunologic signatures)	Skupine, ki predstavljajo stanje celic in perturbacije znotraj imunskega sistema.

Tabela 4.2: Skupine genskih setov (*gene sets*) glede na funkcionalna področja skupaj s kratkim opisom.

4.4 Faze analize GSEA

Analiza pomembnosti posameznih genskih setov je sestavljena iz zaporedja korakov. GSEA najprej, glede na izbrano metriko, izračuna pomembnost posameznega gena. Na ta način ustvari urejen seznam genov L glede na pomembnost vsakega izmed teh genov pri razločevanju med ciljnima razredoma.

FAZA 1: Izračun *Enrichment Score (ES)*. ES predstavlja stopnjo do katere je genski set S zastopan na zgornjem in spodnjem delu (ekstremih)

seznama L. ES se izračuna s pomočjo sprehoda po urejenem seznamu genov L in povečevanjem (ko naletimo na gen iz S) in zmanjševanjem (ko naletimo na gen, ki ni v S) tekoče vsote (*running sum statistics*) [39].

FAZA 2: Cenitev pomena nivoja (*significance level*) za ES. Statistična pomembnost ($nominalPvalue$) za ES se izračuna z uporabo permutacij label fenotipa [39]. Permutacija label fenotipa pomeni zamenjavo oznak razredov v katere sodi posamezen učni primer.

FAZA 3: Popravek za večkratno testiranje hipotez. Z normalizacijo ES z velikostjo genskega seta (*gene sets*) S, dobimo oceno NES (*Normalized Enrichment Score*). S to oceno iščemo, kateri genski seti se ujemajo z razredom v tolikšni meri, da tega ne moremo pripisati naključju. To statistiko izračunamo na velikem številu genskih setov, zato je potrebno, da se izognemo povečani verjetnosti lažnih pozitivnih ugotovitev (*false positive*). V naslednjem koraku se izračuna ocena FDR (*False Discovery Rate*), ki pomeni verjetnost, da genski set z določenim NES, predstavlja napako prve vrste (*false postive*). Za najbolj pomembne genske sete tako veljajo tisti, ki imajo FDR oceno < 0.25 [39].

Poglavje 5

Empirični del

V empiričnem delu diplomske naloge bomo predstavili potek raziskovanja potenciala gradnje skupinskih modelov na smisleno povezanih podmnožicah atributov, ki jih predstavljajo genski seti. Uvodoma bomo predstavili postopek izvedbe analize GSEA za razvrščanje genskih setov (podpoglavje: 5.1). Nato nas bo zanimala okvirna ocena uspešnosti različnih metod strojnega učenja na genetskih podatkih (podpoglavje: 5.2). V nadaljevanju nas bo zanimal vpliv izbiranja atributov na točnost posameznih metod (podpoglavje: 5.3). V okviru tega bomo primerjali uporabo različnih načinov izbiranja vhodnih spremenljivk in se osredotočili predvsem na rangiranje atributov s pomočjo metode ReliefF in GSEA rangiranega seznama atributov. V naslednjem koraku se bomo posvetili metodi naključnih podprostorov (podpoglavje: 5.4) in primerjali točnost skupinske metode v primerjavi s točnostjo osnovnega modela. Preučevali bomo tudi vpliv, ki ga na točnost celotne združbe predstavljata velikost podmnožice izbranih atributov in izbira osnovnega modela. Nato se bomo lotili modifikacije metode naključnih podprostorov (podpoglavje: 5.5). Najprej bomo gradili modele na naključno izbranih genskih setih, nato še na rangiranih genskih setih, ki jih bomo razvrščali s pomočjo analize GSEA. Dobljene rezultate obeh pristopov bomo primerjali

med seboj in z ostalimi metodami strojnega učenja. V naslednjem koraku bomo metodo naključnih podprostorov nadgradili še z uporabo odločitvenih pravil za gradnjo osnovnih modelov (podpoglavje: 5.6). V okviru tega bomo na kratko predstavili opis postopka implementacije algoritma CN2, s pomočjo katerega bomo gradili odločitvena pravila in primerjali dosežene rezultate z ostalimi metodami. V nadaljevanju bomo predstavili možne razloge, zakaj gradnja zbirnih modelov na genskih setih, kljub vloženemu trudu, ni dosegla želenih rezultatov (podpoglavje: 5.7). V okviru tega bomo kot enega izmed glavnih razlogov identificirali nihajočo klasifikacijsko točnost osnovnih modelov. Identificirano težavo bomo poskusili nasloviti z uporabo postopka skladanja klasifikatojev (*stacking*) na meta nivoju (podpoglavje: 5.8). Poglavje bomo zaključili s kratkim povzetkom doseženih rezultatov (podpoglavje: 5.9).

5.1 Prečno preverjanje in priprava podatkov za analizo GSEA

Uspešnost posameznih metod (klasifikatorjev) smo preizkusili na 10 različnih zbirkah podatkov o ekspresiji genov, pridobljenih z uporabo tehnologije mikročipov DNA. Za vsako posamezno zbirko smo izvedli postopek prečnega preverjanja. Prečno preverjanje je način ocenjevanja uspešnosti metod strojnega učenja, ki nam omogoča posplošitev dobljenih rezultatov. K-kratno prečno preverjanje (*K-fold cross validation*) je postopek s katerim učno množico razdelimo na K približno enako močnih podmnožic. Na vsaki izmed teh podmnožic zgradimo hipotezo tako, da za učenje uporabimo unijo preostalih podmnožic. Hipotezo testiramo na delu, ki ga pri učenju nismo uporabili. Postopek ponovimo K-krat. Uspešnost končne hipoteze ocenimo kot povprečno uspešnost vseh K hipotez na njihovih pripadajočih testnih podmnožicah [29].

Pri svojem delu smo uporabili priljubljeno orodje za strojno učenje Weka

(`http://www.cs.waikato.ac.nz/ml/weka/`). To orodje smo uporabili predvsem zato, ker je odprto kodno orodje, napisano v programskem jeziku java. Posledično imamo dostop do izvorne kode, ki jo lahko uporabimo za lažje in hitrejše delo ter spreminjamo glede na svoje potrebe. Dostop do izvorne kode se je izkazal kot velika prednost, saj nam je omogočil uporabo vgrajenih metod za prečno preverjanje, branje podatkov, filtriranje atributov ipd., ki smo jih potrebovali pri svojem delu. Na ta način smo se izognili potrebi po lastni implementaciji množice funkcionalnosti, ki smo jih potrebovali za svoje delo in se lažje osredotočili na same modifikacije obstoječih algoritmov in implementacijo dodatnih.

Poleg orodja za strojno učenje Weka smo za analizo genskih setov uporabili aplikacijo GSEA. Ta aplikacija je namenjena izvajanju analize na podatkih mikročipov DNA in kot vhod zahteva datoteke v lastnem formatu. Primere iz učne množice je zato potrebno iz oblike, ki jo uporablja Weka, pretvoriti v obliko primerno za uporabo v aplikaciji GSEA. Pomembne so predvsem tri vrste datotek.

- **GCT.** V datoteki GCT (*gene culster text file*) so posamezni učni primeri predstavljeni po stolpcih, po vrsticah pa imamo ekspresije genov za posamezen učni primer iz stolpca.

- **CLS.** Datoteka CLS (*categorical class text file*) definira uporabljene labele fenotipa (možne klasifikacijske razrede) in za vsak vzorec (učni primer) pove, kateri labeli (razredu) pripada. Pri generiranju teh datotek je pomembno, da upoštevamo vrstni red učnih primerov iz datotek GCT.

- **GMT.** Poleg tega potrebujemo še datoteko GMT (*gene matrix transposed file*), ki v vsaki vrstici vsebuje ime genskega seta, opis genskega seta in posamezne gene, ki pripadajo vsakemu izmed genskih setov.

Datoteke GCT, CLS in GMT je bilo potrebno pripraviti za vseh 10 podatkovnih zbirk. Pri tem smo zaradi izvajanja prej omenjenega postopka prečnega preverjena, vnaprej razdelili posamezno podatkovno zbirko na 10 učnih in 10 pripadajočih testnih delov. Za vsakega izmed teh desetih učnih delov posamezne zbirke smo pripravili ustrezne datoteke in izvedli analizo GSEA. Posledično je bilo potrebno pripraviti veliko število datotek, ki smo jih potrebovali za obdelavo posamezne zbirke podatkov. Poleg tega je bilo potrebno izvesti tudi veliko število posameznih analiz. Skupine genov (genski seti) so razporejene v različne kategorije, kar seveda še dodatno poveča kompleksnost, saj je potrebno za posamezno kategorijo genskih setov izvesti ločeno analizo. Pri izdelavi diplomskega dela smo uporabili tri različne podkategorije genskih setov. Uporabljene podskupine genskih setov so predstavljene v tabeli 5.1. Razlog za uporabo tega na prvi pogled precej kompliciranega načina dela, bomo predstavili v poglavju na temo izbiranja atributov (podpoglavje: 5.3.1).

Skupina	Opis
C2-CP (*chanonical pathways*)	Kanonične reprezentacije procesov sestavljene na podlagi znanja domenskih eksperotv.
C5-BP (*GO biological pathways*)	Genski seti sestavljeni s pomočjo genskih ontologij (GO) za biološke procese.
C5-MF (*GO molecular function*)	Genski seti sestavljeni s pomočjo genskih ontologij (GO) za molekularne funkcije.

Tabela 5.1: Uporabljene podskupine genskih setov (*gene sets*) s kratkim opisom

5.2 Osnovna uspešnost posameznih metod

Najprej nas je zanimalo, kako se na splošno obnašajo različne vrste klasifikacijskih algoritmov na problemu genetskih podatkov mikromrež DNA. V ta namen smo izvedli 10 ponovitev prečnega preverjanja na vseh desetih podatkovnih zbirkah in izmerili povprečno uspešnost posameznega klasifikacijskega algoritma. Pri tem smo uporabili vse atribute posamezne podatkovne zbirke. Izbrali smo nekaj različnih skupinskih (združevalnih) in nezdruževalnih algoritmov, ki smo jih uporabili z njihovimi privzetimi nastavitvami.

Odločili smo se za uporabo skupinskih metod **baggging**, **boosting**, **random forest** in **random subspace**, ki smo jih predstavili v okviru metod skupinskega učenja. Z namenom primerjave z »običajnimi« metodami strojnega učenja smo preverili tudi uspešnost nekaterih nezdruževalnih algoritmov. **J48** je Wekina implementacija Quilanovega algoritma *C4.5* za gradnjo odločitvenega drevesa [33]. **RandomTree** je implementacija odločitvenega drevesa z elementi naključnosti, saj je drevo v postopku dodajanja posameznih vozlišč, zgrajeno z izbiranjem najboljšega atributa med K naključno izbranimi atributi. **REPTree** je prav tako implementacija odločitvenega drevesa, vendar je to odločitveno drevo porezano. **JRip** je implementacija Cohenovega [8] algoritma za učenje odločitvenih pravil . **SimpleLogistic** je Wekina implementacija klasifikacijskega algoritma, ki kombinira uporabo logistične regresije in odločitvenih dreves [30]. **IBk** [1] je implementacija lenega algoritma za iskanje K najbližjih sosedov, ki klasificira primere v večinski razred K najbližjih učnih primerov. **NaiveBayes** [23] je implementacija metode Naivnega Bayesa, ki primere klasificira z računanjem pogojnih verjetnosti.

Tabela 5.2 prikazuje skupno klasifikacijsko točnost desetih ponovitev poskusa za posamezen algoritem. Opazimo lahko, da skupinski klasifikacijski modeli dosegajo boljšo povprečno uspešnost. Slednje lahko morda pripišemo

Algoritem	AvgCA	AvgAUC
AdaBoostM1 *	83,99	0,817
RandomSubspace*	83,96	0,832
RandomForest*	83,17	0,833
Bagging*	82,75	0,828
IBk	81,77	0,78
SimpleLogistic	81,62	0,8
NaiveBayes	80,18	0,74
JRip	75,16	0,715
REPTree	74,64	0,675
J48	72,67	0,696
RandomTree	68,95	0,654

Tabela 5.2: Povprečna klasifikacijska točnost in povprečna ploščina pod krivuljo ROC (AUC) za posamezne klasifikacijske algoritme. Algoritmi označeni z zvezdico sodijo v kategorijo skupinskih algoritmov.

prednostim združevanja. Metoda naključnih podprostorov (*RandomSubspace*) denimo kot privzeti algoritem za gradnjo osnovnih klasifikatorjev uporablja *REPTree*. V primerjavi z uspešnostjo osnovnega modela ($74,64\%$), pa lahko vidimo, da se skupinska metoda odreže veliko boljše ($83,96\%$).

Zanimivo pa je, da se tudi logistična regresija (*SimpleLogistic*) in metoda najbližjih sosedov (*IBk*), odrežejo povsem primerljivo z skupinskimi modeli. Še posebej morda preseneča uspeh metode najbližjih sosedov, če upoštevamo, da imajo naši podatki zelo veliko število atributov. Pri velikem številu atributov in majhnem številu učnih primerov lahko pride do t.i. pojava prekletstva dimenzionalnosti (*curse of dimensionality*). Bistvo tega pojava je dejstvo, da pri računanju razdalj v visodimenzinalnem prostoru, vsi učni primeri lahko postanejo z matematičnega vidika popolnoma enako oddaljeni (ekvidistančni). Posledično lahko razdalja med primeri, ki jo za klasifikacijo uporablja metoda najbližjih sosedov, izgubi svoj pomen in postane neuporabna [2]. Zakaj se to v našem primeru ni zgodilo, je vsekakor zanimivo vprašanje.

Ostali modeli, ki sodijo v kategorijo odločitvenih dreves (*REPTree*, *J48*,

RandomTree) in odločitvenih pravil (*JRip*), pa se v primerjavi s prej omenjenimi, odrežejo slabše. Eden izmed glavnih razlogov je verjetno veliko število atributov. Skupna značilnost vseh teh algoritmov je namreč izbiranje atributov v postopku gradnje klasifikatorja. Spričo velike izbire lahko hitro pride do napačne odločitve, kar vodi v neoptimalno strukturo posameznega klasifikatorja in posledično v slabšo uspešnost na testni množici. Problemu izbiranja atributov in gradnje klasifikatorjev na manjših podmnožicah, se bomo posvetili v naslednjem podpoglavju. Druga možna razlaga izvira iz lastnosti naših učnih podatkov. Poleg velikega števila atributov je za uporabljene podatkovne zbirke značilno tudi zelo majhno število učnih primerov. V primeru, da je za določanje razreda posameznega primera pomembnih veliko vhodnih spremenljivk (atributov), odločitveno drevo ne bo moglo upoštevati vseh. Če bi želeli upoštevati vse pomemebne atribute, bi namreč potrebovali ogromno drevo. Za gradnjo ogromnega drevesa pa nimamo na voljo dovolj učnih primerov. Algoritmu v procesu generiranja odločitvenih vozlišč posledično zelo hitro zmanjka učnih primerov in gradnja se ustavi. Poleg tega je lastnost odločitvenih dreves in odločitvenih pravil tudi to, da sestavljajo logične pogoje. V primeru, da gre za vpliv večih atributov, je ta vpliv boljše seštevati kot to delata na primer logistična regresija in metoda najbljižjih sosedov.

Razlog za slabše performanse vseh metod, kot bi jih potencialno lahko dosegli, je morda do neke mere lahko tudi posledica uporabe privzetih nastavitev. Pri skupinskih algoritmih smo povečali le število iteracij (iz privzetih 10 na priporočeno vrednost 100). S tem smo morda storili krivico ostalim metodam, vendar je potrebno poudariti, da na naš cilj ni bil nastavljanje parametrov in doseganje čim večje uspešnosti, pač pa smo želeli pridobiti zgolj orientacijske vrednosti.

5.3 Izbiranje atributov

Razvrščanje in ocenjevanje kakovosti posameznih vhodnih spremenljivk (atributov) glede na njihovo sposobnost pri razločevanju med učnimi primeri iz posameznega ciljnega razreda, je pomembno področje strojnega učenja. Postopki za izbiro in ocenjevanje atributov se pogosto uporabljajo že med samo gradnjo modela na učni množici. Sem sodi na primer izbiranje najboljšega atributa pri dodajanju vozlišč v procesu konstrukcije odločitvenega drevesa. Poleg tega vgrajenega načina selekcije, pa lahko podmnožico najboljših atributov izberemo že vnaprej. Nova učna množica z zmanjšano dimenzionalnostjo je nato vhod v posamezen algoritem za gradnjo modela.

Za ocenjevanje kakovosti atributov obstaja veliko različnih metod. Večina metod za ocenjevanje kakovosti atributov izhaja iz predpostavke, da so atributi med seboj neodvisni. Tej skupini pripadajo na primer informacijski prispevek (*information gain*), Gini index in J-ocena. Vendar pa se pri reševanju realnih problemov domnveva o neodvisnosti pogosto izkaže za napačno. Velikokrat namreč pravo pomembnost posameznih atributov določajo šele kombinacije vrednosti večih atributov. Metoda za ocenjevanje kvalitete posameznih atributov, ki pri svoji oceni upošteva tudi pogojno odvisnost, je ReliefF [34]. To metodo smo uporabili tudi mi. Želeli smo namreč primerjati uspešnost posameznih algoritmov na učni množici z zmanjšanim številom atributov. Zanimalo pa nas je tudi, kako se kot metoda za razvrščanje odreže ReliefF, v primerjavi z metodo, ki jo uporablja GSEA, saj tudi ta v začetni fazi ustvari rangiran seznam atributov, s pomočjo katerega nato razvršča genske sete. Želeli smo torej izvedeti, kako dober je ta seznam, v primerjavi z metodami za rangiranje atributov, ki izhajajo s področja strojnega učenja. Postopek in rezultate bomo bolj podrobno predstavili v nadaljevanju.

5.3.1 Izbiranje atributov in postopek prečnega preverjanja

Poleg vprašanj, kot so na primer, kakšno število atributov je potrebno izbrati in katero metodo uporabiti za izbiro, se ne glede na uporabljeno metodo, najprej pojavi povsem praktično vprašanje, na katerega je morda smiselno opozoriti.

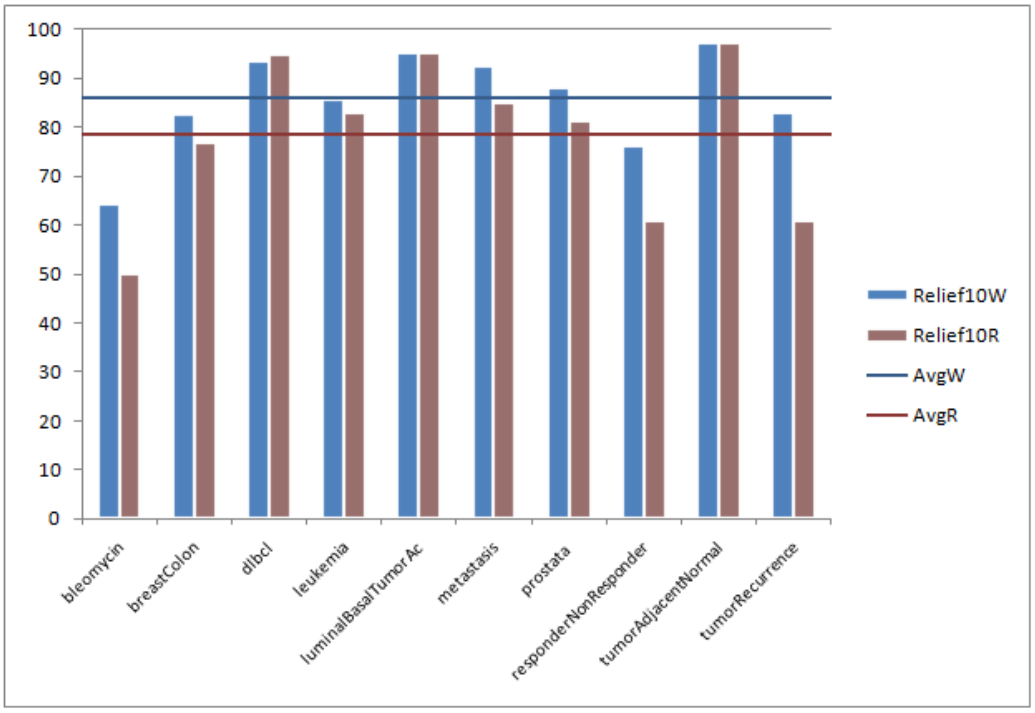
Prvo vprašanje, s katerim smo se srečali, je, kako pravzaprav izvesti postopek ocenjevanja in izbiranja atributov v kombinaciji s postopkom prečnega preverjanja. Ponujata se namreč dve možnosti. Prva, bistveno bolj enostavna s stališča implementacije in poleg tega tudi računsko manj zahtevna opcija je, da kvaliteto atributov ocenimo na celotni učni množici in šele nato izvedemo postopek 10 kratnega prečnega preverjanja s temi vnaprej razvrščenimi atributi. Druga, računsko in izvedbeno bolj zahtevana možnost je, da v vsaki izmed iteracij 10 kratnega prečnega preverjanja, izvedemo tudi postopek razvrščanja atributov. V posamezni iteraciji torej najprej razdelimo učno množico na učni in testni del. Za razvrščanje in ocenjevanje atributov pa uporabimo le podatke iz te nove učne množice.

Izkaže se da je prva, bolj enostavna opcija, žal napačna. Prva možnost je pravzaprav ena izmed nevarnih pasti, v katero se lahko občasno ujamejo raziskovalci s področja strojnega učenja [12]. V primeru, da se odločimo atribute oceniti pred samim postopkom prečnega preverjanja, na celotni učni množici in šele kasneje razdelimo to učno množico na novo učno in testno podmnožico, smo naredili napako. Pri razvrščanju in kasnejšem filtriranju na ta način vnaprej razvrščenih atributov smo za gradnjo našega modela, uporabili tudi podatke iz testne množico. Pri pravilni izvedbi testiranja modelov s prečnim preverjanjem se je tej zmoti potrebno izogniti, saj morata učna in testna množica ostati strogo ločeni in testne množice v nobenem primeru ne smemo uporabiti pred samim postopkom testiranja. Ne smemo si

namreč privoščiti, da bi izhajali iz domneve, da bomo v vsaki izmed desetih iteraciji prečnega preverjanja, dobili enako razvrstitev atributov. Obstaja namreč velika verjetnost, da bo v vsaki izmed posameznih iteracij, rangiran seznam atributov drugačen.

V primeru, da pomotoma izberemo prvo možnost in v procesu gradnje uporabimo podatke iz testne množice, lahko posledično dobimo preveč optimistične napovedi o uspešnosti posamezne metode. Zanimalo nas je, kako se to praktično odraža na naših podatkih, zato smo izvedli oba načina filtriranja atributov in primerjali dobljene rezultate. Slika 5.1 prikazuje povprečno klasifikacijsko točnost za posamezne podatkovne zbirke, v primeru pravilnega in napačnega načina izbiranja atributov. Iz slike je razvidno, da pri večini podatkovnih zbirk napačen postopek izbiranja atributov na celotni učni množici, dosega boljšo klasifikacijsko točnost. Podobno smo ponovili še na drugačnem številu izbranih atributov in ugotovili, da problem preoptimističnih prognoz uspešnosti modelov trdovratno vztraja.

Pravilna izvedba postopka prečnega preverjanja in razvrščanja atributov, je torej pomembna za bolj objektivno oceno performans modelov. Slednje je postalo velikega praktičnega pomena pri izvedbi analize GSEA in razvrščanju genskih setov, kjer se je zaradi pravilne izvedbe postopka prečnega preverjanja pokazala potreba po izvedbi relativno velikega števila posameznih analiz. Dodaten problem je, da postopka izvajanja analize GSEA nismo uspeli avtomatizirati, zato smo morali pri analizi posameznih učnih podmnožic ročno nastavljati vse potrebne parametre. Uporaba na ta način rangiranih genskih setov pa predstavlja tudi oviro pri uporabi vgrajenega prečnega preverjanja v Weki. Postalo je namreč nemogoče, da bi pri izvedbi posameznega testiranja s prečnim preverjanjem, uporabljali naključne podmnožice učnih in testnih množic, zato smo delitev na podmnožice izvedli le enkrat in potem izvajali prečno preverjanje na vnaprej pripravljenih particijah testnih in učnih množic. Na teh istih podmnožicah smo izvedli tudi analizo GSEA. Poleg tega smo preverili še, ali ta deterministična izbira učnih in testnih



Slika 5.1: Primejava klasifikacijske točnosti na posameznih podatkovnih zbirkah v primeru napačno (rdeča) in pravilno (modra) izvedenega postopka prečnega preverjanja.

podmnožic, morebiti vpliva na dobljene rezultate o uspešnosti posameznih metod, vendar se je izkazalo, da ne pride do omembe vrednih odstopanj.

5.3.2 ReliefF in GSEA

V naslednjem koraku nas je zanimalo, kako se posamezni skupinski (združevalni) in nezdruževalni klasifikacijski algoritmi odrežejo na izbrani podmnožici atributov. V ta namen smo za ocenjevanje kakovosti posameznih atributov uporabili dve različni metodi za ocenjevanje in razvrščanje atributov. Prva metoda je na področju strojnega učenja pogosto uporabljena metoda ReliefF, druga pa je rangiran seznam atributov, ki ga v procesu ocenjevanja pomemb-

	AvgCA			AvgAUC		
Algoritem	Osnovna	ReliefF	GSEA	Osnovna	ReliefF	GSEA
AdaBoostM1 *	83,99	84,16	83,20	0,82	0,83	0,81
RS *	83,96	83,06	82,52	0,83	0,84	0,82
RF *	83,17	84,13	83,22	0,83	0,84	0,83
Bagging *	82,75	83,79	80,87	0,83	0,83	0,80
IBk	81,77	83,21	82,49	0,78	0,79	0,80
LogReg	81,62	83,47	82,49	0,80	0,82	0,81
NaiveBayes	80,18	81,80	82,53	0,75	0,82	0,80
JRip	75,16	79,02	77,24	0,72	0,75	0,73
REPTree	74,64	75,58	75,59	0,68	0,72	0,71
J48	72,67	76,82	76,23	0,70	0,74	0,74
RandomTree	68,95	76,26	76,14	0,65	0,73	0,73

Tabela 5.3: Povprečna klasifikacijska točnost (*AvgCA*) in povprečna ploščina pod krivuljo ROC (*AvgAUC*) za posamezne klasifikacijske algoritme brez uporabe selekcije atributov, pri uporabi metode ReliefF in metode GSEA za ocenjevanje kakovosti atributov. Algoritmi označeni z zvezdico sodijo v kategorijo skupinskih algoritmov.

nosti posameznih genskih setov, zgradi analitično orodje GSEA.

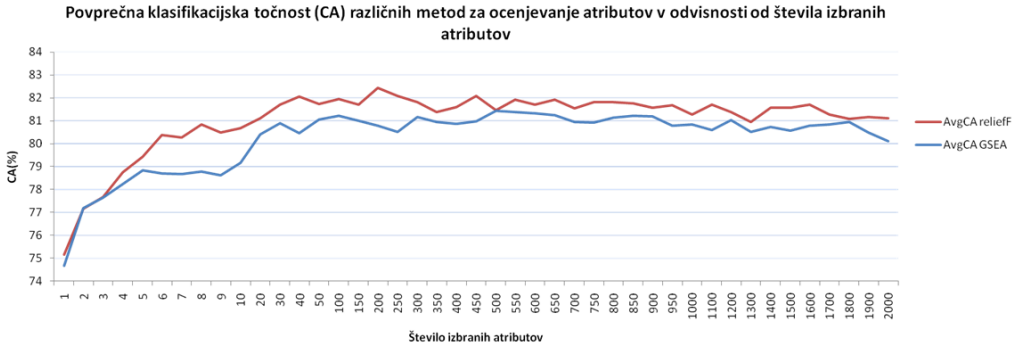
Uspešnost posameznih algoritmov je z ocenama povprečne klasifikacijske točnosti (CA) in ploščine pod krivuljo ROC (AUC) prikazana v tabeli 5.3. Izračunane povprečne ocene uspešnosti so rezultat povprečnih vrednosti prečnega preverjanja podatkovnih zbirk pri uporabi različnega števila atributov (od 1 do 200 atributov). Opazimo lahko, da ima izbiranje atributov, pozitiven vpliv na uspešnost običajnih klasifikacijskih algoritmov. Ne glede na uporabljeno metodo za ocenjevanje in izbiranje atributov se povprečna klasifikacijska točnost te skupine modelov precej izboljša. Nazoren primer tega je odločitveno drevo (*J48*), ki na vseh atributih doseže $72,67\%$ natančnost, z izbranimi atributi pa se točnost zviša na $76,82\%$ (ReliefF) oz. $76,23\%$ (GSEA). Slednje je morda lep primer tega, kako lahko odvečni in redundantni atributi negativno vplivajo na uspešnost algoritmov strojnega učenja.

Precej bolj zanimiv je vpliv izbiranja podmnožice najboljših atributov na skupinske metode. Glede na dobljene rezultate se namreč zdi, da s selekcijo atributov ne dosežemo pričakovanih učinkov izboljšane uspešnosti. Celo več, zdi se, da lahko morda dosežemo celo nasproten učinek, saj se uspešnost skupinskih algoritmov v nekaterih primerih celo poslabša. Slednje je še posebej opazno, če uporabimo metodo rangiranja GSEA. Tudi v primeru uporabe metode ReliefF izboljšanje ni ravno omembe vredno. Eden izmed možnih razlogov za slabšo uspešnost metod za izbiranje atributov v primeru skupinskih metod je morda dejstvo, ki izhaja iz teorije združevanja. Eden izmed pogojev za uspeh skupinskih metod je poleg točnosti tudi različnost osnovnih klasifikatorjev. Problem pri grajenju močnih osnovnih klasifikatorjev na izbranih podmnožicah kvalitetnih atributov je, da so si osnovni klasifikatorji med seboj morda preveč podobni. Na ta način lahko izgubimo različnost na osnovnem nivoju, kar vpliva na uspešnost skupinske sheme. To je ena izmed možnih razlag. Seveda bi bilo za potrditev te domneve vsekakor potrebno opraviti več poskusov. Druga možna razlaga je, da smo z izbiranjem manjše podmnožice dobrih atributov, skupinski model prikrajšali za atribute, ki so na splošno gledano zelo slabi, v kakšnih konkretnih situacijah pa morda znajo biti zelo dobri. Skupinski modeli, ki zgradijo veliko število različnih osnovnih modelov, lahko morda pri gradnji naletijo tudi na take situacije, zato jim morebiti utegne škodovati, če jim te atribute odstranimo.

5.3.3 Primerjava metode ReliefF in GSEA

Zanimala nas je tudi primerjava uspešnosti med metodama za razvrščanje atributov ReliefF in GSEA. V ta namen smo za vse algoritme (tako skupinske kot tudi nezdruževalne) pri različnem številu izbranih atributov, izračunali povprečno klasifikacijsko točnost, doseženo z uporabo metode ReliefF in metode GSEA. Povprečna klasifikacijska točnost je prikazana na Sliki 5.2. Opazimo lahko, da sta metodi enako dobri le pri manjšem številu atributov.

Slednje gre pripisati opažanju, da so najbolje rangirani atributi pri obeh metodah pogosto identični. Treba je seveda opozoriti, da gre samo opažanje, ki ga nismo sistematično preverili. Pri večjem številu atributov se zdi, da se ReliefF na splošno odreže boljše.



Slika 5.2: Primerjava povprečne klasifikacijske točnosti (CA) metode za ocenjevanje atributov ReliefF (rdeča) in GSEA (modra) pri različnem številu izbranih atributov

Pojavi se seveda logično vprašanje, zakaj ReliefF večinoma deluje nekoliko boljše, kot GSEA seznam urejenih atributov. V ta namen smo informativno primerjali podobnost razvrščanja metode GSEA z nekaterimi drugimi znanimi metodami za ocenjevanje kakovosti atributov. Poleg že omenjene metode ReliefF, smo atribute rangirali še z oceno informacijski prispevek (*information gain*), Gain Ratio in SVM (*wrapper* metoda za razvrščanje atributov v Weki). Urejen seznam atributov, ki ga izdela posamezna metoda, smo primerjali s seznamom, ki ga uporablja GSEA. Zanimala nas je podobnost med posameznimi metodami. Podobnost smo merili tako, da smo pogledali odstotek enakih atributov v določenem deležu najboljše razvrščenih atributov na urejenem seznamu GSEA in rangiranem seznamu drugih metod. Dobljene rezultate prikazuje tabela 5.4.

	odstotek enakih			
no_of_att	RELIEFF	INFORMATION GAIN	GAIN RATIO	SVM
6	0,33	0,5	0,5	0
10	0,3	0,5	0,6	0,2
20	0,5	0,7	0,55	0,45
30	0,53	0,7	0,53	0,3
40	0,55	0,75	0,58	0,38
50	0,5	0,78	0,64	0,36
60	0,52	0,75	0,58	0,38
70	0,56	0,67	0,54	0,34
100	0,58	0,65	0,49	0,29
150	0,58	0,69	0,51	0,35
200	0,59	0,72	0,47	0,37
500	0,6	0,74	0,65	0,45
750	0,6	0,74	0,74	0,47

Tabela 5.4: Odstotek enakih atributov med top atributi, ki jih predlaga GSEA in nekaterimi drugimi metodami za ocenjevanje kakovosti atributov

Opazimo lahko, da je odstotek enakih atributov med ReliefF in GSEA manjši, kot če metodo GSEA primerjamo z oceno informacijski prispevek ali Gain Ratio. Informacijski prispevek in Gain Ratio sta tako imenovali kratkovidni metodi, kar pomeni, da ne upoštevata pogojne odvisnosti med atributi. Na podlagi tega bi morda lahko sklepali na večjo podobnost med GSEA in t.i. kratkovidnimi metodami ocenjevanja pomembnosti atributov. ReliefF pa je metoda, ki se zna spopasti z močno koreliranimi spremenljivkami. V genetskih podatkih mikročipov DNA so tovrstni atributi prav gotovo prisotni, saj gre za podatke biološke narave, ki so rezultat zapletene interakcije različnih procesov.

Seveda je predstavljena tabela in morebitni zaključek, ki ga lahko potegnemo, zgolj informativne narave. Predstavljeni odstotki so bili namreč izračunani le na eni podatkovni zbirki (Leukemia). Poleg tega smo pri računanju odstotka podobnosti uporabili precej primitivno obliko štetja enakih atributov brez upoštevanja vrstnega reda. Matematično gledano metrika *signal2noise ratio*, ki jo za ocenjevanje pomembnosti atributov uporablja GSEA, na prvi pogled res izgleda precej »kratkovidno«, čeprav se v matematične podrobnosti nismo spuščali, saj nas niso toliko zanimali urejeni seznami genov, pač pa bolj končni rezultat. Končni rezultat analize GSEA predstavljajo urejeni genski seti, ki jim bomo več pozornosti namenili v nadaljevanju. Urejen seznam pomembnosti posameznih atributov, ki ga ustvari GSEA, je le vmesni korak. Vendar je ta vmesni korak morda pomemben člen v verigi, saj se uporablja za kasnejše razvrščanje genskih setov. S tega zornega kota bi bilo morda zanimivo z biološkega stališča analizirati rangiranje genskih setov, ki ga dobimo z uporabo metod za ocenjevanje pomembnosti atributov, ki izvirajo s področja strojnega učenja. Seveda se zaradi pomanjkanja ustreznega znanja s področja genetike tega nismo mogli kompetentno lotiti.

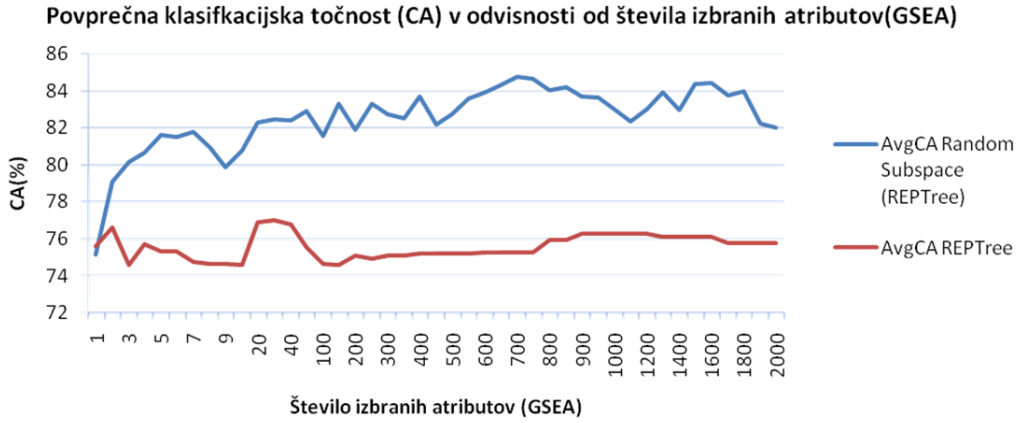
5.4 Metoda naključnih podprostorov

V nadaljevanju se bomo, bolj kot na druge metode, prvenstveno osredotočili na metodo naključnih podprostorov (*random subspace*), ki smo jo na kratko predstavili že v poglavju o metodah skupinskega učenja (podpoglavje: 3.2.4). Glavna značilnost te metode je gradnja množice osnovnih klasifikatorjev na naključno izbranih podmnožicah atributov. Z izbiranjem različnih podmnožic prostora vhodnih spremenljivk dosežemo različnost osnovnih klasifikatorjev. Pod pogojem, da so osnovni klasifikatorji bolj uspešni od naključnega ugibanja in med seboj neodvisni, z združevanjem osnovnih klasifikatorjev v glasovalno shemo načeloma dosežemo sinergijske učinke združevanja,

ki se odražajo v večji uspešnosti.

5.4.1 Primerjava med uspešnostjo skupinskega in osnovnega modela

Pojav združevalnega doprinosa je prikazan na Sliki 5.3. Modra črta na grafu prikazuje povprečno klasifikacijsko točnost pri različnem številu izbranih atributov z rangiranega seznama atributov v primeru, da uporabimo skupinsko metodo naključnih podprostorv (*random subspace*), ki kot osnovni klasifikator uporablja algoritem REPTree. Rdeča črta pa prikazuje točnost osnovnega klasifikatorja REPTree. Opazimo lahko, da je uspešnost osnovne in skupinske metode enaka le pri enem samem atributu. Pri večjem številu atributov se skupinski algoritem z gradnjo večih osnovnih klasifikatorjev na podmnožicah atributov iz določenega nabora možnih atributov odreže boljše od svojega osnovnega algoritma, zgrajenega na popolnoma istem naboru atributov.



Slika 5.3: Primerjava klasifikacijske točnosti skupinske metode naključnih podprostorov (*random subspace*), ki kot osnovni klasifikator uporablja REPTree (modra) in klasifikacijske točnosti osnovnega klasikatatorja REPTree (rdeča) pri različnem številu izbranih atributov.

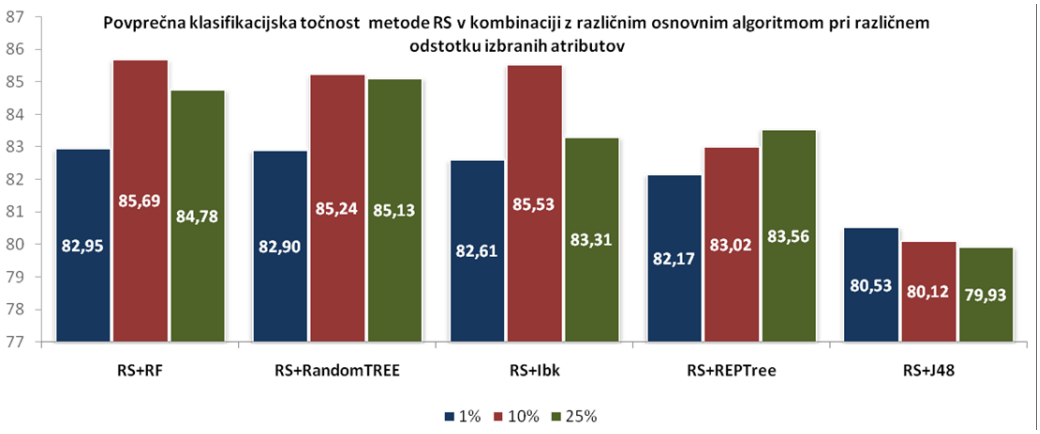
5.4.2 Vpliv velikosti podprostorov in izbire osnovnega modela

Dva izmed parametrov, ki jih lahko nastavimo pri uporabi metode naključnih podprostorov, sta osnovni algoritem in velikost izbranega podprostora atributov. Zanimalo nas je, kakšen vpliv ima izbira teh dveh parametrov na uspešnost metode.

V ta namen smo na naših podatkih izvedli večje število ponovitev prečnega preverjanja in pri tem spreminjali izbrani osnovni algoritem ter velikost naključno izbrane podmnožice atributov, na katerem je zgrajen posamezen osnovni klasifikator. Dobljene rezultate prikazuje graf na Sliki 5.4.

Opazimo lahko, da na uspešnost vplivata tako osnovni algoritem kot tudi velikost podmnožice izbranih atributov. Metoda se najslabše odreže v kombinaciji z odločitvenim drevesom J48, za katerega smo že prej lahko opazili, da se obnaša slabše kot ostali algoritmi (73,67% glej: tabela 5.2). Zanimiva pa je morda relativno dobra uspešnost algoritma RandomTree, ki se sam po sebi ne odreže najboljše (68,95% glej: tabela 5.2). V primeru, da ga kombiniramo z metodo naključnih podprostorov, se zdi, da kot osnovni algoritem morda v našem primeru deluje boljše od v Weki privzetega osnovnega algoritma REPTree.

Poleg tega je kombinacija naključnih podprostorov in RandomTree povsem primerljiva z uporabo naključnega gozda (*random forest* RF) na osnovnem nivoju. Za kombinacijo naključnega gozda in naključnih podprostorov bi morda lahko rekli, da gre pravzaprav za dva nivoja združevanja. Tudi naključni gozd je po naravi skupinski algoritem. Posledično se poveča kompleksnost izvajanja, saj že na osnovnem nivoju pri izgradnji posameznega klasifikatorja uporabljamo skupinski algoritem. Sodeč po dobljenih rezultatih bi morda lahko rekli, da povečana kompleksnost ne odtehta rezultatov,



Slika 5.4: Klasifikacijska točnost skupinske metode naključnih podprostorov (*random subspace* RS), ki kot osnovni klasifikator uporablja različne algoritme, pri različnem odstotku izbranih atributov.

saj kombinacija naključnih podprostorov z RandomTree dosega primerljive rezultate na bistveno hitrejši način.

5.5 Metoda združevanja osnovnih modelov na genskih setih

Pomembna lastnost genskih podatkov mikročipov DNA, ki smo jih uporabili pri izdelavi diplomskega dela, je možnost uporabe smiselnih skupin atributov, kot jih definirajo genski seti. V nadaljevanju se bomo posvetili uporabi takih smiselnih skupin genov v procesu gradnje skupinskih modelov. Zanimalo nas je, kako se obnaša metoda naključnih podprostorov, ki namesto na naključnih podskupinah atributov osnovne klasifikatorje gradi na tovrstnih smiselnih skupinah genov. Na ta način smo želeli nasloviti nekatere pomanjkljivosti metode naključnih podprostorov.

Ključ uspeha metode naključnih podprostorov je grajenje klasifikator-

jev na naključnih podmnožicah atributov. Gradnja osnovnih klasifikatorjev na naključno izbranih atributih pa lahko povzroči tudi določene probleme. Nekateri izbrani podprostori so, kot posledica naključne izbire, lahko neinformativni. Na tovrstnih slabo izbranih atributih ne moremo zgraditi dobrih osnovnih klasifikatorjev, kar posledično zmanjšuje uspešnost celotne združbe.

O modificiranju metode naključnih podprostorov je na primer razmišljal Garcia [17], ki v enem izmed svojih člankov predlaga metodo z imenom »*Not So Random Subspace (NSRS)*« (»*Ne tako zelo naključni podprostori*«). Ta modifikacija metode naključnih podprostorov stremi k izbiranju bolj informativnih podskupin atributov, pri tem pa na zanimiv način uporabi metodo boostinga. Metoda pred izgradnjo osnovnega klasifikatorja na naključnem podprostoru minimalizira klasifikacijsko napako posameznega podprostora in se na ta način poskuša izogniti gradnji osnovnih klasifikatorjev na slabih skupinah atributov. Klasifikacijska napaka, ki jo minimizira, je utežena z distribucijo primerov v učni množici, ki jo dobimo z uporabo boostinga [17]. Predlagana možnost se izkaže za uspešnejšo od uporabe osnovne oblike metode naključnih podprostorov.

5.5.1 Naključni genski seti

Naš cilj je bil preizkusiti, kako deluje metoda naključnih podprostorov, če namesto na naključno izbranih skupin atributov osnovne klasifikatorje gradi na atributih, ki pripadajo določenemu genskemu setu. Z uporabo ekspertnega predznanja, ki ga predstavljajo definirane skupine genov, smo želeli zgraditi bolj točne osnovne klasifikatorje in na ta način izboljšati uspešnost metode naključnih podprostorov.

Za potrebe ocenjevanja uspešnosti metode je bilo potrebno prilagoditi obstoječo implementacijo metode naključnih podprostorov v orodju za strojno

učenje Weka. Spremeniti je bilo potrebno predvsem način gradnje osnovnih klasifikatorjev, ki so v obstoječi implementaciji zgrajeni na naključno izbranih atributih. Posamezni genski set iz seznama genskih setov, ki pripadajo določenemu funkcionalnemu sklopu genskih setov, je sestavljen iz imena in pripadajočih genov. Vsakega izmed osnovnih klasifikatorjev smo zato zgradili na enem izmed teh genskih setov in pri tem uporabili točno določene gene. Osnovni klasifikatorji za klasifikacijo novega primera uporabljajo večinsko glasovalno shemo. Modificirano metodo naključnih podprostorov, ki smo jo dobili na ta način, ostale komponente (razredi) v Weki uporabljajo kot vsak drug klasifikator. Na ta način ni bilo potrebno spremeniti implementacije ostalih metod, na primer metode za ocenjevanje klasifikatorjev, prečnega preverjanja ipd.

Zanimalo nas je, kako se metoda obnese z različno velikim odstotkom uporabljenih genskih setov iz posameznega funkcionalnega sklopa genskih setov (C2BP, C5BP, C5MF). Pomen posameznih sklopov je opisan v sklopu predstavitve metode GSEA (tabela: 5.1). Z izbiranjem različnega odstotka genskih setov posledično vplivamo na število zgrajenih osnovnih klasifikatorjev. Manjši odstotek uporabljenih genskih setov tako pomeni tudi manjše število osnovnih klasifikatorjev.

Kot algoritem za gradnjo osnovnih klasifikatorjev smo najprej uporabili privzeti algoritem REPTree. Rezultati so predstavljeni v tabeli 5.5. Dobljena klasifikacijska točnost je rezultat večjega števila ponovitev poskusov prečnega preverjanja za posamezne genske sklope in posamezen odstotek naključno izbranih genskih setov. Za orientacijo je podana tudi klasifikacijska točnost nespremenjene metode naključnih podprostrov, ki gradi osnovne klasifikatorje na naključno izbranih atributih.

Glede na dobljene rezultate se zdi, da je originalna metoda uspešnejša. Še posebej to velja, če uporabimo manjši odstotek naključnih genskih setov (0,1 in 1 odstotek). Slednje je verjetno predvsem posledica majhnega

Odstotek naključnih genskih setov	Metoda			
Odstotek uporabljenih genskih setov	C2CP	C5BP	C5MF	Osnovna metoda
0.1%	72,86	69,67	74,02	83,96
1%	77,27	76,46	75,26	83,96
10%	80,17	80,63	80,57	83,96
50%	80,49	81,55	81,15	83,96

Tabela 5.5: Klasifikacijska točnost modificirane metode naključnih podprostorov. Osnovni klasifikatorji so zgrajeni na različnih funkcionalnih sklopih genskih setov (C2CP, C5BP, C5MF). Tabela prikazuje doseženo točnost metode, pri različnem odstotku uporabljenih naključnih genskih setov iz posameznega sklopa. Podana je tudi osnovna natančnost originalne metode naključnih podprostorov, ki temelji na naključnem izbiranju atributov iz celotne množice razpoložljivih atributov.

števila osnovnih klasifikatorjev. Pri večjem odstotku izbranih genskih setov se uspešnost sicer izboljša, vendar še vedno ne doseže osnovne različice. Eden izmed razlogov je morda dejstvo, da kljub temu da obe metodi zgradita večje število osnovnih modelov, mora v primeru modificirane metode posamezen klasifikator na osnovnem nivoju, osnovni model zgraditi na točno določenih atributih. Pri osnovni varianti metode pa dobi vsak osnovni klasifikator na voljo večjo število naključno izbranih atributov. Iz te večje množice lahko morda nato v procesu gradnje izbere boljše atribute. Slednje je le ugibanje. Poleg tega lahko opazimo še, da je pri večjem številu izbranih genskih setov, klasifikacijska točnost približno enka, ne glede na to kateri genetski sklop uporabimo. Slednje morda niti ni tako zelo presenetljivo, saj smo genske sete iz posameznih sklopov izbirali povsem naključno.

5.5.2 Urejeni genski seti

V naslednjem koraku smo poskusili osnovne klasifikatorje zgraditi še na rangiranih genskih setih. Rangirani genski seti, ki smo jih uporabili za gradnjo

Odstotek naključnih genskih setov	Metoda		
Odstotek uporabljenih genskih setov	Urejeni seti	Naključni seti	Osnovna metoda
0.1%	78,52	72,86	83,96
1%	82,49	77,27	83,96
10%	81,61	80,17	83,96
50%	81,44	80,49	83,96

Tabela 5.6: Klasifikacijska točnost modificirane metode naključnih podprostorov pri naključni izbiri genskih setov in uporabi najbolje ocenjenih genetskih setov. Podana je tudi osnovna natančnost originalne metode naključnih podprostorov, ki temelji na naključnem izbiranju atributov iz celotne množice razpoložljivih atributov.

klasifikatorjev so rezultat analize GSEA. Postopek izvedbe analize GSEA je opisan v enem izmed prejšnjih poglavij (poglavje: 4). Urejena lestvica genskih setov odraža pomembnost posameznega genskega seta za ločevanje učnih primerov, ki pripadajo dvema različnima klasifikacijskima razredoma. Najboljše rangirani genski seti odražajo pomembnost, ki ga ima posamezna povezana skupina genov pri določanju razreda. Na ta način smo želeli namesto naključnih genskih setov, ki morda niso povezani z razredom, uporabiti le tiste smiselne skupine genov, ki v največji meri določajo razred primerov v naši učni množici. V ta namen smo metodo naključnih podprostorov spremenili tako, da smo posamezne osnovne klasifikatorje zgradili na genih, ki sestavljajo posameznega izmed najbolje ocenjenih genskih setov. Izbrali smo različno velik delež najboljše ocenjenih genskih setov. Rezultati večih ponovitev 10 kratnega prečnega preverjanja so predstavljeni tabeli 5.6.

Opazimo lahko, da ima uporaba najbolje ocenjenih genskih setov določen vpliv na uspešnost naše modificirane metode naključnih podprostorov. Gradnja osnovnih klasifikatorjev na genih, ki pripadajo genskim setom, ki so pomembni za razločevanje med primeri učne množice, daje boljše rezultate kot uporaba naključno izbranih genskih setov.

Ta vpliv je najbolj opazen pri manjšem odstotku genskih setov. Slednje

je verjetno posledica tega, da z manjšim deležem genskih setov, posledično zgradimo tudi manjše število osnovnih klasifikatorjev. V primeru manjšega števila naključno izbranih genskih setov obstaja veliko tveganje, da bomo izbrali genske sete, ki ne nosijo informacije za reševanje danega problema. Takih osnovnih klasifikatorjev je v tem scenariju zelo malo, zato obstaja možnost, da je med njimi veliko nenatančnih, posledično pa tudi združevanje ne more oplemenititi rezultatov. V primeru, da uporabimo manjše število osnovnih klasifikatorjev zgrajenih na informativnih podmnožicah atributov, se tej nevarnosti do neke mere lahko izognemo. Slednje lahko morda pojasni precejšno izboljšanje uspešnosti na majhni podmnožici genskih setov.

Pri večjem odstotku vključenih genskih setov je sicer tudi opaziti določeno izboljšanje, vendar ne v taki meri, kot bi si morda želeli. Zdi se pravzaprav celo, da se trend počasi obrača in se uspešnost z vključevanjem večjega števila genskih setov morda celo znižuje. Slednje je morebiti posledica tega, da ko se premikamo po urejenem seznamu od glave proti repu, genski seti postajajo vedno manj informativni. Več ko uporabimo takih nenatančnih genskih setov, bolj to negativno vpliva na celotno združbo, saj se v glasovanje čedalje bolj mešajo napačni glasovi. V večinski glasovalni shemi pa velja pravilo "en osnovni klasifikator en glas". Možna korekcija tega problema bi bila morda utežitev glasov posameznih osnovnih klasifikatorjev glede na rang iz urejenega seznama. Vendar pa, kot bomo predstavili kasneje (podpoglavje: 5.7), padanje uspešnosti glede na rang posameznega genskega seta, verjetno ni glavni problem, s katerim bi lahko ta pojav v celoti pojasnili.

Rezultati dajejo morda tudi slutiti, da pri večjem številu zgrajenih osnovnih klasifikatorjev naključne metode izbiranja lahko delujejo precej primerljivo s sistematično izbiro. Slednje postane še posebej velika prednost, če pogledamo s stališča vloženega dela, ki ga je potrebno opraviti za razvrščanje genskih setov, medtem ko je izbiranje večjega števila naključnih genskih setov zelo enostavno. Kljub uporabi rangiranih genskih setov osnovna varianta še vedno dosega boljše rezultate kot naša modificirana metoda.

5.6 Naključni podprostori v kombinaciji z odločitvenimi pravili CN2

5.6.1 Odločitvena pravila

Eden izmed ciljev diplomskega dela je bil tudi uporaba odločitvenih pravil, zgrajenih na genskih setih in vključenih v glasovalno shemo skupinskega klasifikatorja. Slednje je prav tako modifikacija metode naključnih podprostorov, le da na osnovnem nivoju za gradnjo osnovnih klasifikatorjev uporabimo algoritem za indukcijo odločitvenih pravil. V ta namen smo implementirali znani algoritem za gradnjo odločitvenih pravil CN2 [6, 5].

Poskus generiranja velikega števila naključnih odločitvenih pravil (*random rules*), ki temelji na ideji stohastične diskriminacije, je predstavljen v delu Pfahringerja in sodelavcev [31]. Za razliko od klasičnih pristopov gradnje odločitvenih pravil s pristopom deli vladaj, kamor sodi na primer algoritem CN2, je v primeru te metode naključnih pravil uporabljena filozofija od konkretnega k splošnemu. Popolnoma naključno generiranje pravil se izkaže za precej neučinkovito, zato postopek deluje na ta način, da v prvem koraku inicializira pokritje vseh učnih primerov. V naslednjih iteracijah naključno izbere učni primer, ki je do tega trenutka podpovprečno pokrit. Za izbrani učni primer metoda nato generira začetno odločitveno pravilo. V naslednji fazi izmed preostalih učnih primerov naključno izbere vzorec učnih primerov, ki jih potem pokrije z odvzemanjem pogojev iz začetnega pravila. Na ta način zagotavlja enakomerno pokritje učne množice, saj vedno izbere primer z nizkim pokritjem. V povprečju pa s pokrivanjem naključno izbranega vzorca pri generiranju posameznega pravila zagotovi relativno močna pravila [31].

Tudi nas je zanimalo, kako se odločitvena pravila obnašajo v okviru sku-

pinskih metod, vendar pa odločitvenih pravil nismo želeli graditi na celotni množici vhodnih spremenljivk, pač pa smo želeli za njihovo gradnjo uporabiti vnaprej definirane genske sete. V ta namen smo implementirali algoritem CN2. Postopek implementacije in delovanje algoritma bomo predstavili v nadaljevanju.

5.6.2 Implementacija algoritma CN2

Algoritem CN2 smo implementirali v programskem jeziku java in ga kot nov klasifikacijski algoritem vključili v odprtokodno orodje za strojno učenje Weka. Pri implementaciji smo si pomagali z opisom algoritma v člankih originalnih avtorjev Clarka in Nibbleta [6, 5]. Poleg tega nam je bila v veliko pomoč implementacija algoritma v programskem jeziku C, ki jo lahko najdemo na spletni strani enega izmed avtorjev (`http://www.cs.utexas.edu/users/pclark/software`). S pomočjo te kode smo lahko rekonstruirali določene podrobnosti delovanja algoritma, ki v člankih niso natančno predstavljene.

Algoritem CN2 ima dva visoko nivojska načina delovanja. Generiramo lahko neurejen seznam odločitvenih pravil (*unordered rule list*) ali pa urejen seznam odločitvenih pravil t.i. odločitveni seznam (*decision list*). Razlika med obema načinoma delovanja je pomembna predvsem pri klasifikaciji novih primerov. V primeru neurejenega seznama odločitvenih pravil lahko nov primer pokriva več različnih pravil. V urejenem seznamu pa se to ne more zgoditi. Urejen seznam pravil pri klasifikaciji primera preiskujemo po vrsti, saj vsako naslednje pravilo vsebuje tudi vse pogoje iz prejšnjega. Tak seznam preiskujemo dokler se ne sproži eno izmed pravil.

S stališča implementacije je razlika med urejenim in neurejenim seznamom pomembna le pri načinu obravnavanja učnih podatkov, kar bomo opisali v

nadaljevanju. Nizkonivojska procedura iskanja najboljšega pravila je tako v primeru neurejenih kot tudi urejenih pravil popolnoma enaka. Ena izmed pomembnih karakteristik CN2 algoritma je iskanje v snopu. Pri dodajanju pogojev v odločitveno pravilo se poskušamo izogniti pastem požrešnega iskanja najboljšega vozlišča, saj v snopu obdržimo več potencialno najboljših vozlišč. Kot mero za oceno kvalitete posameznega vozlišča lahko uporabimo različne ocene. V naši implementaciji smo omogočili uporabo Laplaceve ocene, entropije in naivne ocene.

Neurejen seznam pravil (*Unordered rule list*). Algoritem generira pravila za vsako vrednost razredne spremenljivke posebej. Potem, ko najde najboljše pravilo za razred, ki ga trenutno preiskuje, izloči vse pokrite primere pozitivnega (trenutnega) razreda in nadaljuje z iskanjem novih pravil. Ko se zaradi pomanjkanja dobrih pravil zaključi postopek iskanja za vse možne vrednosti razredne spremenljivke, se na konec seznama doda privzeto pravilo. Naloga privzetega pravila je primerom, ki jih ne pokriva nobeno izmed pravil, pripisati večinski razred. Pri klasifikaciji novega testnega primera se lahko zgodi, da ta primer pokriva več različnih odločitvenih pravil, zato za končno napoved uporabimo glasovalni mehanizem. Na Sliki 5.5a je prikazan primer neurejenega seznam pravil, ki ga proizvede naša implementacija CN2 za eno izmed vzorčnih podatkovnih zbirk v Weki, ki se ukvarja z problemom podpisovanja tipa kontaktnih leč (*contact-lenses.arff*).

Urejen seznam pravil (*Ordered rule list)*. Druga možnost je generiranje urejenega seznama pravil oziroma odločitvenega seznama (*decision list*). Pomembna razlika v delovanju je, da ne fiksiramo razreda, pač pa iščemo najboljše selektorje, ki pokrivajo največ učnih primerov. Ustrezen razred za posamezno pravilo dodamo šele na koncu [25]. Za razliko od prejšnjega načina po zaključku generiranja posameznega pravila ne glede na razred odstranimo vse pozitivne primere, ki jih pravilo pokriva. V primeru neurejenih pravil smo namreč odstranili samo primere razreda, ki ga trenutno preiskujemo. Z odstranitvijo vseh pozitivnih primerov algoritmu preprečimo

```
=== Classifier model (full training set) ===

CN2 rule induction algorithm.
Star size: 5
algorithm type: Unordered
Error estimate: laplacian
Discretiazion Type: default
INDUCED RULES:
IF    astigmatism = no   AND tear-prod-rate = normal
   THAN  contact-lenses = soft [5 0 1]

IF    spectacle-prescrip = myope   AND astigmatism = yes   AND tear-prod-rate = normal
   THAN  contact-lenses = hard [0 3 0]

IF    age = young   AND astigmatism = yes   AND tear-prod-rate = normal
   THAN  contact-lenses = hard [0 2 0]

IF    tear-prod-rate = reduced
   THAN  contact-lenses = none [0 0 12]

IF    spectacle-prescrip = hypermetrope   AND astigmatism = yes
   THAN  contact-lenses = none [0 1 5]

IF    age = presbyopic   AND spectacle-prescrip = myope   AND astigmatism = no
   THAN  contact-lenses = none [0 0 2]

 <DEFAULT> contact-lenses = none [5 4 15]


Time taken to build model: 5.37 seconds
```

(a) CN2: Neurejen seznam odločitvenih pravil.

```
=== Classifier model (full training set) ===

CN2 rule induction algorithm.
Star size: 5
algorithm type: Ordered
Error estimate: laplacian
Discretiazion Type: default
INDUCED RULES:
IF    tear-prod-rate = reduced
   THAN  contact-lenses = none [0 0 12]
ELSE
IF    astigmatism = no
   THAN  contact-lenses = soft [5 0 1]
ELSE
IF    spectacle-prescrip = myope
   THAN  contact-lenses = hard [0 3 0]
ELSE
IF    age = young
   THAN  contact-lenses = hard [0 1 0]
ELSE
IF    age = pre-presbyopic
   THAN  contact-lenses = none [0 0 1]
ELSE
 <DEFAULT> contact-lenses = none [0 0 1]
```

(b) CN2: Urejen seznam odločitvenih pravil.

Slika 5.5: CN2: Neurejen (*unordered)* in urejen(*ordered*) seznam odločitvenih pravil.

odkrivanje istega pravila v naslednji iteraciji. Na Sliki 5.5b je upodobljen primer urejenega seznama, ki smo ga generirali z algoritmom CN2. Pomembna razlika tega seznama v primerjavi z neurejenimi pravili je, da moramo pri klasifikaciji novega primera urejen seznam preiskovati po vrsti, saj imamo sedaj opraviti z pravili oblike IF pogoj THEN razred, ELSE IF pogoj THEN razred ELSE IF .... ELSE DEFAULT. Slednje pri velikem številu pravil otežuje razumevanje, saj si pravila ne smemo razlagati ločeno od drugih, pač pa moramo upoštevati vsa pravila pred njim [5]. Dobra lastnost je, da se izognemo situaciji, ko bi nov primer pokrivalo več pravil.

Pomemben aspekt implementacije je bila tudi diskretizacija zveznih atributov, ki se izvaja med samo gradnjo pravil. Način diskretizacije na žalost ni opisan v članku Clarka in Nibbleta [6, 5], zato smo morali postopek sprogramirati s pomočjo analize in rekonstrukcije programske kode originalnih

avtorjev. Postopek diskretizacije prav gotovo vpliva na dobljena odločitvena pravila in je lahko pomemben dejavnik uspešnosti delovanja odločitvenih pravil, ki bi mu bilo po mnenju nekaterih potrebno posvetiti več pozornosti [14].

Na gradnjo odločitvenih pravil lahko gledamo kot na preiskovanje prostora. Za veliko težavo se je izkazala časovna kompleksnost algoritma CN2. V okviru nizkonivojskega iskanja najboljšega vozlišča je pri dodajanju novega pogoja v posamezno odločitveno pravilo namreč potrebno preveriti vse možne vrednosti za vse atribute. Ta kombinatorični problem postane še posebej očiten v primeru, ko smo soočeni z učnim problemom z velikim številom atributov. Med tovrstne visoko dimenzionalne podatke prav gotovo sodijo podatki mikromrež DNA. Časovno zahtevnost dodatno povečuje tudi dejstvo, da so atributi zvezne narave, zato je potrebno v procesu gradnje pravil iskati najboljše možne delitve. Posledično se je testiranje algoritma za oceno osnovne natančnosti s prečnim preverjanjem na vseh atributih genetskih podatkovnih zbirk s časovnega vidika izkazalo za nemogoče. Seveda počasnosti izvajanja prav gotovo ne moremo pripisati le kompleksnosti algoritma, pač pa je eden izmed ključnih faktorjev lahko tudi neoptimalna implementacija. Naše implementacije algoritma zato, po zaslugi izjemne počasnosti, nismo mogli uporabiti na vseh atributih ter na ta način ugotoviti njegove osnovne natančnosti. Uporabili smo ga lahko le na manjših podmnožicah atributov posameznega genskega seta, kjer je število atributov obvladljivo za pomanjkljivosti trenutne implementacije.

Rezultate, ki smo jih dobili z generiranjem neurejenih odločitvenih pravil CN2 na genskih setih, lahko razberemo iz tabele 5.7. V tabeli smo za primerjavo predstavili tudi rezultate modificirane metode naključnih podprostorov, ki smo jo predstavili v prejšnjem poglavju. Za orientacijo je podana tudi klasifikacijska točnost originalne metode naključnih podprostorov. Pri tem smo odločitvena pravila gradili na rangiranih genskih setih GSEA. Uporabili smo različne odstotke najboljše ocenjenih setov.

Odstotek naključnih genskih setov	Metoda		
Odstotek uporabljenih genskih setov	RS+REPTree	CN2 genski seti	Osnovna metoda
0.1%	78,53	77,04	83,96
1%	82,50	79,16	83,96
10%	81,61	79,39	83,96
50%	81,44	78,94	83,96

Tabela 5.7: Primerjava med klasifikacijsko točnostjo odločitvenih pravil CN2, zgrajenih na različnem odstotku GSEA razvrščenih genskih setov; klasifikacijsko točnostjo modificirane metode naključnih podprostorov, zgrajenih na istih genskih setih in klasifikacijsko točnostjo osnovne metode.

Opazimo lahko, da v smislu natančnosti, naš poskus generiranja odločitvenih pravil v primerjavi z ostalima metodama, precej konstantno izkazuje nižjo uspešnost. Zakaj je temu tako je težko odgovoriti. Eden izmed razlogov je, poleg tega da drevesne strukture morda delujejo bolje, tudi morebitna napaka v naši implementaciji. Naša programska koda je namreč, v veliki meri po zaslugi ne najbolj posrečenega načrtovanja, postala precej kompleksna. Slednje je postalo še posebej očitno pri implementaciji diskretizacije zveznih atributov. Ali naša implementacija resnično vsebuje napake in kako resne so te morebitne napake, pa na žalost ne moremo natančno odgovoriti, saj nismo pripravili ustreznih testov, s katerimi bi lahko potrdili ali ovrgli naše sume.

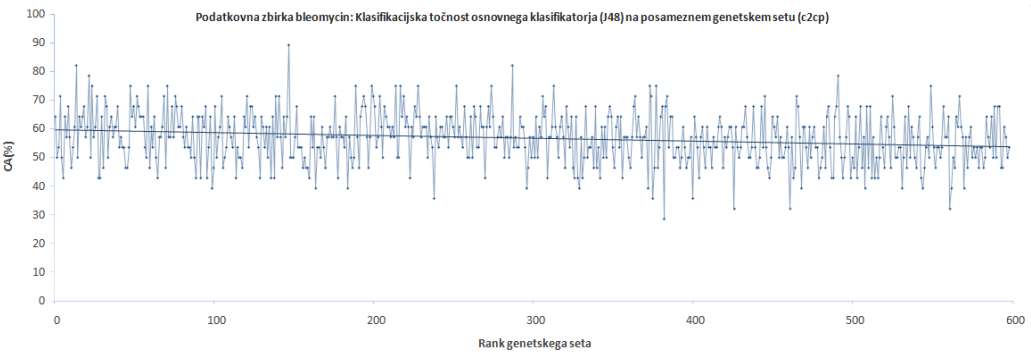
5.7 Analiza rezultatov

5.7.1 Smiselnost uporabe rangiranih genskih setov

V prejšnjem delu poglavja smo predstavili različne možne modifikacije metode naključnih podprostorov zgrajenih na smiselnih skupinah genov. Poskusili smo z gradnjo osnovnih klasifikatorjev na naključnih in na urejenih genskih setih ter predstavili metodo gradnje odločitvenih pravil na genskih

setih. V primerjavi z osnovno metodo nobena izmed implementiranih metod na žalost ni dosegla večje uspešnosti. Posledično smo za te primerjalno gledano slabše performanse poskušali poiskati možne razloge.

Eden izmed možnih razlogov za slabšo uspešnost je lahko neinformativnost osnovnih klasifikatorjev zgrajenih na določenih genskih setih. Z namenom, da bi ugotovili, kako dobro se posamezni osnovni klasifikator obnaša na posameznem genskem setu, smo poskusili zgraditi osnovne klasifikatorje le na eni podatkovni zbirki. Pri tem pa smo spremljali, kaj se dogaja z natančnostjo posameznega osnovnega klasifikatorja. Kot osnovni klasifikator smo uporabili odločitveno drevo J48. Vsakega izmed dreves smo zgradili na posameznem genskem setu iz rangiranega seznama genskih setov GSEA, ter ocenili njegovo natančnost s prečnim preverjanjem. Rezultati so prikazani na Sliki 5.6.



Slika 5.6: Klasifikacijska točnost osnovnega klasifikatorja (*J48*) zgrajenega na atributih posameznega genskega seta v odvisnosti od doseženega ranga genskega seta.

Opazimo lahko, da ne glede na dosežen rang, ki ga določenemu genskemu setu pripisuje GSEA, klasifikacijska točnost ves čas niha. Na grafu je poleg natančnosti posameznega klasifikatorja, prikazana tudi trendna črta. Opazimo lahko, da tudi za klasifikatorje zgrajene na slabše ocenjenih genskih setih, trend klasifikacijske točnosti pravzaprav ne pada, pač pa ostaja bolj

ali manj enak. Ta poskus smo opravili tudi na ostalih podatkovnih zbirkah in ugotovili, da kažejo identične simptome. Seveda smo ta poskus opravili le za en možen osnovni algoritem (odločitveno drevo J48), zato rezultatov verjetno ne moremo zadovoljivo posplošiti in zagotovo trditi, da to velja tudi za vse ostale osnovne algoritme. Kljub temu, da opažanja ne moremo v zadostni meri posplošiti, pa nam ta rezultat daje slutiti, da združevanje s preprostim uteževanjem glasov osnovnih klasifikatorjev glede na rang posameznega genskega seta, verjetno ne bi prineslo boljših rezultatov. Točnost osnovnih klasifikatorjev namreč niha brez kakšnega posebnega trenda, ne glede na dosežen rang. V luči teh rezultatov se zastavi tudi vprašanje, ali je glede na nihajočo klasifikacijsko točnost, sploh smiselno uporabljati tehniko rangiranja genskih setov s pomočjo analize GSEA in ali ni morebiti boljše uporabiti kar naključnih genskih setov. Še posebej, ker je zadnja opcija veliko enostavnejša in se glede na rezultate, ki smo jih predstavili v enem izmed prejšnjih delov empirične raziskave (podpoglavje: 5.5.2), po uspešnosti pri večjem številu uporabljenih genskih setov odreže povsem primerljivo.

5.7.2 Slaba uspešnost osnovnih klasifikatorjev

V nadaljevanju nas je zanimalo, ali imajo na genskih setih zgrajeni osnovni modeli, poleg brez posebnega trenda nihajoče točnosti, morebiti še kakšno drugo težavo. Tudi originalna metoda naključnih prodprostorov se namreč prav gotovo sooča z istim problemom, ki je posledica naključne izbire atributov za gradnjo osnovnih modelov. Naključna izbira podmnožic atributov spominja na loterijo in metoda posledično včasih izbere boljše, včasih pa slabše nabore atributov, na katerih lahko zgradi različno uspešne osnovne modele. Nihajoča točnost zato najbrž ne more v celoti pojasniti, zakaj se zdi, da je naključna izbira kljub elementu naključnosti, uspešnejša od predlagane metode informirane izbire smiselno povezanih atributov.

Z namenom, da bi našli odgovor na to vprašanje, smo se osredotočili na povprečno natančnost osnovnih modelov zgrajenih na naključnih podmnožicah atributov in jo primerjali s povprečno natančnostjo modelov zgrajenih na smiselno povezanih skupinah genov. Kot osnovni algoritem smo izbrali odločitveno drevo J48. Najprej smo za različno velik delež vseh atributov (1%, 10%, 50%) naključno izbirali podmnožice atributov pripadajočih velikosti in na njih gradili osnovne modele. Poskus smo ponovili 200-krat za vsak odstotek atributov in za vsako uporabljeno podatkovno zbirko. Pri tem smo merili klasifikacijsko točnost vsakega izmed zgrajenih modelov (odločitveno drevo). Postopek smo izvedli na vseh podatkovnih zbirkah in na koncu izračunali povprečno klasifikacijsko točnost za posamezno velikost nabora naključno izbranih atributov. V naslednji fazi smo odločitvena drevesa zgradili še na vseh genskih setih. Pri tem smo prav tako spremljali klasifikacijsko točnost posameznega drevesa in na koncu izračunali povprečno točnost za posamezno podatkovno zbirko. Rezultati so povzeti v tabeli 5.8.

Podatkovna Zbirka	J48(1%)	J48(10%)	J48(50%)	J48(Genetski seti)
bleomycin	58,4	60,3	63,2 *	56,6
breastColon	80,7	84,3*	81,7	76,4
dlbcl	77,7	80,9*	76,8	75,6
leukemia	78,1	85,7*	84,3	72,2
luminalBasalTumor	84,7	90,0	93,8*	75,3
metastasis	62,0	64,7	60,2	68,5*
prostata	70,2	77,8	80,8*	67,4
responderNonResponder	63,3	61,3	55,0	75,3*
tumorAdjacentNormal	91,5	92,7	94,5*	85,9
tumorRecurrence	56,1*	55,0	55,8	55,1
Skupno povprečje	72,3	75,3*	74,6	70,8

Tabela 5.8: Primerjava klasifikacijske točnosti odločitvenega drevesa J48 zgrajenega na različnem odstotku naključno izbranih atributov (1%, 10%, 50%) s povprečno klasifikacijsko točnostjo J48 zgrajenega na vseh genskih setih (C2CP) po posameznih podatkovnih zbirkah. Najvišje dosežene vrednosti so označene z zvezdico.

Na podlagi rezultatov predstavljenih v tabeli 5.8 lahko ugotovimo, da za vse, razen dveh podatkovnih zbirk, odločitveno drevo zgrajeno na genskih setih dosega slabšo povprečno uspešnost, kot odločitveno drevo zgrajeno na katerikoli izmed različno velikih naključno izbranih podmnožic atributov. Na podlagi tega lahko morda sklepamo, da je v povprečju morebiti boljše, če za gradnjo osnovnega modela izberemo atribute naključno, kot če uporabljamo genske sete. Poleg tega lahko v tem opažanju, da se naključno izbiranje obnese boljše, vidimo tudi razlog, zakaj se vse naše modificirane metode gradnje skupinskih klasifikatorjev odrežejo slabše od originalne metode. Možna razlaga je, da zaradi v povprečju slabših osnovnih modelov, tudi združevanje posledično daje slabše rezultate. Vkolikor je to res, potem se lahko resno vprašamo o smiselnosti uporabe genskih setov in morda pridemo tudi do zaključka, da smo se celotnega postopka morebiti lotili v napačnem vrstnem redu. Razvili smo namreč kar nekaj različnih metod, ki vse temeljijo na gradnji osnovnih modelov na genskih setih in za vse razvite metode kar naprej ugotavljali, da niso uspešnejše od osnovne metode. Ob predpostavki, da so naše modificirane metode manj uspešne, ker so povprečno manj uspešni naši osnovni modeli zgrajeni na genskih setih, bi bilo morda bolj racionalno že v začetku izmeriti uspešnost osnovnih modelov. Na podlagi tega bi morda lahko že vnaprej okvirno sklepali kakšne rezultate lahko pričakujemo.

5.8 Naključni podprostori v kombinaciji z metodo skladanja

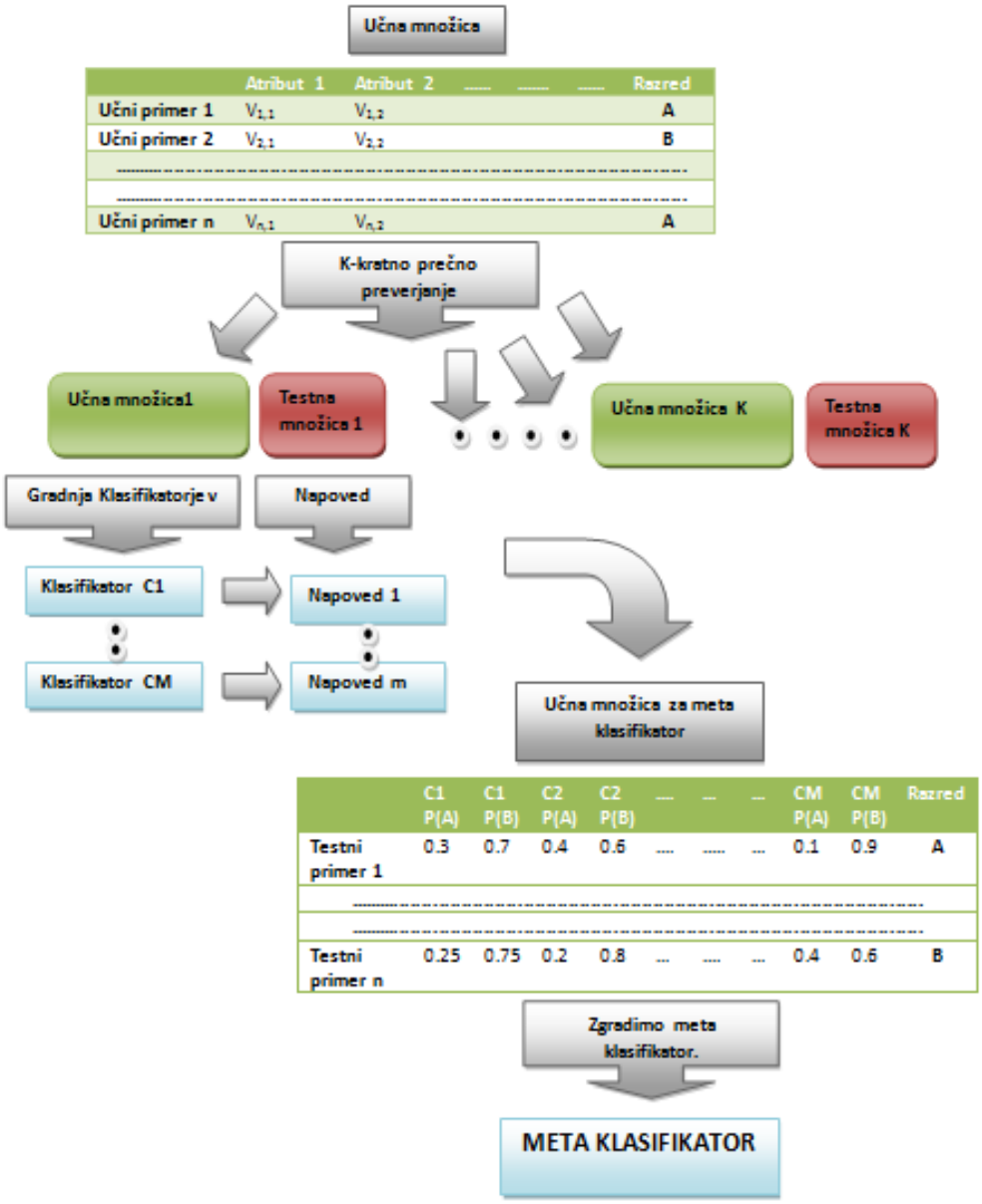
Predstavljena metoda združevanja klasifikatorjev, kjer smo osnovne klasifikatorje sprva zgradili na naključnih in kasneje na rangiranih geskih setih, se ni odrezala, kot smo se nadejali. Da bi izboljšali uspešnost naše skupinske metode, smo iskali možne poti kako se približati temu cilju. Kot ena izmed morebitnih rešitev se je ponudila možnost, da bi namesto preproste večinske

glasovalne sheme, uporabili bolj prefinjen način združevanja.

Namesto večinskega glasovanja smo želeli uporabiti bolj pameten način združevanja in na ta način nasloviti v prejšnjem delu predstavljena problema nihajoče klasifikacijske točnosti in nenatančnosti osnovnih modelov (podpoglavje: 5.7). Cilj drugačnega načina združevanja je zmanjšati vpliv slabših genskih setov in pri glasovanju uporabiti le bolj natančne osnovne klasifikatorje, zgrajene na tistih genskih setih, ki nosijo veliko informacije ter so pomembni za razločevanje med posameznimi razredi primerov v učni množici. V ta namen smo se s pomočjo metode skladanja klasifikatorjev (*stacking*) [41] na meta nivoju želeli naučiti, kateri so tisti genski seti oziroma modeli zgrajeni na atributih takega genskega seta, ki ne dajejo rezultatov. Slabše klasifikatorje bi s pomočjo klasifikatorja na meta nivoju v končni napovedi želeli ignorirati. Skladanje klasifikatorjev je metoda za združevanje osnovnih modelov. Glavna značilnost te metode je uporaba višje nivojskega modela za združevanje nižje nivojskih modelov, s katero lahko dosežemo večjo klasifikacijsko uspešnost [40].

Skladanje klasifikatorjev smo na kratko predstavili že v poglavju o skupinskih metodah strojnega učenja (podpoglavje: 3.2.6). V nadaljevanju bomo opisali tudi delovanje naše metode skladanja klasifikatorjev in njeno implementacijo. Metodo smo implementirali v programskem jeziku java in jo vključili v že omenjeno orodje za strojno učenje Weka. Naša implementacija temelji na modifikaciji osnovne variante variante skladanja klasifikatorjev (*stacking* [41]). Za razliko od osnovne variante modificirana metoda, uporablja verjetnostno distribucijo, ki sta jo predlagala Ting in Witten [40]. Prednost uporabe verjetnostne distribucije je, da poleg napovedi poskušamo uporabiti tudi zanesljivost, ki jo posamezni osnovni klasifikator pripisuje svoji odločitvi.

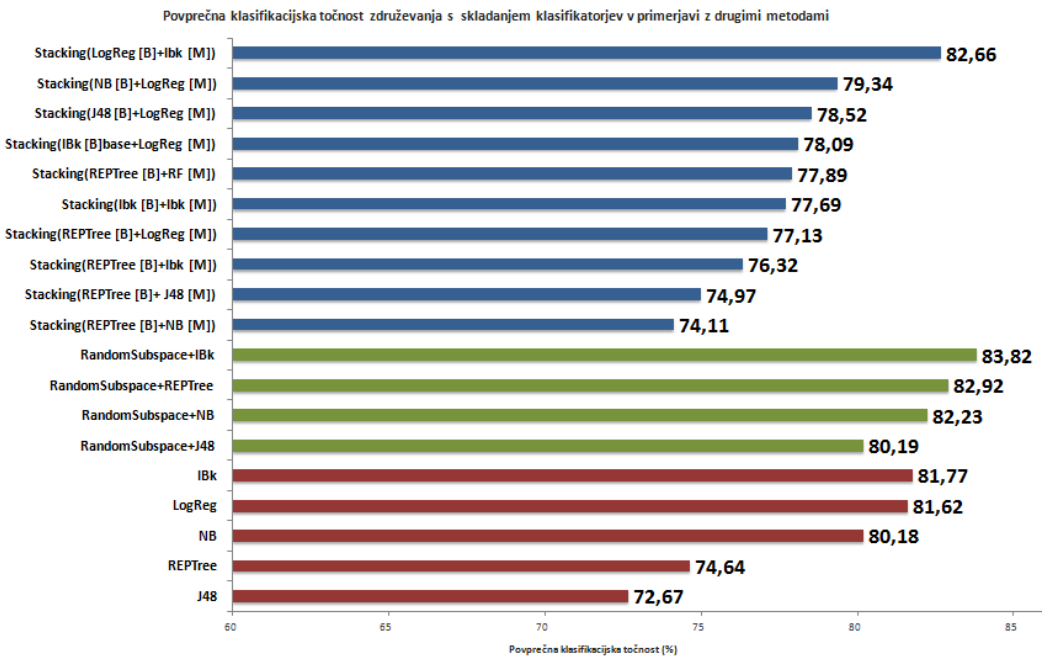
Postopek delovanja skladanja klasifikatorjev je prikazan na Sliki 5.7. Vhod v algoritem je učna množica. Pri grajenju klasifikatorja najprej generiramo



Slika 5.7: Metoda skladanja klasifikatojev (*stacking*).

podatke, na katerih se bo učil klasifikator na meta nivoju. Da bi pridobili podatke, ki jih bo lahko uporabil meta klasifikator za svoje učenje, moramo najprej zgraditi osnovne klasifikatorje in zabeležiti njihovo napoved. V ta namen potrebujemo testne množice, ki jih bomo uporabili za generiranje teh napovedi. Testne množice dobimo tako, da izvedemo notranji postopek prečnega preverjanja na naši učni množici. Faza prečnega preverjanja je nujna, saj potrebujemo interne testne množice na katerih bomo preizkusili natančnost posameznega osnovnega klasifikatorja. V posamezni fazi prečnega preverjanja na novi učni množici za vsakega izmed osnovnih klasifikatorjev zgradimo klasifikator na podmnožici atributov, ki ustrezajo posameznemu genskemu setu. Osnovni klasifikacijski algoritem je enak za vse osnovne klasifikatorje, le atributi so različni. Po končanem prečnem preverjanju dobimo novo učno množico z novimi atributi. Atributi te nove učne množice so verjetnostne napovedi (verjetnostna distribucija) posameznega klasifikatorja. Ker rešujemo dvorazredni problem, v našem konkretnem primeru dobimo dva nova atributa za vsakega izmed osnovnih klasifikatorjev. Prvi atribut je verjetnost, ki jo klasifikator pripiše prvemu razredu, drugi pa verjetnost za drugi razred. V našem konkretnem problemu dvorazredne klasifikacije bi bilo sicer dovolj, če bi uporabili le enega, saj sta atributa poplnoma soodvisna, vendar je to posledica implementacije algoritma, ki je primeren tudi za reševanje večrazrednih klasifikacijskih problemov. Primeri nove učne so torej napovedi za posamezen primer iz testnih množic, ki smo jih generirali med prečnim preverjanjem. Končni razred posameznega novega primera je pravi razred posameznega testnega primera. Na tej množici zgradimo naš meta klasifikator. V zaključni fazi gradnje celotnega klasifikatorja ponovno zgradimo osnovne klasifikatorje, tokrat na celotni množici, ki je bila prvotni vhod v naš učni algoritem.

Pri klasifikaciji novega testnega primera najprej novo instanco posredujemo vsakemu izmed osnovnih klasifikatorjev, ki nam vrnejo verjetnostne napovedi. Te verjetnostne napovedi nato v obliki novih atributov posredu-



Slika 5.8: Povprečna klasifikacijska točnost metode skladanja osnovnih klasifikatojev (*stacking*) zgrajenih na genskih setih, ob uporabi različnih algoritmov na osnovnem [B] in meta nivoju [M] (modra), v primerjavi z metodo naključnih podprostorov (zelena) in nekaterimi nezdruževalnimi algoritmi (rdeča).

jemo klasifikatorju na meta nivoju. Klasifikator na meta nivoju je zadolžen za končno napoved razreda.

Dosežene rezultate uporabe skladanja klasifikatorjev zgrajenih na rangiranih genskih setih (modra), izražene v smislu povprečne klasifikacijske točnosti, v primerjavi z nezdruževalnimi metodami (rdeča) in skupinsko metodo naključnih podprostorov z različnimi osnovnimi algoritmi (zelena), prikazuje graf na Sliki 5.8. Pri tem smo pri skladanju klasifikatorjev poskusili uporabiti nekaj različnih algoritmov, tako na osnovnem kot tudi na meta nivoju. Iz grafa je precej hitro razvidno, da se z izjemo ene same kombinacije osnovnega in meta algoritma, metoda skladanja precej konsistentno obnaša slabše od originalne metode naključnih podprostorov in močno teži k temu,

da dosega celo slabše rezultate od uporabe nekaterih nezdruževalnih metod (IBk, NB, LogReg).

Logično vprašanje, ki se poraja ob pogledu na dosežene rezultate, je, kako pojasniti to precej očitno degradacijo v performansah. Čeprav je na prvi pogled iz dobljenih vrednosti težko izluščiti kakšno zakonitost, ki bi pojasnila vpliv medsebojne interakcije osnovnega in meta algoritma na skupne rezultate obeh, pa se včasih zazdi, da se uspešnost spusti, oziroma vsaj poskuša približati, ravni uspešnost osnovnih klasifikatorjev (primer stackinga z REPTree na osnovnem nivoju vs. REPTree sam po sebi). Precej pozitivno odstopa kombinacija NB in IBk. Ta primerjalno gledano presenetljivo dober rezultat, pa izgubi nekaj svojega sijaja, če upoštevamo, da se ti dve metodi že v svoji osnovni varianti, sami po sebi, obnašata precej dobro. V primeru, da ju združimo, lahko dobimo le za odtenek boljši rezultat, kot če bi uporabili kar vsakega posamezno s privzetimi nastavitvami in brez kakršnekoli selekcije atributov.

Za trenutek izhajajmo iz pesimistične predpostavke, da si skladanje klasifikatorjev, na način, kot smo si ga zamislili in implementirali, prizadeva doseči padec natančnosti na nivo osnovnega klasifikacijskega algoritma, ki smo ga vključili v skladenjsko shemo. Pri tem seveda poskušamo domnevati tudi, da nismo zagrešili kakšne resne napake pri sami implementaciji oz. napačno interpretirali rezultatov. Ob teh predpostavkah je ta padec natančnosti, ne samo velik argument proti uporabi te metode v primerjavi z ostalimi skupinskimi metodami, pač pa tudi precej potraten način za doseganje ne preveč dobrih rezultatov, če upoštevamo precej izdatno količino dodatnega dela, ki ga je potrebno vložiti v cel postopek (gradnja velikega števila osnovnih modelov, rangiranje genskih setov, gradnja meta modela s prečnim preverjanjem).

Eden izmed možnih razlogov, zakaj skladanje istovrstnih klasifikatorjev v trenutni implementaciji ne prinaša želenih rezultatov, je morda možno lo-

cirati v sami teoretični definiciji skladanja klasifikatorjev. Wolpretov [41] koncept stackinga je sicer široko definiran in dopušča tudi možnost uporabe istega algoritma, vendar pa tipične opredelitve stackinga (napr. [13]) navadno vključujejo uporabo heterogenih algoritmov na osnovnem nivoju. Pri tem pa kot eno izmed glavnih konkurenčnih prednosti skladanja heterogenih algoritmov navajajo dejstvo, da za reševanje učnega problema nismo več prisiljeni iskati in izbirati najbolj primernega algoritma za gradnjo modela na določeni problemski domeni, saj lahko za reševanje učnega problema hkrati uporabimo več različnih modelov, zgrajenih z različnimi algoritmi [41]. Na ta način lahko izkoristimo prednosti in kompenziramo slabosti posameznega algoritma na določeni domeni. Poleg tega se večina avtorjev, ki se ukvarjajo s področjem skladanja klasifikatorjev, običajno ne ukvarja z vprašanjem, kako na ta način kombinirati istovrstne algoritme, pač pa bolj posvečajo pozornost vprašanjem, kot so: kateri algoritem uporabiti na meta nivoju ([40, 42]; katere atribute uporabiti za učenje klasifikatorja na meta nivoju ([40, 13]); in ali ni morebiti včasih s stališča učinkovitosti, najbolje uporabiti kar najboljšega algoritma (v smislu natančnosti), ki ga imamo trenutno v naši skladenjski shemi (*select best*) [42].

Če bolj podrobno pogledamo, kakšen je pravzaprav praktičen rezultat, ki ga dobimo s pomočjo skladanja klasifikatorjev v našem konkretnem primeru in poskušamo ugotoviti, kje bi lahko tičal problem skladanja velikega števila istovrstnih klasifikatorjev, potem se zazdi, da naša trenutna shema morda vsebuje resno pomanjkljivost. Recimo, da imamo prvotno 50 osnovnih modelov, naš model pa potem na meta nivoju iz verjetnostnih napovedi teh 50 osnovnih modelov, zgradi meta klasifikator (na primer odločitveno drevo). S pomočjo tega meta klasifikatorja se pri klasifikaciji novih primerov odloča, napovedi katerih osnovnih klasifikatorjev bo upošteval. To odločitveno drevo na meta nivoju ima lahko le nekaj vozlišč in zato upošteva le napovedi majhnega števila osnovnih klasifikatorjev. Z uporabo tega meta modela smo torej povzročili, da se na koncu upošteva glas zelo majhnega števila osnovnih kla-

sifikatorjev. Posledično pa to pomeni, da smo skupinsko metodo, pravzaprav zreducirali na nezdruževalno. Skupinsko metodo, ki temelji na številčnosti in pluralnosti glasov, smo s tega zornega kota, prikrajšali za osnovno bistvo njene uspešnosti in jo transformirali v model, ki se po obnašanju precej približa osnovnemu klasifikatorju. Slednje pa morda lahko pojasni slabe rezultate in odgovori na vprašanje, zakaj se naš postopek skladanja ne obnese.

Ena izmed možnosti za korekcijo omenjenega problema bi bil morda drugače zasnovan postopek skladanja klasifikatorjev. Na področju skladanja klasifikatorjev obstaja široka paleta različnih metod, od katerih bi se morebiti splačalo več pozornosti nameniti *gradingu* [36]. Grading je metoda za skladanje klasifikatorjev, ki poskuša zaznati in popraviti napovedi osnovnih klasifikatorjev. Pri tem pa za vsak osnovni klasifikator, zgradi svoj metaklasfikator, katerega naloga je da napove, kdaj se bo osnovni klasifikator zmotil. Poleg uporabe drugačne metode skladanja obstaja tudi veliko manevrskega prostora za izboljšanje na področju atributov, ki jih uporablja meta klasifikator za svoje učenje, saj je možno, da verjetnostne napovedi osnovnih klasifikatorjev, ki jih trenutno uporabljamo, niso najboljša izbira.

V naslednjem koraku smo v iskanju boljših rezultatov poskusili še z implementacijo prej omenjene metode *grading* [36] in za vsak osnovni klasifikator zgradili še klasifikator na meta nivoju. Temu ustrezno se je povečala kompleksnost celotnega postopka. V primerjavi s prvotno metodo skladanja klasifikatorjev (*stacking*) so doseženi rezultati rahlo boljši, vendar je razlika tako minorna, da o njih niti ni vredno poročati. Rezultati namreč še vedno nakazujejo na trend, da bi bilo v večini primerov boljše uporabiti le en sam nezdruževalni model s privzetimi nastavitvami.

Odgovor na vprašanje, zakaj je temu tako, smo poskusili poiskati v teoriji skladanja klasifikatorjev. V procesu iskanja možnih odgovorov, se zazdi, da pravzaprav ni nekega formalnega dokaza, zakaj (če sploh) ta postopek strojnega učenja na meta nivoju deluje. Na področju skladanja klasifika-

torjev obstaja namreč kar nekaj različnih raziskav, ki včasih prihajajo do različnih zaključkov. Posledica česa so te nasprotujoče ugotovitve o dejanski uspešnosti skladanja klasifikatorjev je težko ugotoviti. Eden izmed možnih razlogov je prav gotovo uporabljena metodologija. Poleg različnih raziskav na temo (ne)uspešnosti skladanja klasifikatorjev, tudi sam utemeljitelj te metode Wolpret [41] v okviru hevrističnih opažanj izpostavi, da se pod določenimi pogoji skladenjska shema, lahko obnaša slabše, kot če bi uporabili zgolj napovedi osnovnega klasifikatorja samega po sebi. Poleg tega opozarja, da osnovni klasifikatorji ne bi smeli biti med seboj preveč korelirani (podobno kot se to zahteva v primeru skupinskega učenja) in naj bi bili čim bolj ortogonalni. Izpostavlja pa tudi potrebo po tem, da naj bi bili osnovni klasifikatorji načeloma čim bolj drugačni oziroma različnih tipov (≫*span the space*≪) t.j. omogočali naj bi pogled na učno množico z različnih zornih kotov (različna pristranskost oz. *bias*) in poudarja željo po dodani informaciji (in ne po podvajanju informacije). S tega zornega kota morda strategija uporabe istega algoritma za gradnjo osnovnih modelov ni najbolj smiselna, saj naši osnovni klasifikatorji niso ortogonalni, imajo vsi isto pristranskost in morda niti ne znajo pogledati učne množice z različnih zornih kotov. Slednje nas je pripeljalo do zaključka, da skladanje klasifikatorjev morda ni pravo sredstvo, ki bi ga potrebovali za dosego cilja večje uspešnosti predlagane metode izbiranja atributov z genskimi seti.

5.9 Kratek povzetek rezultatov in glavne ugotovitve

V tem sklepnem delu diplomske naloge bomo na kratko povzeli dosežene rezultate opravljenih poskusov ter poskušali strniti nekatere najpomembnejše ugotovitve. Prav tako bomo poskušali opozoriti tudi na določena odprta vprašanja, ki jih predstavljena empirična raziskava, pušča neodgovorjena.

V uvodnem delu empiričnega dela smo začeli s kratko predstavitvijo izvedbe analize GSEA, s pomočjo katere lahko razvrstimo genske sete glede na pomembnost za ločevanje med ciljnima razredoma (podpoglavje: 5.1). Nato smo poskušali oceniti osnovno natančnost metod strojnega učenja na uporabljenih genetskih podatkovnih zbirkah (podpoglavje: 5.2). V okviru tega smo ugotovili, da so skupinske metode, v skladu s pričakovanji, res uspešnejše od nezdruževalnih, vendar nas je presenetilo opažanje, da se tudi nekatere nezdruževalne metode obnašajo precej primerljivo na primer logistična regresija in metoda najbližjih sosedov. V luči dobrih rezultatov teh dveh metod se nam je zdelo zanimivo, da kljub temu, da smo za gradnjo modela uporabili vse atribute, ne pride do t.i. prekletstva dimenzionalnosti (*curse of dimensionality*). Odločitvena pravila in drevesa se na naših podatkih na splošno obnašajo najslabše, kar je morebiti posledica prevelikega števila atributov. V primeru odločitvenih dreves lahko slabe rezultate pripišemo morda tudi dejstvu, da je učnih primerov zelo malo in vkolikor je za reševanje problema pomembnih več atributov drevesa zaradi omejitev, ki izhajajo iz zakonitosti gradnje modela, ne morejo upoštevati vseh, saj jim prehitro zmanjka učnih primerov.

V naslednjem koraku smo preučevali vpliv izbiranja atributov na uspešnost metod stojnega učenja (podpoglavje: 5.3). Pri tem smo najprej naleteli na praktično vprašanje pravilne izvedbe postopka prečnega preverjanja. Zanimalo nas je, kakšen učinek ima s stališča implementacije bistveno bolj preprost, vendar s stališča teorije napačno izveden postopek, pri katerem atribute ocenimo samo enkrat na celotni učni množici in se s tako ocenjenimi atributi lotimo prečnega preverjanja. Empirično smo ugotovili, da se učinek te napake v postopku odraža v preoptimistični oceni točnosti našega modela. To ugotovitev smo v nadaljevanju upoštevali pri pridobivanju vseh rezultatov in s pravilno izvedenim postom prečnega preverjanja ugotavljali vpliv izbiranja atributov na različne metode stojnega učenja. V okviru tega smo ugotovili, da ima izbiranje zelo velik pozitiven vpliv na nezdruževalne

metode. V primeru skupinskih metod pa je ta vpliv na uspešnost zanemarljiv oziroma, da v določenih primerih lahko izbiranje atributov deluje celo konktraproduktivno. V primeru nezdruževalnih metod je velik pozitiven vpliv selekcije kakovostne podmnožice atributov možno zaznati pri metodah odločitvenih dreves in pravil, kar lahko morda pripišemo temu, da so te vrste metod občutljive na slabe atribute. V odgovor na vprašanje, zakaj izbiranje atributov nima nobenega vidnega vpliva na skupinske metode (oziroma ima lahko celo nasproten učinek), smo identificirali dva možna vzroka. Prvi je, da lahko morda zanemarljiv vpliv izbiranja podmnožice kvalitetnih atributov za gradnjo skupinskega modela pripišemo temu, da z manjšim številom atributov posledično zmanjšamo tudi različnost ovnovih modelov. Druga možna razlaga je, da skupinske metode prikrajšamo za atribute, ki so na splošno gledano slabi, v kakšnih konkretnih situacijah pa znajo biti ravno ti atributi zelo koristni. Skupinske metode se z gradnjo velikega števila osnovnih modelov, lahko morda znajdejo tudi v takih situacijah in odkrijejo tudi takšne posrečene kombinacije. V naslednjem koraku smo primerjali, kako se v smislu uspešnosti pri razvrščanju atributov po pomembnosti, odrežeta metoda ReliefF in GSEA rangiran seznam atributov. Rezultati so pokazali, da je ReliefF povprečno uspešnejši. Z namenom, da bi pojasnili, zakaj metoda GSEA dosega slabšo uspešnost, smo primerjali odstotek enakih atributov med metodo GSEA in drugimi metodami za razvrščanje atributov in ugotovili, da ima GSEA morda več skupnega s t.i. kratkovidnimi ocenami, ki ne znajo upoštevati pogojne odvisnosti atributov.

Nato smo se osredotočili na metodo naključnih podprostorov (podpoglavje: 5.4). Zanimalo nas je, kako se metoda naključnih podprostorov s strategijo gradnje večih osnovnih modelov obnaša v primerjavi z uporabo osnovnega modela samostojno. Pri tem smo v skladu s predvidevanji ugotovili, da se ne glede na število izbranih atributov, skupinska metoda obnaša precej boljše kot osnovni model sam po sebi. Raziskovali smo tudi vpliv velikosti izbranih podmnožic atributov in izbire osnovnega algoritma in pričakovano

ugotovili, da imata lahko oba dejavnika določen vpliv na uspešnost skupinske metode. Opazili pa smo, da lahko morda metode, ki se primerjalno gledano samostojno obnašajo zelo slabo, pri uporabi v skupinski združevalni shemi dosegajo zelo dobre rezultate. Poleg tega smo se vprašali tudi o smiselnosti uporabe dvonivojskega združevanja, kjer namesto nezdruževalnih modelov, tudi na osnovnem nivoju uporabimo skupinske, saj dosežena uspešnost morda ne odtehta časovne kompleksnosti izvajanja.

V nadaljevanju smo se lotili gradnje skupinskega modela na genskih setih (podpoglavje: 5.5). Najprej smo osnovne modele gradili na naključnih skupinah genov. Pri tem smo poskusili uporabiti različno velik odstotek genskih setov iz različnih funkcionalnih skupin. Ugotovili smo, da funkcionalna skupina na uspešnost modela nima znatno velikega vpliva. Precejšen vpliv pa ima odstotek izbranih genskih setov, saj je metoda izrazito manj uspešna pri manjšem odstotku izbranih genskih setov. Slednje smo pripisali dejstvu, da pri manjšem odstotku izbranih skupin genov, neobhodno zgradimo tudi manj osnovnih modelov. Kljub temu, da je modificirana metoda z uporabo večjega odstotka genskih setov uspešnejša, pa ta naša modificirana metoda v nobenem primeru ni dosegla natančnosti osnovne metode naključnih podprostorov. V naslednjem koraku smo poskusili uporabiti razvrščene genske sete, ki smo jih po pomembnosti razvrstili z analitičnim orodjem GSEA. Prvotna domneva, da se bodo boljše razvrščeni genski seti odrezali boljše kot naključna izbira, se je izkazala za upravičeno. Še posebej je to vidno pri manjšem odstotku izbranih genskih setov, medtem ko se pri večjem številu ta vpliv zmanjšuje. Glede na to, da je tudi ta modifikacija, kljub večji uspešnosti v primerjavi s prejšnjo, še vedno slabša od osnovne variante modela s privzetimi nastavitvami, se je porodil dvom, ali je to ob upoštevanju količine vloženega dela za razvrščanje skupin genov in gradnjo takih modelov, to sploh smiselno početje.

V naslednjem koraku smo želeli ugotoviti, kako se obnaša predlagana metoda vodene gradnje skupinskih modelov na smiselnih skupinah genov, če

na osnovnem nivoju uporabimo odločitvena pravila zgrajena z algoritmom CN2 (podpoglavje 5.6). V ta namen smo implementirali algoritem CN2 ter ga vključili v orodje za strojno učenje Weka. Ugotovili smo, da je algoritem časovno zelo kompleksen, saj poleg postopka diskretizacije zveznih atributov, s taktiko iskanja pravil v snopu preišče velik del prostora, zato ga je bilo nemogoče uporabiti na vseh atributih naših podatkovnih zbirk. Zgradili pa smo lahko skupinski model, kjer so odločitvena pravila zgrajena na posameznih skupinah genov. Gradnja odločitvenih pravil na genskih setih je v primerjavi z originalno metodo naključnih podprostorov daleč počasnejša in tudi znatno manj uspešna. Posledično smo se začeli še bolj zavedati dragocenega pomena optimizacije, načrtovanja kode in ustreznega testiranja, saj ne moremo zagotovo trditi, ali je problem manjše uspešnosti samo v uporabi odločitvenih pravil, ali pa morda pri taki ogromni količini atributov in zgeneriranih pravil prihaja med izvajanjem programa do morebitnih resnih napak.

Odsotnost spodbudnih rezultatov uporabe genskih setov za gradnjo osnovnih modelov nas je pripeljala do tega, da smo se usmerili v iskanje možnih razlogov, zakaj je temu tako (podpoglavje: 5.7). V ta namen smo raziskali, kaj se pravzaprav dogaja z natančnostjo osnovnih modelov zgrajenih na skupinah genov. Ugotovili smo, da ne glede ne dosežen rang posameznega genskega seta, natančnost osnovnih modelov konstantno niha. Poleg tega v tem nihanju ni zaznati kakšnega posebnega trenda, ki bi nakazoval na to, da so boljše ocenjeni genski seti povprečno tudi bolj uspešni. Slednje spoznanje, ob upoštevanju časovne kompleksnosti izvedbe analize GSEA, poraja dvom v smiselnost in racionalnost rangiranja genskih setov. Poleg nihajoče točnosti smo identificirali še drug problem, ki tiči v povprečni natančnosti osnovnih modelov. V okviru tega smo ugotovili, da so modeli zgrajeni na naključno izbranih podmnožicah atributov v večini primerov povprečno bolj uspešni, kot so povprečno uspešni modeli, ki jih dobimo z uporabo genskih setov. Na ta način smo prišli do zaključka, da se morda naš temeljni problem, zakaj s svojimi modifikacijami ne moremo doseči niti natančnosti osnovnega mo-

dela, skriva v tem, da za gradnjo skupinskega modela uporabljamo povprečno manj uspešne osnovne modele .

V zadnjem koraku na ciljni ravnini smo ta problem poskusili nasloviti z še zornega kota meta učenja in uporabili metodo skladanja klasifikatorjev (podpoglavje: 5.8). V ta namen smo implementirali algoritem za skladanje klasifikatorjev (*stacking*), saj smo upali, da lahko morda z dodatnim modelom na meta nivoju ublažimo vpliv slabih osnovnih klasifikatorjev v naši skupinski združevalni shemi in posledično zgradimo uspešnejši skupinski model. Naša domneva se je žal izkazala za napačno. Ugotovili smo, da se skladanje v našem konkretnem primeru obnaša kontraproduktivno, saj se glede na doseženo natančnost, v primerjavi z drugimi metodami, v večini primerov odreže presenetljivo slabo in se precej približa natančnosti osnovnega algoritma s privzetimi nastavitvami. Za pojasnilo smo se obrnili k teoriji in ugotovili, da ja skladnje klasifikatorjev morda rahlo protislovno področje brez formalnega dokaza, ki bi predstavljal zagotovilo za njegov uspeh, posledično različne raziskave poročajo o različnih rezultatih. Poleg tega smo morda z združevanjem osnovnih modelov, ki so zgrajeni z istovrstnim algoritmom, kršili določene teoretične predpostavke, saj metoda skladanja običajno ni mišljena za združevanje istovrstnih algoritmov, pač pa različnih. Ena izmed glavnih idej, ki upravičuje uporabo te metode je namreč zmožnost poudariti prednosti in kompenzirati slabosti posamezne vključene učne metode, ki izhaja iz dejstva, da lahko pogledamo učno množico z zornega kota različnih učnih strategij, ki imajo različno pristranskost (*bias*).

Poglavje 6

Zaključek

V diplomskem delu smo raziskovali možno modifikacijo metode naključnih podprostorov. Glavna ideja je bila ugotoviti, kako se obnaša skupinski model, če namesto naključne izbire v postopek gradnje vpeljemo informirano izbiro in osnovne modele gradimo na smiselnih podmnožicah atributov. Za gradnjo osnovnih modelov smo uporabili skupine genov, ki jih definirajo genski seti. V želji, da bi dosegli kakršnekoli rezultate o katerih bi bilo vredno poročati, smo poskusili raziskati mnogo različnih poti, ki bi nas morebiti lahko pripeljale do tega cilja. Vendar pa se je cilj vztrajno izmikal, zato smo preprosto uživali v sami poti in se v tem procesu iskanja verjetno še največ naučili o tem, kaj potencialno ne deluje.

Če strnemo v prejšnjem poglavju predstavljene ugotovitve, bi morda lahko rekli, da se gradnja osnovnih modelov z informirano izbiro, ki jo predstavljajo genski seti, v našem konkretnem premeru in na naših konkretnih podatkih ni izkazala za uspešnejšo od originalne metode. Kljub temu, da smo poskušali identificirati možne razloge, ne moremo z gotovostjo zatrditi, da smo našli točen odgovor na vprašanje, zakaj modificirana metoda ne deluje. Poleg tega tudi ne moremo ponuditi dokazov na podlagi katerih bi lahko

zanesljivo sklepali, da informirana izbira podmnožic atributov na splošno ne daje rezultatov in da se je pri gradnji skupinskih modelov potrebno izogibati uporabi te strategije. Vse kar lahko ponudimo, je morda namig, da se ta strategija v našem primeru ni obnesla. Posledica česa so slabi doseženi rezultati je težko ogovoriti, saj obstaja več možnih razlogov. Eden izmed možnih razlogov je, da je za rezultate morda kriva uporabljena metodologija za merjenje rezultatov. Če bi želeli natančno odgovoriti na določena odprta vprašanja, bi morali veliko več premisleka nameniti sami zasnovi poskusov in načinu kako bomo primerjali rezultate posameznih poskusov med seboj, saj ne moremo ignorirati dejstva, da smo morda včasih med seboj primerjali hruške in jabolka (10% genskih setov pomeni napr. bistveno več atributov kot 10% atributov). Problematična je tudi sama uporaba klasifikacijske točnosti. Nasploh je področje merjenja in zagotavljana primerljivosti rezultatov na področju strojnega učenja verjetno zgodba zase.

Drug razlog je pomanjkanje znanja in razumevanja, da bi se lahko učinkovito spopadli s področjem skupinskega stojnega učenja. To področje je namreč samo po sebi svojevrsten izziv in si zagotovo zasluži temeljito pozornost. Po zaslugi svoje uspešnosti skupinsko učenje zbuja veliko zanimanja in je plodno področje za snovanje mnogih člankov na temo potencialnih izboljšav. Posledično je zasuto z velikim številom raziskav, ki občasno vodijo v protislovne rezultate in na ista vprašanja ponujajo drugačne odgovore. Tako se zdi, da na področju skupinskega učenja obstaja kar nekaj odprtih vprašanj in precej različnih razlag, odsotna pa je morda neka skupna krovna teorija. V odsotnosti univerzalne razlage se človek pogosto sreča s vprašanji kot so: kaj sploh deluje, kaj je prava razlaga, katero teorijo uporabiti ipd. in v vsej zmedi kaj hitro podvomi v lastno razumevanje določenih konceptov. Posledično obstaja možnost, da smo s pristopom informirane izbire podmnožic atributov, morda kršili kakšno pomembno teoretično predpostavko. Katero in zakaj, na žalost ne moremo natančno odgovoriti.

Tretji razlog je povezan z uporabljenimi podatki. Genetski podatki mi-

kročipov DNA niso ravno ena izmed »standardnih« (tipičnih) podatkovnih zbirk, ki se jih na področju stojnega učenja pogosto uporablja kot osnovo za primerjanje različnih učnih algoritmov. Genetski podatki izvirajo iz domene, ki jo zaradi pomanjkljivega poznavanja področja biologije in povezanih znanosti, ne moremo kompetentno raziskovati. Na račun boljšega poznavanja domene naših učnih podatkov, bi morda lahko dosegli bistveno več, vendar naš primarni cilj ni bil doseči najboljše možne rezultate (vsekakor ne za vsako ceno), pač pa raziskovanje samo. Poleg nepoznavanja uporabljenih genetskih podatkovnih zbirk, je morda popolnoma legitimno pojasnilo tudi, da je za slabe rezultate kriva sama lastnost uporabljenih podatkov. Dimenzionalnost podatkov je namreč neprimerno večja od števila učnih primerov. Posledično je težko na podlagi tako majhnega učnega vzorca karkoli zanesljivo sklepati.

Na vprašanje, zakaj naša modificirana metoda ni uspešnejša od osnovne variante, je torej težko zadovoljivo odgovoriti. Celoten proces našega raziskovanja lahko morda jedernato povzamemo kar z Einsteinovimi besedami: Če bi točno vedeli kaj delamo narobe, potem se temu ne bi reklo raziskovanje.

Literatura

[1] David W Aha, Dennis Kibler, and Marc K Albert. Instance-based learning algorithms. *Machine learning*, 6(1):37–66, 1991.

[2] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is "nearest neighbor" meaningful? In *Database Theory—ICDT'99*, pages 217–235. Springer, 1999.

[3] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[4] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] Peter Clark and Robin Boswell. Rule induction with cn2: Some recent improvements. In *Machine learning—EWSL-91*, pages 151–163. Springer, 1991.

[6] Peter Clark and Tim Niblett. The cn2 induction algorithm. *Machine learning*, 3(4):261–283, 1989.

[7] Jacqueline Cloos, Wim PH de Boer, Mireille HJ Snel, Paul van den IJssel, Bauke Ylstra, C René Leemans, Ruud H Brakenhoff, and Boudewijn JM Braakhuis. Microarray analysis of bleomycin-exposed lymphoblastoid cells for identifying cancer susceptibility genes. *Molecular cancer research*, 4(2):71–77, 2006.

[8] William W Cohen and Yoram Singer. A simple, fast, and effective rule learner. In *Proceedings of the National Conference on Artificial Intelligence*, pages 335–342. John Wiley & Sons Ltd, 1999.

[9] Marquis de Condorcet. Essay on the application of analysis to the probability of majority decisions. *Paris: Imprimerie Royale*, 1785.

[10] Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.

[11] Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.

[12] Alain Dupuy and Richard M Simon. Critical review of published microarray studies for cancer outcome and guidelines on statistical analysis and reporting. *Journal of the National Cancer Institute*, 99(2):147–157, 2007.

[13] Saso Džeroski and Bernard Ženko. Is combining classifiers with stacking better than selecting the best one? *Machine learning*, 54(3):255–273, 2004.

[14] Peter Flach and Nada Lavrač. *Rule induction*. Springer, 2003.

[15] Yoav Freund, Robert Schapire, and N Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

[16] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[17] Nicolás García-Pedrajas and Domingo Ortiz-Boyer. Boosting random subspace method. *Neural Networks*, 21(9):1344–1362, 2008.

[18] Todd R Golub, Donna K Slonim, Pablo Tamayo, Christine Huard, Michelle Gaasenbeek, Jill P Mesirov, Hilary Coller, Mignon L Loh, James R Downing, Mark A Caligiuri, et al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, 286(5439):531–537, 1999.

[19] Tin Kam Ho. Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278–282. IEEE, 1995.

[20] Tin Kam Ho. The random subspace method for constructing decision forests. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(8):832–844, 1998.

[21] Tin Kam Ho. A numerical example on the principles of stochastic discrimination. *arXiv preprint cs/0402021*, 2004.

[22] Onkološki inštitut Ljubljana. Sistemsko zdravljenje raka dojk. *PACIENTKE Z RAKOM DOJK–TRENDI IN NOVOSTI*, page 99.

[23] George H John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 338–345. Morgan Kaufmann Publishers Inc., 1995.

[24] P Juvan et al. Tehnologija dna mikromrež in njena uporaba v medicini. *11SDMI*, page 2.

[25] Branko Kavšek. *Odkrivanje podskupin z uporabo algoritmov za učenje pravil*. PhD thesis, Fakulteta za računalništvo in informatiko, 2004.

[26] EM Kleinberg. An overtraining-resistant stochastic modeling method for pattern recognition. *The annals of statistics*, 24(6):2319–2349, 1996.

[27] Eugene M Kleinberg. On the algorithmic implementation of stochastic discrimination. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22.

[28] Igor Kononenko. Estimating attributes: analysis and extensions of relief. In *Machine Learning: ECML-94*, pages 171–182. Springer, 1994.

[29] Igor Kononenko and Marko Robnik-Šikonja. *Inteligentni sistemi*. UL, Fakulteta za računalništvo in informatiko, 2010.

[30] Niels Landwehr, Mark Hall, and Eibe Frank. Logistic model trees. *Machine Learning*, 59(1-2):161–205, 2005.

[31] Bernhard Pfahringer, Geoffrey Holmes, and Cheng Wang. Millions of random rules. 2004.

[32] Vincent Pisetta. *New insights into decision tree ensembles*. PhD thesis, Université Lumière Lyon, Informatique et Mathématiques, 2012.

[33] John Ross Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[34] Marko Robnik-Šikonja. *Lastnosti in uporaba hevristične funkcije Relief v strojnem učenju*. PhD thesis, PhD thesis, University of Ljubljana, Faculty of Computer and Information Science, 2001.

[35] Robert E Schapire. A brief introduction to boosting. In *Ijcai*, volume 99, pages 1401–1406, 1999.

[36] Alexander K Seewald and Johannes Fürnkranz. An evaluation of grading classifiers. In *Advances in Intelligent Data Analysis*, pages 115–124. Springer, 2001.

[37] Margaret A Shipp, Ken N Ross, Pablo Tamayo, Andrew P Weng, Jeffery L Kutok, Ricardo CT Aguiar, Michelle Gaasenbeek, Michael Angelo, Michael Reich, Geraldine S Pinkus, et al. Diffuse large b-cell lym-

phoma outcome prediction by gene-expression profiling and supervised machine learning. *Nature medicine*, 8(1):68–74, 2002.

[38] Dinesh Singh, Phillip G Febbo, Kenneth Ross, Donald G Jackson, Judith Manola, Christine Ladd, Pablo Tamayo, Andrew A Renshaw, Anthony V D'Amico, Jerome P Richie, et al. Gene expression correlates of clinical prostate cancer behavior. *Cancer cell*, 1(2):203–209, 2002.

[39] Aravind Subramanian, Pablo Tamayo, Vamsi K Mootha, Sayan Mukherjee, Benjamin L Ebert, Michael A Gillette, Amanda Paulovich, Scott L Pomeroy, Todd R Golub, Eric S Lander, et al. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. *Proceedings of the National Academy of Sciences of the United States of America*, 102(43):15545–15550, 2005.

[40] Kai Ming Ting and Ian H Witten. Issues in stacked generalization. *arXiv preprint arXiv:1105.5466*, 2011.

[41] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[42] Bernard Zenko. *Izboljššave metode skladanja klasifikatorjev*. PhD thesis, Magistrsko delo, Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, 2003.