Sveučilište u Zagrebu

FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Marijo Šimunović

Detekcija intronskih i egzonskih sekvenci u molekuli DNK upotrebom stabala odlučivanja

SEMINARSKI RAD

Zagreb, 2018.

Sadržaj

Popis slika

Popis tablica

Uvod

Početak 21. stoljeća donio je sa sobom i kraj Projekta humanog genoma. Točno pedeset godina nakon što je otkrivena struktura DNK molekule[1], u travnju 2003. godine, Međunarodni konzorcij za sekvenciranje ljudskog genoma objavio je uspješan dovršetak sekvenciranja nešto više od tri milijarde nukleotidnih baza - koliko ih sadrži ljudski DNK. Pod okriljem ovog projekta sekvencirani su i genomi drugih eukariotskih organizama[1] te stotine vrsta bakterija i arheja. Sekvenciranje novih vrsta nastavljeno je i nakon dovršetka projekta[2].

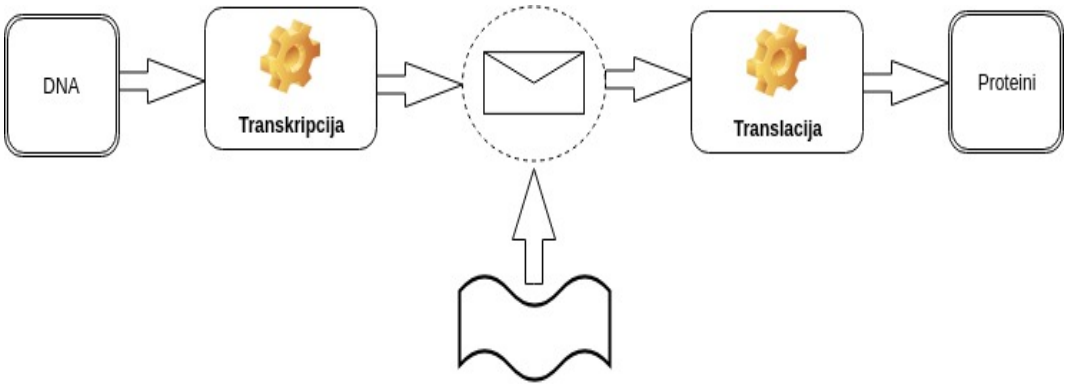
Budući da se radi o ogromnim količinama podataka, vrlo brzo postalo je jasno kako ih je moguće smisleno interpretirati samo uz pomoć odgovarajućeg sofvera. Kao posljedica toga, počelo se razvijati interdisciplinarno područje bioinformatike čiji je cilj kombiniranjem znanja iz područja molekularne biologije i računalnih znanosti te podataka o genomu dovesti do novih spoznaja o načinu na koji funkcionira živi svijet na Zemlji. Jedan od primjera primjene računalne znanosti, detekcija granica kodirajućih i nekodirajućih dijelova gena, obrađen je u ovom seminaru.

[1] Kvasac *Saccharomyces cerevisiae*, oblić *Caenorhabditis elegans*, vinska mušica *Drosophila melanogaster*, domaći miš *Mus musculus* i biljka *Arabidopsis thaliana*

1 Teorijski okvir

1.1 Središnja dogma molekularne biologije

U jednoj stanici organizma nalazi se informacija potrebna za razvoj cjelokupne jedinke. Kod većine organizama ta informacija kodirana je u molekuli koja se naziva deoksiribonukleinska kiselina (DNK) uz izuzetak nekih organizama koji za tu svrhu koriste ribonukleinsku kiselinu[2] (RNK). Tok genetskih informacija unutar biološkog sustava naziva se središnja dogma molekularne biologije. Slika 1 prikazuje analogiju središnje dogme i općeg komunikacijskog modela u računalnoj znanosti. Ribonukleinska kiselina (RNK) je polinukleotidna jednolančana molekula i nastaje transkripcijom komplementarnih baza gena iz molekule DNK. Ovaj se proces odvija pod utjecajem enzima RNK polimeraza koji prepoznaju karakteristično mjesto (promotor) na DNK molekuli gdje započinju proces prepisivanja korištenjem ribonukleotidnih molekula – adenin (A) se prepisuje u uracil (U), timin (T) u adenin, guanin (G) se prepisuje u citozin (C) te obratno, citozin u guanin. Transkripcija se nastavlja dok enzim ne stigne do karakterističnog niza zvanog terminator. Molekula RNK se prenosi iz jezgre stanice u citoplazmu. Na ribosomima u procesu translacije nastaju molekule proteina[3].

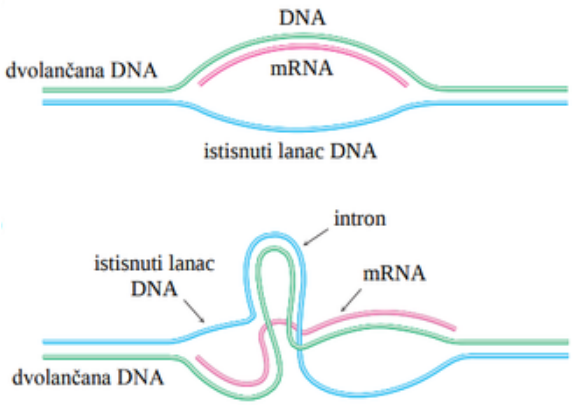


Slika 1: Središnja dogma molekularne biologije po uzoru na Shannonov [4] dijagram općeg komunikacijskog sustava. *DNK molekula je izvor informacije, a poruka sadrži uputu za sintezu proteina. Odašiljanje poruke je transkripcijski proces koji kodira poruku. Signal se prenosi prema prijemniku, ribosomima u obliku RNK molekule. U procesu translacije dekodira se poruka i nastaje molekula proteina. Šum označava greške u internim biološkim funkcijama stanice, ali i vanjske utjecaje koji mogu utjecati na ovaj proces.*

[2]Vjerojatno najpoznatiji primjer je HIV virus

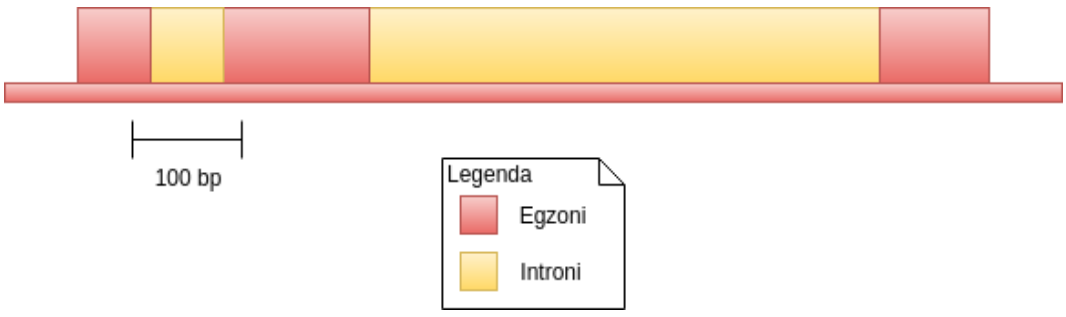
1.2 Izrezivanje RNK molekule

Elektronskim mikroskopom utvrđena je mozaična struktura gena (slika 2). Ovo je potvrđeno u eksperimentima zonskog centrifugiranja gdje je ustanovljena razlika sedimentacije primarnih transkripata i zrelih mRNK molekula [5].



Slika 2: Detekcija introna elektronskim mikroskopom. *Kod kontinuiranih gena vidi se samo jedna omča dvolančane DNK (gore). Ako gen sadržava introne, vide se dvije omče jednolančane DNK i omča dvolančane DNK (dolje). Crvenom bojom prikazana je mRNK. Preuzeto iz [5].*

Jednu DNK molekulu čini više različitih gena između koji se nalazi niz nekodirajućih nukleotidnih baza. Geni iz DNK se transkripcijom prepisuje u primarni transkript RNK. Unutar većine gena[3] nalaze se intervencijske sekvence (slika 3). Intervencijske sekvence nazivaju se introni.



Slika 3: Introni. *Struktura ljudskog β-globin gena. Ovaj gen dug je 1423 bazna para (bp) i sadrži dva intron, prvi dužine 131 bp i drugi dužine 851 bp što čini oko 69% dužine gena. Prilagođeno prema [3].*

[3]kod eukariotskih organizama

Introni ne kodiraju proteine nego se izrezuju iz primarnog transkripta. Zrela mRNK (engl. *messenger RNA*) molekula sastoji se samo od kodirajajućih dijelova nazvanih egzoni. Granica između egzona i introna naziva se donorsko mjesto (EI), a granica između introna i egzona akceptorsko mjesto (IE). Mehanizam obrade primarnog transkripta u kojem se izbacuju nekodirajući te spajaju kodirajući slijedovi baznih parova naziva se izrezivanje ili prekrajanje (engl. *splice*). Detalji mehanizma izrezivanja nadilaze opseg ovog rada i detaljno su opisani u [2, 3]. Međutim, jasna je motivacija za pronalaskom algoritma koji može detektirati ove lokacije u genu. Kada se sekvencira novi genom ili pronađe novi protein za koji nije poznat gen koji ga kodira ovakav algoritam može detektirati potencijalne kandidate i uvelike suziti izbor između milijuna, odnosno milijardi potencijalnih pozicija u genomu.

2 Opis razvojnog okruženja za dubinsku analizu

2.1 Skup podataka za dubinsku analizu

Skup podataka nad kojim je napravljena dubinska analiza potječe iz banke genoma (Genbank 61.1) i može se preuzeti sa repozitorija za strojno učenje UCI[4] u komprimiranoj datoteci. Skup sadrži 3190 instanci sa 63 atributa. Prvi atribut u tablici je klasa. Podaci pripadaju jednoj od tri kategorije:

- "IE" - slijed u genomu koji se nalazi na granici intron egzon
- "EI" - slijed u genomu koji se nalazi na granici egzon intron
- "N" - slijed u genomu za koji je poznato da ne sadrži granicu između egzona i introna

Drugi atribut je tekstualni identifikator instance. Preostali atributi su zapravo sekvenca šezdeset slova koje označavaju nukleotide sekvence u DNK molekuli za koju želimo utvrditi kojoj klasi pripada. U ovom radu se na pozicije pojedinačnih nukleotidnih baza (slova) referiramo pozitivnim indeksima 1, 2, ..., 60. Nukleotidi u skupu podataka su označeni na način prikazan Tablicom 1.

Tablica 1: Oznake nukleotida u skupu podataka za dubinsku analizu. *A, G, C i T se koriste ako se na toj lokaciji pojavljuje isključivo jedna od četiri baze. Ukoliko se na istom indeksu u sekvenci pojavljuju različite baze, uz sve ostale pozicije s jednakim nukleotidima koristimo oznake D, N, S i R.*

Oznaka atributa	Nukleotid
A	Adenin
T	Timin
G	Guanin
C	Citozin
D	Adenin ili Guanin ili Timin
N	Adenin ili Guanin ili Citozin ili Timin
S	Citozin ili Guanin
R	Adenin ili Guanin

[4] `https://archive.ics.uci.edu/ml/datasets/Molecular+Biology+(Splice-junction+Gene+Sequences)`

2.2 Programski alati za dubinsku analizu

Za obradu i preprocesiranje skupa podataka korišten je programski jezik *Python* (verzija 3.5) u kombinaciji sa bibliotekama *pandas*, *sklearn* te *matplotlib* i *seaborn* za grafički prikaz rezultata. *pandas* je biblioteka otvorenog koda koja implementira brze i fleksibilne strukture podataka kako bi se korisniku omogućilo što lakše i intuitivnije upravljanje podacima[6]. Ova biblioteka omogućuje korištenje različitih tipova podataka

- Tabularni podaci sa stupcima heterogenog tipa (kao u SQL ili Excel tablicama)
- Uređeni i neuređeni skupovi vremenskih podataka
- Proizvoljni podaci u matričnom obliku s oznakama redova i stupaca
- Bilo kakav drugi oblik statističkih skupova podataka koji ne mora nužno biti označen

scikit-learn je biblioteka otvorenog koda i sadrži skup algoritama za strojno učenje i dubinsku analizu podataka[7]. Obuhvaća širok spektar funkcionalnosti, od preprocesiranja podataka preko učenja različitih modela do evaluacije dobivenih rezultata. Skripta za obradu podataka napisana je u razvojnom okruženju *Jupyter Notebook* i dostupna je u *GitHub*[5] repozitoriju. Dubinska analiza provedena je korištenjem programskog alata Weka[8].

[5] `https://github.com/m5imunovic/CourseWork`

3 Predobrada podataka i izbor atributa

3.1 Predobrada

Za predobradu podataka korišten je programski jezik *Python* u kombinaciji s paketom *pandas*. Datoteka s podacima *splice_orig.csv* je učitana u *pandas DataFrame* objekt te su dodani nazivi atributa: *class*, *id* i *dna* (vidi tablicu 2).

Tablica 2: Primjeri instanci iz skupa podataka za dubinsku analizu.

class	id	dna
EI	ATRINS-DONOR-905	AGACCCGCCGGGAGGCGGAGGACCTGC...
EI	BABAPOE-DONOR-30	GAGGTGAAGGACGTCCTTCCCCAGGAG...
EI	CHPIGECA-DONOR-378	CAGACTGGGTGGACAACAAAACCTTCA...
..	..	..
N	ORAHBG2F-NEG-181	ATCAATAAGCTCCTAGTCCAGACGCCAT...
N	ORARGIT-NEG-241	TCTCGGGGGCGGCCGGCGCGGCGGGG...
N	TARHBD-NEG-1981	AGGCTGCCTATCAGAAGGTGGTGGCTG...

Nakon što smo izbacili stupac *id* i podijelili stupac *dna* podaci imaju oblik kao u tablici 3.

Tablica 3: Primjeri instanci iz procesiranog skupa podataka za dubinsku analizu. *Stupac id je jedinstveni identifikator vrste za svaki red u tablici i zbog toga ga ne koristimo u analizi podataka. Stupac DNA dijelimo na šezdeset stupaca, za svaki nukleotid u DNA nizu po jedan novi stupac, naziva dna_x gdje x označava indeks nukleotida u originalnom nizu.*

class	dna_1	dna_2	dna_3	dna_4	dna_5	...
EI	A	G	A	C	C	...
EI	G	A	G	G	T	...
EI	C	A	G	A	C	...
..	..	..	..	..	..	...
N	A	T	C	A	A	...
N	T	C	T	C	G	...
N	A	G	G	C	T	...

Razdiobe jedinstvenih oznaka nukleotida prikazane su u tablici 4. Vidimo da vrlo mali udio atributa ima oznaku D, N, S ili R (ukupno 15 redaka). Zbog toga možemo izbaciti ove retke bez značajnog smanjenja skupa podataka za dubinsku analizu. S druge

Tablica 4: Udjeli nukleotida u skupu podataka za dubinsku analizu. *A, G, C i T čine značajnu većinu u skupu podataka. D, N, S i R čine neznatan dio skupa.*

Oznaka atributa	Broj atributa	Udio atributa (%)
A	44475	23.244
T	46298	24.196
G	50226	26.249
C	50281	26.278
D	2	0.0010
N	56	0.0293
S	1	0.0005
R	1	0.0005

strane, na ovaj način sprječavamo da algoritam daje previše važnosti podacima koji spadaju u ovu marginalnu skupinu (engl. *outliers*). Jedna od instanci iz klase EI[6] ima nukleotidni niz *ACACAGGGCACCCCCTCANNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN* . Čak 42 baze označene su s oznakom N. Ako bi svaki N zamijenili s jednom od četiri moguće baze dobili bismo 4^{42} kombinacija. Bez provjere u postojećim bazama podataka ne možemo znati koje su od tih kombinacija uistinu dijelovi donorskih sekcija u genomu. Korištenjem ovih kombinacija značajno bismo utjecali na distribuciju skupa podataka jer većina lokacija u genomima nije ni donorska ni akceptorska sekcija gena. Primjerice, u DNK molekuli čovjeka oko 27% genoma čine sekvence koje se prepisuju u RNK, a od toga je samo oko 1.5% kodirajućih sekvenci [5].

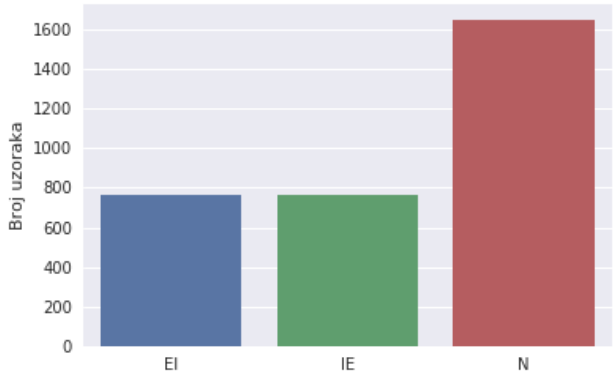
Ovako obrađen skup podataka spremljen je u datoteku *splice_filtered.csv* za daljnju analizu.

3.2 Skupovi podataka za treniranje i testiranje

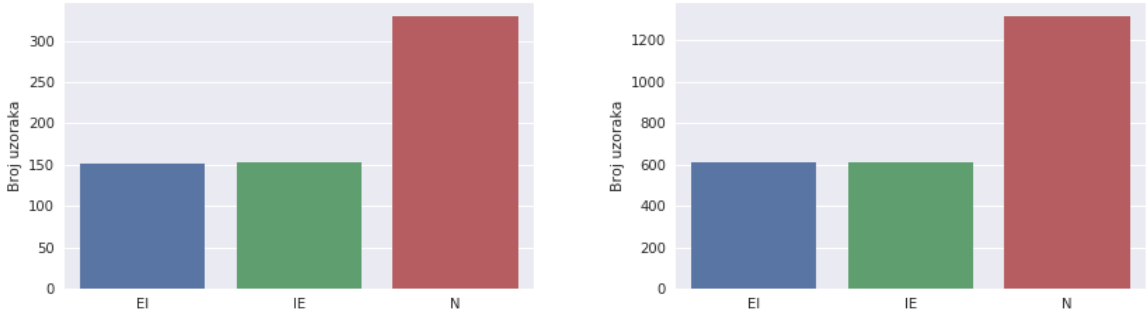
Prije nego krenemo s procesom obrade skup podataka dijelimo na dva dijela: skup za trening i skup za testiranje. Za testiranje izdvajamo 20% od ukupnog skupa podataka i njih koristimo samo za završnu ocjenu odabranih modela. Želimo osigurati da su podaci stratificirani, odnosno da su razdiobe podataka jednake u ishodišnom skupu podataka kao i generiranim skupovima.

Metoda *test_train_split* iz paketa *sklearn* ima argument *stratify* kojem predajemo stupac s klasama. Slike 4 i 5 potvrđuju da su podaci pravilno podijeljeni.

[6]HUMALPI1-DONOR



Slika 4: Distribucija instanci u skupu podataka po klasama. *Prevladava klasa N, odnosno nizovi koji ne pripadaju ni akceptorskim ni donorskim nizovima nukleotida. U pojedinačnom genomu ova razlika je još izraženija [3] i kada bismo htjeli koristiti ovaj algoritam na drugim skupovima podataka morali bismo prilagoditi broj instanci odgovarajućim omjerima.*



(a) Trening (b) Test

Slika 5: Distribucija instanci u skupovima podataka za trening i test po klasama. *Vidimo da je distribucija skupa za trening (a) i skupa za testiranje (b) jednaka distribuciji izvornog skupa podataka. Razlika je samo ukupnom broju instanci u pojedinome skupu.*

3.3 Odabir atributa

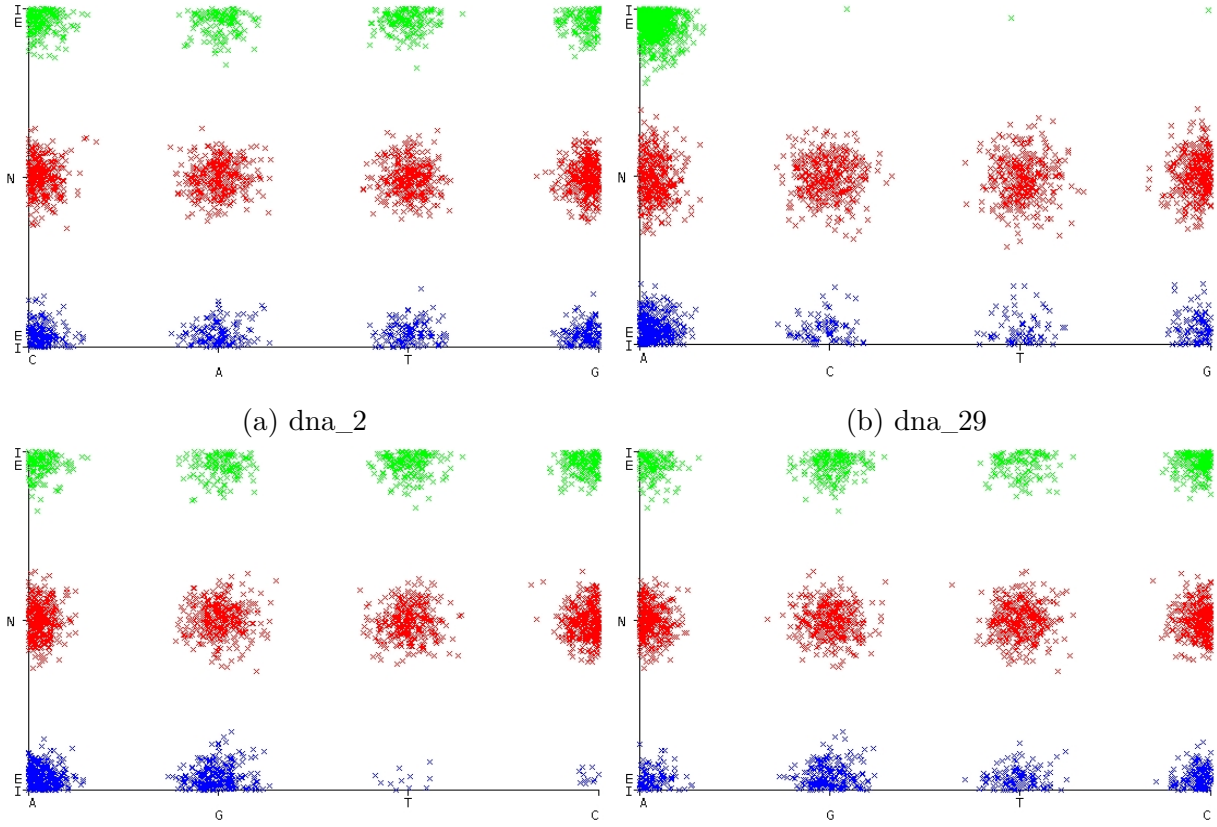
Pri kreiranju modela za dubinsku analizu podataka obično nije nužno koristiti sve dostupne atribute skupa podataka. U odabiru najboljih atributa za model primjenjuju se različite tehnike koje se u grubo mogu svrstati u tri kategorije. Metode **filtriranja** najčešće koriste različite statističke testove kako bi se odredila korelacija[7] između atributa i izlazne varijable (klase). Druga kategorija, metode **omotača**[8] koriste podskup atributa nad kojim treniraju model. Na osnovu zaključaka iz trenutnog modela, dodaju se ili oduzimaju određeni atributi - problem odabira atributa svodi se na problem pretraživanja.

[7] termin korelacija ovdje koristimo u širem smislu, ne isključivo u statističkom kontekstu

[8] engl. *wrapper methods*

Ugradbene metode[9] kombiniraju kvalitete filterskih i metoda omotavanja korištenjem algoritama koji imaju vlastite ugrađene metode selekcije atributa.

Budući da su svi podaci u skupu nominalni, te da imamo veliki broj atributa, moguće ih je prikazati samo indirektnom mjerom ili parcijalno (slika 6).



(a) dna_2 (b) dna_29

(c) dna_33 (d) dna_54

Slika 6: Frekvencija vrijednosti atributa po klasama. *Distribucija vrijednosti atributa uniformna je na indeksima udaljenima od sredine nukleotidnog niza (6a i 6d). Za indekse neposredno prije granice (6b) i poslije granice (6c) donorsko-akceptorskog područja vidimo da postoje izražena koncentracija određenih vrijednosti baza.*

Ovakva razdioba upućuje na zaključak da nisu svi atributi jednako značajni. Algoritam stabla odlučivanja (vidi poglavlje 4) ima prirodno ugrađen mehanizam za odabir atributa. Najprije se u postupku konstrukcije rangiraju atributi, a zatim se u postupku podrezivanja smanjuje ukupan broj atributa. Neki računalni znanstvenici[9] zbog ovih karakteristika koriste stablo odlučivanja kao algoritam za odabir atributa. Atributi izabrani na ovaj način zatim se koriste u drugome modelu koji može biti bilo klasifikacijski bilo regresijski postupak. U tom smislu, možemo promatrati stabla odlučivanja kao ugradbenu metodu

[9]engl. *embedded methods*

odabira atributa. Budući da je procedura izgradnje stabla odlučivanja poprilično brza onda je moguće i koristiti sve atribute, a algoritam bi se u teoriji sam trebao pobrinuti da odabere najbolje atribute. Kako bismo provjerili ovu hipotezu, kreirat ćemo i dodatne varijante stabla odlučivanja, nad podskupovima atributa. Kao metodu omotača koristimo odabir atributa pomoću višeslojnog perceptrona. Budući da smo već utvrdili da postoji određena frekvencijska relacija vrijednosti atributa i klasa provodimo postupak selekcije atributa korištenjem prikladnog statističkog testa. Kao logičan postupak nameće se tzv. Hi-kvadrat test, koji možemo svrstati u skupinu filterskih metoda odabira atributa.

3.3.1 Hi-Kvadrat test

Hi-Kvadrat je test nezavisnosti koji se koristi kako bi se odredilo postoji li značajna veza između dvije nominalne (kategoričke) varijable. Frekvencija svake vrijednosti jedne nominalne varijable (atributa) uspoređuje se sa kategorijama druge nominalne varijable (klase). Računaju se očekivane vrijednosti frekvencija vrijednosti atributa i uspoređuju sa stvarnim vrijednostima iz skupa podataka. Podaci se prikazuju u kontigencijskim tablicama (tablica 5).

Tablica 5: Primjer kontigencijske tablice. *Stupci su kategorije jedne varijable (atributa), a redovi kategorije druge varijable (klase). Krajnji redak i stupac predstavljaju sumu vrijednosti iznad, odnosno lijevo. Na osnovu sumarnih vrijednosti računaju se očekivane vrijednosti kategorija.*

	Atr1	...	AtrN	SumA
Klasa1	O_{11}	...	O_{1N}	$\sum_{j=1}^{N} O_{1j}$
..	...	O_{mn}	...	...
KlasaM	O_{M1}	...	O_{MN} ...	...
SumK	$\sum_{i=1}^{M} O_{i1}$	...	...	$\sum_{i=1}^{M} \sum_{j=1}^{N} O_{ij}$

Najprije se računaju očekivane vrijednosti na osnovu tablice kontigencije korištenjem sljedeće formule

$$E_{ij} = \frac{\sum_{k=1}^{N} O_{ik} \sum_{k=1}^{M} O_{kj}}{N} \tag{3.1}$$

gdje je

- $\sum_{k=1}^{N} O_{ik}$ - suma i-tog retka u kontigencijskoj tablici
- $\sum_{k=1}^{M} O_{kj}$ - suma j-tog stupca u kontigencijskoj tablici

Tablica 6: Sumarni podaci vrijednosti atributa za hi-kvadrat test. *Tablični pregled vrijednosti koje se koriste u hi-kvadrat testu za odabir atributa. Za atribut na indeksu 2 vidimo da se broj nukleotidnih baza poklapa sa distribucijom klasa u skupu. Za atribut na indeksu 29 vidimo da vrijednosti značajno odstupaju od razdiobe klasa.*

class	2A	2C	2G	2T	Suma	29A	29C	29G	29T	Suma
EI	176	229	195	161	761	443	106	113	99	761
IE	162	235	172	196	765	762	1	1	1	765
N	437	389	423	399	1648	400	425	409	414	1648
Suma	775	853	790	756	3174	1605	532	523	514	3174

- N - ukupan broj instanci u skupu podataka.

Nakon izračuna očekivanih vrijednosti, možemo se pristupiti izračunu vrijednosti Hi-kvadrat testa neovisnosti

$$\chi^2 = \sum_{i=1}^{M} \sum_{j=1}^{N} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \tag{3.2}$$

Hi-kvadrat testira određenu hipotezu čiju istinitost pretpostavljamo. Nulta hipoteza pretpostavlja da ne postoji statistički značajna veze između dvije varijable. Alternative hipoteza, pojednostavljeno, pretpostavlja suprotno od nulte hipoteze. Upotreba Hi-kvadrat testa detaljno je obrađena u [10].

Tablica 6 prikazuje primjer kontigencijske tablice za dva atributa, *dna_2* i *dna_29*, iz podataka nad kojima vršimo analizu. Odgovarajuća nulta i alternativna hipoteza mogu biti:

- H_0: Nukleotid na n-tom indeksu nije povezan s klasom nukleotidnog niza, i
- H_1: Nukleotid na n-tom indeksu povezan je s klasom nukleotidnog niza.

Koristeći (3.1) možemo izračunati da očekivana vrijednost kategorije T za atribut *dna_2* u slučaju klase IE iznosi $E_{IE2T} = 756 * 765/3174 = 182$ i razlika u odnosnu na stvarnu vrijednost je samo 14. U slučaju za atribut *dna_29* očekivana vrijednost na istoj poziciji u tablici iznosi $E_{IE29T} = 514 * 765/3174 = 124$, odnosno razlika je čak 123.

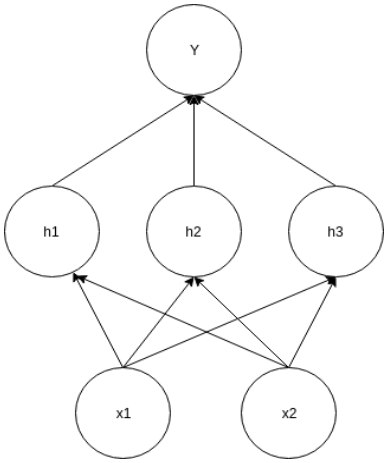
Weka chiSquaredAttributeEval

Kako ne bismo morali ručno raditi izračun Hi-kvadrat vrijednosti cijelog seta koristimo modul *chiSquaredAttributeEval* za selekciju atributa unutar Weka programskog alata. Ovaj paket nije u osnovnom instalacijskom paketu alata i treba ga naknadno instalirati korištenjem izbornika *Tools->Package Manager* u početnom pregledniku programa

Weka. Postupak odabira je vrlo brz. Algoritam rangira atribute po kvaliteti te proizvoljno odlučujemo koliko ćemo parametara od toga odabrati za naš model. Kako bi smo povećali nepristranost postupka koristimo deseterostruku unakrsnu krosvalidaciju. Detalji korištenog modela s parametrima nalaze se u Dodatku A.

3.3.2 Višeslojni perceptron

Višeslojni perceptron je tip neuronske mreže bez povratnih veza. Neuronska mreža (slika 7) modelira razdiobu koja generira ulazni skup podataka $p_{model}(X)$.



Slika 7: Model jednostavne neuronske mreže. *x_i su atributi instance koji čine ulazni sloj mreže, h_i su neuroni skrivenog sloja, a y je neuron u izlaznom sloju. Višeslojni perceptron mora imati minimalno ova tri sloja, a broj skrivenih slojeva u teoriji nije ograničen.*

Osnovna procesna jedinica u neuronskoj mreži je neuron, nazvan tako jer je modeliran prema stanicama živčanog sustava. Neuroni su u mreži organizirani u slojeve. Ulazni sloj prihvaća instancu kao ulazni podatak u obliku vektora atributa X_N. Neuron izračunava ponderiranu sumu atributa

$$z = f(\mathbf{x}; \mathbf{w}, b) = \mathbf{x}^T\mathbf{w} + b \tag{3.3}$$

gdje su w parametri neurona, a b pomak (engl. **bias**). Nad rezultatom ove funkcije izračunava nelinearna transformacija, sigmoidna funkcija oblika

$$y = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{3.4}$$

ili $tanh(z)$. U posljednje vrijeme, najčešće se koristi ReLU[10] aktivacijska funkcija.

Rezultat nelinearne transformacije iz svih neurona jednog sloja prosljeđuje se neuronima u sljedećem sloju. Završni sloj na izlazima daje vektor vjerojatnosti Y_K da određena instanca pripada klasi k korištenjem tzv. *softmax* funkcije

$$P(y = k|\mathbf{x}) = \frac{e^{y_k}}{\sum_{i=1}^{K} e^{y_i}} \tag{3.5}$$

[10]engl. *Rectified Linear Unit*

Broj slojeva te broj neurona u sloju pripadaju tzv. hiperparametrima algoritma. Parametri mreže, tj. težine W koje se koriste za izračun ponderirane sume uče se pomoću metode povratne propagacije greške. Stvarni izlazi mreže uspoređuju se sa željenim izlazima u svakom koraku učenja te se koristi odgovarajuća mjera pogreške. Metoda povratne propagacije određuje koliko trebamo prilagoditi parametre u pojedinim neuronima kako bi smo ovu grešku spustili na željenu razinu [11].

Weka MultilayerPerceptron

Weka omogućava korištenje višeslojnog perceptrona u postupku odabira atributa. Koristi se modul *ClassifierSubsetEval* unutar kojeg se podešava algoritam za selekciju *classifiers.functions.MultiLayerPerceptron*. Postupak treniranja mreže za odabir traje oko tri sata na računalu s IntelCore i5 procesorom i 8GB RAM memorije. Treniranje modela izvršeno je nad cjelokupnim trening setom. Ovaj model izabrao je ukupno trinaest atributa. Detalji modela s korištenim parametrima nalaze se u dodatku A.

4 Stabla odlučivanja

Stabla odlučivanja vrlo su popularna metoda dubinske analize podataka zbog svoje jednostavnosti i brzine. Algoritam koristi "podijeli-pa-vladaj" paradigmu i najjednostavnije se može opisati u rekurzivnom obliku. Najprije se odabire atribut koji će biti korijen stabla. Za svaku moguću vrijednost koju taj atribut može imati izvodi se jedna grana. Na ovaj način se skup podataka dijeli u podskupove. Ako jedan od čvorova stabla dobivenih na ovaj način sadrži samo instance jedne klase, zaustavlja se rekurzivni postupak. Inače, nastavljamo dijeliti preostale podskupove prema novim atributima. Odabir atributa nije proizvoljan, želimo odabrati atribute na takav način da stablo bude što manje tako da naš algoritam radi brže.

Indukcijski zadatak provodi se nad skupom objekata koji su opisani kolekcijom atributa. Svaki atribut mjeri neko važno svojstvo objekta te u pravilu poprima veličinu iz diskretnog skupa vrijednosti. Objekti pripadaju jednoj od dvije ili više međusobno isključivih klasa. Iz skupa podataka za trening u kojem su klase objekata poznate izvode se klasifikacijska pravila. Ukoliko su atributi adekvatni (objekti s istim vrijednostima atributa pripadaju istoj klasi) uvijek je moguće konstruirati stablo odlučivanja koje će ispravno klasificirati sve objekte u trening skupu [12]. Međutim, stablo odlučivanja koje ispravno klasificira samo trening skup nije korisno jer zapravo samo izražava podatke koje imamo u tablici (trening skupa) u obliku stabla. Kako bi se konstruirano stablo moglo primjeniti i na buduće, dotad neviđene podatke (testni skup) stablo odlučivanja mora sadržavati smislene informacije o odnosima atributa objekta i klase tog objekta.

Računalni znanstvenik John Ross Quinlan dao je značajan doprinos razvoju algoritama za dubinsku analizu temeljenim na stablima odlučivanja [13], [14], [15]. Istraživanje je potaknuto potrebom da se unaprijede algoritmi sa sposobnošću pronalaska znanja u samim skupovima podataka bez korištenja domenskih eksperata. Prvi iz niza varijanti stabla odlučivanja koje je dizajnirao, ID3, je dizajniran za skupove podataka s velikim brojem objekata i velikim brojem atributa objekta poštujući ograničenje da konstruirano stablo odlučivanja bude jednostavno (kako bi se ubrzao proces generiranja takvog stabla uz minimalne računalne resurse). Posljedica ovakvih ograničenja u dizajnu je da ID3 ne garantira globalno optimalno stablo odlučivanja [14].

4.1 C4.5 algoritam

C4.5 je nasljednik algoritma ID3[11]. Kao i prethodni, C4.5 generira klasifikator u obliku stabla odlučivanja, ali može kreirati i klasifikator u razumljivijem formatu, u obliku skupa pravila. Pretpostavimo da postoji skup S slučajeva. C4.5 konstruira stablo na sljedeći

[11] C4.5 također ima nasljednika, komercijalni algoritam naziva C5.0

način:

- Ako svi slučajevi u S pripadaju istoj klasi ili veličina skupa nije značajna, stablo je samo čvor s oznakom najčešće klase u skupu S
- Inače, odabrati test na jednom atributu s dva ili više ishoda i postaviti ovaj test u korijen stabla s granom za svaki ishod. Podijeliti skup S na podskupove *S1*, *S2*, ... ovisno o ishodu svakog test te nastaviti s procedurom rekurzivno za svaki podskup.

Ova definicija je poprilično općenita jer bi mogli odabrati mnogo različitih testova. C4.5 koristi dvije heuristike za rangiranje testova

- *Informacijska vrijednost IG*[12] - minimizira ukupnu entropiju podskupova i
- *Omjer dobiti GR*[13] - dijeli IG sa informacijom iz ishoda testa.

Dozvoljeni su i numerički i nominalni atributi. Numberički atributi u pravilu koriste testove s pragovima, te ishodi ovise o tome jesu li vrijednosti atributa veće ili manje (ili jednake) od praga. Kod atributa s diskretnim vrijednostima u pravilu testovi imaju jednak broj ishoda kao i broj mogućih vrijednosti tog atributa, ali je moguće i grupiranje vrijednosti kako bi se smanjio broj ishoda testa, a samim time i složenost konstruiranog stabla.

Inicijalno konstruirano stablo je podložno pretreniranju (engl. *overfitting*) zbog čega se koristi algoritam podrezivanja stabla. Podrezivanje se odvija od listova prema korijenu. Računa se pesimistična procjena učestalosti pogrešak korištenjem binomne razdiobe za slučaj kada je registrirano E događaja, koji ne pripadaju najčešćoj klasi, u N pokušaja, s korisnički definiranim intervalom pouzdanosti. Za podstablo se sumira procjena pogreške svih grana i uspoređuje sa greškom u slučaju da se cijelo podstablo zamjeni čvorom. Ukoliko potonja greška nije veća stablo se podrezuje. Također, moguće je zamjenjivanje podstabla samo jednom njegovom granom ukoliko to ne povećava pogreške[15].

4.1.1 Informacijska vrijednost i omjer dobiti

Algoritam C4.5 koristi osnovne postulate informacijske teorije u procesu izgradnje stabla. Prema teoriji informacije, informacija sadržana u poruci ovisi o vjerojatnosti te poruke p i može se mjeriti u bitovima kao negativan logaritam te vjerojatnosti [4]. Pretpostavimo da postoji skup instanci S veličine $|S|$ unutar kojeg svaka od instanca pripada točno jednoj od J klasa gdje ukupan broj instanci klase j u skupu S označavamo sa $|C_{jS}|$. Ako

[12] engl. *Information Gain*
[13] engl. *Gain Ratio*

nasumično izaberemo jednu instancu iz skupa vjerojatnost da ona pripada klasi $j \in 1, .., J$ iznosi

$$p = \frac{|C_j S|}{|S|}. \tag{4.1}$$

Očekivana informacija takve poruke u odnosu na pripadnost klasi izražena je sumom svih klasa u proporciji sa njihovim udjelima u skupu S.

$$info(S) = -\sum_j \frac{|C_{jS}|}{|S|} \cdot \log_2 \frac{|C_{jS}|}{|S|} \tag{4.2}$$

U trening skupu $info(S)$ mjeri prosječnu informaciju potrebnu da bi se identificirala klasa instance iz skupa S. Nakon što smo podijelili skup S u n particija u ovisnosti od ishoda testa X očekivanu informaciju računamo kao težinsku sumu podskupova

$$info_X(S) = -\sum_i \frac{|S_i|}{|S|} \cdot info(S_i) \tag{4.3}$$

Konačno, informacijsku dobit računamo kao

$$gain(X) = info(S) - info_X(S) \tag{4.4}$$

U algoritmu ID3 favorizirani su testovi s puno ishoda. Na primjer, u našem testnom skupu atribut *id* je jedinstven za svaku instancu. Korištenjem takvog atributa dobili bismo veliki broj podskupova od kojih svaki sadrži samo jedan slučaj. Budući da bi ovi podskupovi imali samo jednu klasu, $info_X S$ bi uvijek bio 0, odnosno iformacijska vrijednost od korištenja ovakvog atributa bi bila maksimalna. Međutim, ovakvo stablo ne bi moglo klasificirati ni jednu novu instancu (jer bi ona imala *id* koji se nije nalazio u skupu za trening). To je zapravo razlog zašto smo izbacili ovaj atribut u procesu predobrade. C4.5 je popravio ovaj nedostatak korištenjem normalizacije. Izraz

$$splitinfo(X) = -\sum_i^n \frac{|S_i|}{|S|} \cdot \log_2 \frac{|S_i|}{|S|} \tag{4.5}$$

predstavlja potencijalnu informaciju dobivenu dijeljenjem skupa S u n podskupova. Ova veličina povezana je s informacijskim sadržajem ishoda testa za razliku od informacije klasifikacije u slučaju informacijske dobiti.

$$GR(X) = info(S)/splitinfo(X) \tag{4.6}$$

Omjer dobiti 4.6 izračava udio informacije generiran podjelom nastalom korištenjem testa X [14].

4.1.2 Weka implementacija

C4.5 algoritam implementiran je pod nazivom J48 u programskom alatu *Weka*. Korištene su standardne postavke algoritma. Na veličinu stabla najviše utječu dva parametra, minimalni broj objekata u podgrupi *minNumObj* čijim povećavanjem smanjujemo stablo i faktor pouzanosti *confidenceFactor* čijim povećavanjem povećavamo generirano stablo.

4.1.3 Nedostatci C4.5 algoritma

Postoje dva glavna razloga koja utječu na lošije performanse algoritma: ograničenje jednostruke pokrivenosti i fragmentacija. Ograničenje jednostruke pokrivenosti uzrokovano je heurističkom prirodom metode. Konstrukcija C4.5 stabala odvija se tako da se odabire atribut koji je najdiskriminatorniji u cijelom skupu podataka te se na osnovu njega podaci dijele u nepreklapajuće podskupove koji bi trebali imati što više uzoraka iste klase. Svaka podgrupa odgovara jednom pravilu i svaka instanca zadovoljava ograničenja samo jednog pravila (instanca se ne može nalaziti u više podgrupa odjednom). S druge strane, to znači da svaka instanca mora zadovoljiti samo jedno pravilo što rezultira generiranjem manjeg broja značajnih pravila čak i u višedimenzionalnim skupovima podataka. Mali broj značajnih pravila može uzrokovati pristranost u predikcijama. Fragmentacija nastaje zbog generiranje mnogih lokalno važnih, ali globalno nevažnih pravila u podskupovima dublje u stablu [16].

5 Rezultati analize

U radu su testirane performanse četiri modela (tablica 7).

Tablica 7: Popis testiranih modela.

Model	Opis
Chi2Top10	Stablo odlučivanja sa najboljih 10 atributa odabranih Hi-kvadrat testom
Chi2Top20	Stablo odlučivanja sa najboljih 20 atributa odabranih Hi-kvadrat testom
MP	Stablo odlučivanja s 13 atributa koji su odabrani korištenjem višeslojnog perceptrona
C4.5All	Standardno stablo odlučivanja koje koristi sve dostupne atribute u skupu podataka

Razumijevanje rezultata neophodno uključuje i razumijevanje problema koji se modelom opisuje te karakteristike skupa podataka nad kojim je taj model razvijen. U našem slučaju, klasifikator koji bi uvijek davao klasu N (nije detektirana granica IE ili EI) bi i dalje imao veliku točnost, budući da je većina genoma sastavljena od nekodirajućih dijelova. Za procjenu uspješnosti klasifikacije koriste se različite mjere. Osnovne mjere kvalitete modela su:

- TP (engl. *true positives*) - broj pozitivno predviđenih instanci koje su stvarno pozitivne
- TN (engl. *true negatives*) - broj negativno predviđenih instanci koje su stvarno negativne
- FP (engl. *false positives*) - broj pozitivno predviđenih instanci koje su stvarno negativne
- FN (engl. *false negatives*) - broj negativno predviđenih instanci koje su stvarno pozitivne

Iz osnovnih mjera izvodim složenije mjere kvalitete modela

- Točnost - postotak točno klasificiranih instanci

$$Točnost = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

- TPR - udio pozitivno predviđenih instanci koje su stvarno pozitivne
- FPR - udio negativno predviđenih instanci koje su stvarno negativne

- F-mjera - jer harmonična srednja vrijednost preciznosti i odziva gdje je preciznost

$$preciznost = \frac{TP}{TP + FN} \tag{5.2}$$

i odziv

$$odziv = \frac{TP}{TP + FP} \tag{5.3}$$

- AUC - površina pod ROC krivulja prikazuje odnos između udjela lažnih predviđanja FPR (X-os) i točnih predviđanja TPR (Y-os). Iz ove krivulje možemo očitati vjerojatnost da model bolje rangira pozitivne od slučjano odabranih negativnih instanci.

- ROC

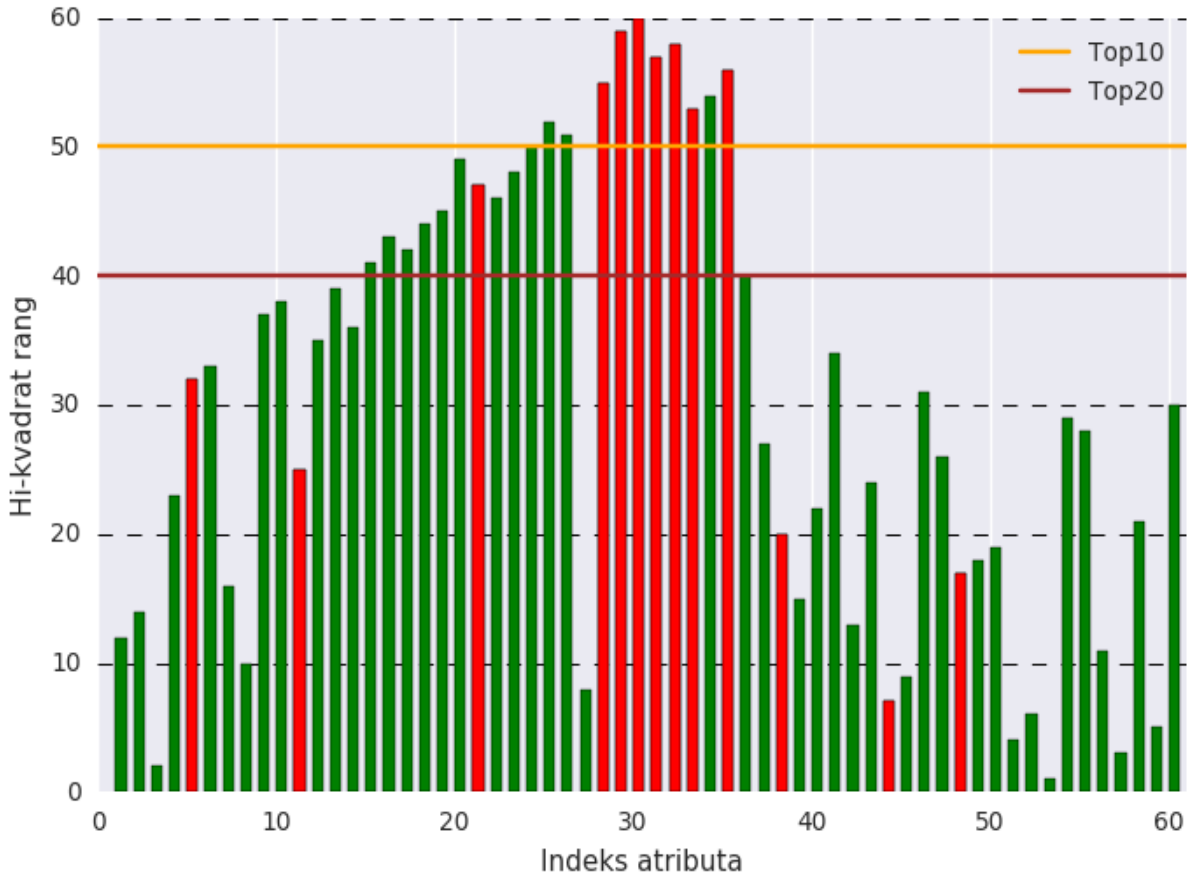
Ovakve mjere su binarne, odnosno imaju smisla za dvije klase. Kada imamo više klasa poopćujemo mjere tako da iteriramo kroz sve klase. U svakoj iteraciji samo jedna klasa je pozitivna, a sve ostale klase grupiramo zajedno kao negativnu klasu (engl. *one-vs-all*). Konačan je rezultat aritmetička sredina rezultata svih pojedinačnih iteracija.

Tablica 8: Prikaz rezultata uspješnosti modela. *U posljednja dva stupca (desno) su broj čvorova koji su listovi te ukupan broj čvorova u stablu.*

Algoritam	Točnost	TPR	FPR	F-mjera	PRC	ROC	N Listova	N
C4.5All	93.38	0.934	0.032	0.934	0.913	0.96	103	137
Chi2Top10	93.85	0.939	0.027	0.939	0.905	0.959	88	117
Chi2Top20	93.85	0.939	0.029	0.939	0.916	0.963	115	153
MP	94.64	0.946	0.022	0.947	0.931	0.97	100	133

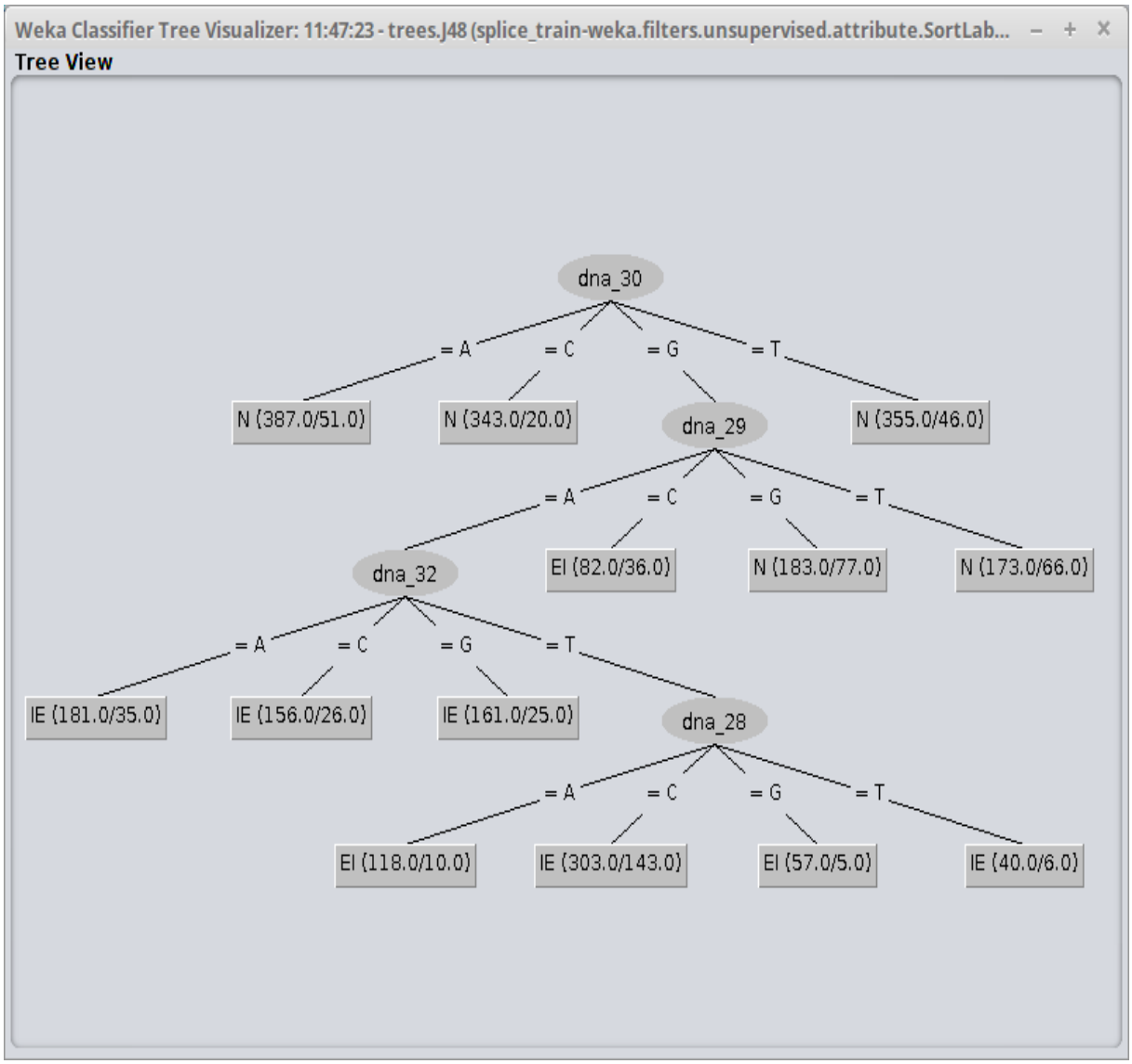
Tablica 8 prikazuje uspješnost algoritama korištenjem gore opisanih mjera. Vidimo da model razvijen s atributima izabranim pomoću višeslojnog perceptrona ima najbolje performanse. Razlika u točnosti od 1% ne čini se velika na prvi pogled, ali u kontekstu količine podataka koje se obrađuju u bioinformatici postaje značajna. Tipičan genom ima nekoliko stotina milijuna do nekoliko milijardi baza. Za sekvencijalni algoritam koji bi testirao svaku poziciju u genomu korištenjem stabla odlučivanja, razlika u točnosti od 1% značila bi u tom kontekstu nekoliko stotina do nekoliko milijuna više točnih klasifikacija. Uspoređujući veličinu stabala vidimo da je, očekivano, najmanje stablo generirano na skupu koji ima najmanje atributa *Chi2Top10*. Međutim, smanjenje je neznazno u odnosu na *C4.5All* koji ima veličinu stabla kao i *MP* model, a za njega nije bila potrebna nikakva predobrada, odnosno selekcija atributa. Rezultati modela *Chi2Top20* pokazuju da selekcija atributa ne mora nužno rezultirati poboljšanim performansama. Ovaj model ima najveće generirano stablo s jednakim postavkama treniranja. Slika 8 može objasniti

nastale razlike. Model *MP* i *Chi2Top10* imaju veliko poklapanje (7 od 10) odabranih atributa. Međutim, *MP* koristi samo jedan atribut po rangu Hi-kvadrat testa između desetog i dvadesetog mjesta. Dok je Hi-kvadrat test najviše težine dao atributima iz sredine niza višeslojni perceptron je prepoznao prediktivnu snagu i kod atributa bliže rubovima nukleotidnog niza, što je u ovom slučaju rezultiralo poboljšanjem točnosti od gotovo 1%.



Slika 8: Grafički prikaz odabira atributa rangiranih Hi-kvadrat testom. *Iznad narančaste linije nalazi se najboljih 10 atributa, a iznad smeđe najboljih 20 atributa po Hi-kvadrat testu. Crvenom su označeni atributi izabrani korištenjem višeslojnog perceptrona.*

Jedna od najvećih kvaliteta algoritma C4.5 je mogućnost iščitavanja značajki skupa podataka iz dobivenog modela (stabla). Slika 9 prikazuje jedan mogući model. Odabran je velik stupanj podrezivanja zbog preglednosti. Zbog toga ovaj model obuhvaća samo četiri atributa. Stablo interpretiramo na slijedeći način. U korijenu stabla je *dna_30* atribut. Gledamo koji je nukleotid na toj poziciji u instanci. Ako se na toj poziciji ne nalazi nukleotid G, onda instanca ne predstavlja nukleotidni niz koji pripada akceptorskoj (klasa IE) odnosno donorskoj grupi (klasa EI). Ako se na toj poziciji nalazi nukleotid G, onda gledamo sadržaj atributa *dna_29*. Nukleotidi G i T na ovoj lokaciji znače da se ne radi niti o akceptorskoj niti o donorskoj sekvenci. Ako je na toj lokaciji nukleotid C,



Slika 9: Stablo odlučivanja - Weka model *Stablo je stvoreno korištenjem skupa podataka sa svim atributima (model C4.5All) s minimalnim brojem objekata u podgrupi minNumObj podešenim na 100. Ovakav model ima značajno nižu točnost (oko 80%) u odnosu na modele u tablici 8*

instancu svrstavamo u donorsku skupinu. Ako je vrijednost atributa A provjeravamo dalje vrijednost atributa *dna_32*. Ako je vrijednost ovog atributa T nastavljamo s provjerom atributa *dna_28* inače je instanca klase IE. Konačno atribut na lokaciji 28 je posljedni i on svrstava instancu u klase EI (ako je vrijednost atributa A ili G) i IE (ako je vrijednost atributa C ili T).

Zaključak

Razvijeni model pokazuje kako stabla odlučivanja omogućuju kreiranje vrlo uspješnih modela s vrlo malo ili nimalo domenskoga znanja. Sami algoritam u sebi ima heuristiku koja mu omogućuje rangiranje atributa po važnosti te odbacivanja manje važnih atributa. Također, pokazano je da selekcija atributa može imati različite efekte na konačne rezultate, od zadržavanja točnosti uz smanjenje veličine stabla, povećanja točnosti uz zadržanu veličinu stabla do degradacije performansi modela (ista točnost, ali povećano stablo odlučivanja). To je samo još jedna potvrda činjenice da put do najboljeg modela u dubinskoj analizi nije pravocrtan nego podrazumijeva iterativni proces uz postupno podešavanje parametara modela.

Stabla odlučivanja su se za ovaj model pokazala kao vrlo brz i uspješan algoritam i potvrdila pretpostavku da heuristike ugrađene u dizajn modela mogu izabrati kvalitetan podskup podataka. Iako je jedan od najstarijih algoritama u domeni, stabla odlučivanja u različitim varijantama ostaju privlačna zbog svoje jednostavnosti i mogućnosti da generiraju rezultate koji pomažu istražiteljima interpretirati svojstva skupova podataka koji se analiziraju što je zapravo i krajnji cilj svake dubinske analize.

Dodatak A - Weka modeli za odabir parametara

Izbor atributa korištenjem Hi-kvadrat testa

data.txt

```
=== Run information ===

Evaluator:     weka.attributeSelection.ChiSquaredAttributeEval
Search:        weka.attributeSelection.Ranker -T -1.7976931348623157E308 -N -1
Relation:      splice_train-weka.filters.unsupervised.attribute.
               SortLabels-Rfirst-last-SNON-CASE
Instances:     2539
Attributes:    61

Evaluation mode:    10-fold cross-validation



=== Attribute selection 10 fold cross-validation stratified, seed: 1 ===

average merit      average rank  attribute
998.05 +-12.276       1   +- 0        31 dna_30
872.552 +-11.093       2.4 +- 0.49     30 dna_29
872.012 +-10.328       2.8 +- 0.75     33 dna_32
843.052 +-10.07        3.8 +- 0.4      32 dna_31
708.743 +-10.873       5   +- 0        36 dna_35
565.947 +- 7.334       6   +- 0        29 dna_28
453.642 +-12.76        7   +- 0        35 dna_34
387.458 +- 7.643       8   +- 0        34 dna_33
298.811 +- 9.201       9   +- 0        26 dna_25
235.924 +- 9.753      10.5 +- 0.67     27 dna_26
228.213 +- 5.309      10.8 +- 0.6      25 dna_24
214.605 +- 5.134      12.2 +- 0.6      21 dna_20
210.861 +- 8.942      12.6 +- 1.02     24 dna_23
196.816 +- 6.578      14.3 +- 0.64     22 dna_21
190.725 +- 5.983      14.7 +- 0.64     23 dna_22
170.088 +- 9.313      16.4 +- 0.66     20 dna_19
169.006 +- 5.131      16.5 +- 0.5      19 dna_18
147.689 +- 5.179      18.2 +- 0.4      17 dna_16
136.26 +- 6.087       18.8 +- 0.4      18 dna_17
105.513 +- 2.335      20.6 +- 0.49     16 dna_15
108.705 +- 5.366      20.6 +- 0.8      37 dna_36
94.753 +- 5.816       21.8 +- 0.4      14 dna_13
78.477 +- 4.819       23.9 +- 0.7      11 dna_10
```

```
78.397 +- 4.408      24.1 +- 0.7      10 dna_9
78.232 +- 7.482      24.2 +- 1.25     15 dna_14
67.761 +- 5.975      26   +- 0.77     13 dna_12
56.327 +- 3.975      27.8 +- 1.17     42 dna_41
54.375 +- 4.025      28.6 +- 1.5       7 dna_6
53.969 +- 4.849      28.8 +- 1.4       6 dna_5
46.905 +- 5.809      30.6 +- 2.24     47 dna_46
46.395 +- 6.133      31.2 +- 1.94     61 dna_60
45.29  +- 3.355      31.6 +- 0.92     55 dna_54
42.811 +- 4.289      32.9 +- 2.74     56 dna_55
39.215 +- 2.867      34.9 +- 2.12     38 dna_37
37.807 +- 4.319      36.1 +- 2.66     48 dna_47
36.914 +- 3.184      36.4 +- 2.15     12 dna_11
37.416 +- 3.352      36.5 +- 2.2      44 dna_43
33.902 +- 3.895      38.9 +- 3.99      5 dna_4
32.718 +- 2.864      39.4 +- 1.85     41 dna_40
31.377 +- 3.444      40.5 +- 2.42     59 dna_58
30.455 +- 3.77       41.4 +- 2.84     39 dna_38
30.714 +- 4.419      41.7 +- 3.58     51 dna_50
29.309 +- 4.321      43.4 +- 4.1      50 dna_49
27.527 +- 3.173      44.7 +- 3.55     49 dna_48
25.91  +- 4.333      45.7 +- 3.93      8 dna_7
26.841 +- 3.411      46   +- 3.71     40 dna_39
25.745 +- 1.951      46.8 +- 2.56      3 dna_2
25.306 +- 2.555      47.3 +- 3.38     43 dna_42
24.058 +- 2.763      48.7 +- 3.44      2 dna_1
23.815 +- 2.079      48.8 +- 2.36     57 dna_56
22.864 +- 3.059      50   +- 3.77      9 dna_8
22.309 +- 1.955      50.6 +- 2.33     46 dna_45
19.023 +- 2.622      54   +- 1.95     28 dna_27
18.605 +- 3.187      54.3 +- 2.79     45 dna_44
18.086 +- 3.286      54.6 +- 3.04     53 dna_52
17.158 +- 2.847      55.6 +- 2.42     60 dna_59
16.736 +- 2.233      56.1 +- 1.87     52 dna_51
15.58  +- 1.234      56.8 +- 0.98     58 dna_57
12.983 +- 1.901      58.3 +- 1.68      4 dna_3
11.054 +- 1.812      59.8 +- 0.4      54 dna_53
```

Izbor atributa korištenjem višeslojnog perceptrona

data.txt

```
=== Run information ===
```

```
Evaluator:     weka.attributeSelection.ClassifierSubsetEval -B
               weka.classifiers.functions.MultilayerPerceptron -T -H
               "Click to set hold out or test instances" -E DEFAULT --
               -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a
Search:        weka.attributeSelection.GreedyStepwise -T -1.7976931348623157E308
               -N -1 -num-slots 1
Relation:      splice_train-weka.filters.unsupervised.attribute.
               SortLabels-Rfirst-last-SNON-CASE
Instances:     2539
Attributes:    61
Evaluation mode:    evaluate on all training data


=== Attribute Selection on all input data ===

Search Method:
        Greedy Stepwise forwards.
        Start set: no attributes
        Merit of best subset found:    0.998

Attribute Subset Evaluator supervised, Class nominal: 1 class:
        Classifier Subset Evaluator
        Learning scheme: weka.classifiers.functions.MultilayerPerceptron
        Scheme options: -L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a
        Hold out/test set: Training data
        Subset evaluation: classification error

Selected attributes: 6,12,22,29,30,31,32,33,34,36,39,45,49 : 13
                     dna_5
                     dna_11
                     dna_21
                     dna_28
                     dna_29
                     dna_30
                     dna_31
                     dna_32
                     dna_33
                     dna_35
                     dna_38
                     dna_44
                     dna_48
```

Literatura

[1] Watson, J. D., Crick, F. H. C., "Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid", Nature, Vol. 171, No. 4536, travanj 1953, str. 737-738.

[2] Nelson, D. L., Cox, M. M., Principles of Biochemistry, 5th ed. New York, USA: W.H. Freeman and Company, 2008.

[3] Brown, T. A., Genomes, 2nd ed. New York, USA: Wiley-Liss, 2002.

[4] Shannon, C. E., "A Mathematical Theory of Communication", Bell System Technical Journal, Vol. 27, No. 4, listopad 1948, str. 623-656.

[5] Berg, J. M., Tymoczko, J. L., Stryer, L., Biokemija, 6th ed. Zagreb, Hrvatska: Školska knjiga, 2013.

[6] McKinney, W., "Data Structures for Statistical Computing in Python", in Proceedings of the 9th Python in Science Conference, Austin, TX, USA, srpanj 2010, str. 51-56.

[7] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., "Scikit-learn: Machine learning in Python", Journal of Machine Learning Research, Vol. 12, 2011, str. 2825-2830.

[8] E. Frank and M. A. Hall and I. H. Witten, The Weka Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques", 4th ed., 2016.

[9] Grabczewski, K., Jankowski, N., "Feature selection with decision tree criterion", in Fifth International Conference on Hybrid Intelligent Systems (HIS'05), studeni 2005, str. 6 pp.-.

[10] Grubišić, A., "Hi-kvadrat test i njegove primjene", Zagreb, 2004, seminarski rad, Fakultet elektrotehnike i računarstva.

[11] Goodfellow, I., Bengio, Y., Courville, A., Deep Learning, 1st ed. MIT Press, 2016.

[12] Witten, I. H., Frank, E., Hall, M. A., Pal, C. J., DATA MINING Practical Machine Learning Tools and Techniques, 4th ed. Cambridge, USA: Morgan Kaufmann, 2016.

[13] Quinlan, J. R., C4.5: Programs for Machine Learning, 1st ed. San Mateo, CA, USA: Morgan Kaufmann, 1993.

[14] Quinlan, J. R., “Induction of Decision Trees”, Machine Learning, Vol. 1, No. 1, ožujak 1986, str. 81-106.

[15] Wu, X., Kumar, V., Ross, Q. J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Yu, P. S., Zhou, Z. ., Steinbach, M., Hand, D. J., Steinberg, D., “Top 10 algorithms in data mining”, Knowledge and Information Systems, Vol. 14, No. 1, siječanj 2008, str. 1-37.

[16] Li, J., Wong, L., “Using Rules to Analyse Bio-medical Data: A Comparison between C4.5 and PCL”, in Advances in Web-Age Information Management, Berlin, Heidelberg.