LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

ĢENĒTISKO ALGORITMU IZMANTOŠANA KLASIFIKĀCIJĀ

MAĢISTRA DARBS

Autors: **Edmunds Ozoliņš**
Stud. apl. Nr. eo12018
Darba vadītājs: profesors Dr. dat. Jānis Zuters

RĪGA 2018

ANOTĀCIJA

Darbā tiek pētīti tie klasifikācijas problēmas risinājumi, kuri izmanto ģenētiskos algoritmus. Klasifikācijas problēmā dati tiek grupēti pa to klasēm. Datu punktiem koordinātu telpā tiek noteiktas to klases. Ģenētiskie algoritmi ir heiristika, kura var uzlabot risinājumus, tos kombinējot. Iespējams klasifikācijas problēmas risinājums ir lēmumu koks.

Maģistra darba mērķis ir izpētīt ģenētisko algoritmu izmantošanas iespējas klasifikācijā.

Rezultātā tika izstrādāti ģenētisko algoritmu atribūtu selekcijas, klasifikācijas un ansambļa risinājumi. Klasifikācijas risinājums izmanto selekcijas risinājumu. Attiecīgi ansambļa risinājums izmanto klasifikācijas risinājuma klasifikatorus. Tika īstenotas autora jaunās idejas un tika veikta testēšana uz vispārzināmām un publiski pieejamām datu kopām.

Atslēgvārdi: klasifikācija, ģenētiskie algoritmi, Java, UCI mašīnmācīšanās repozitorijs.

ABSTRACT

APPLICATION OF GENETIC ALGORITHMS IN CLASSIFICATION

This work studies those solutions of the classification problem, which use genetic algorithms. In the classification problem data is grouped by classes. Classes are determined for data points in coordinate space. Genetic algorithms are a heuristic that can improve solutions by combining them. Decision tree is a feasible solution to the classification problem.

The goal of the master's thesis is to study the possibilities of applying genetic algorithms in classification.

As a result, genetic algorithm attribute selection, classification and ensemble solutions were created. The classification solution uses the selection solution. Correspondingly the ensemble solution uses classification solution's classifiers. Author's new ideas were applied and testing was carried out on well-known and publicly available data sets.

Keywords: classification, genetic algorithms, Java, UCI machine learning repository.

AUTOREFERĀTS

Darbā autors deva aprakstu par klasifikācijas problēmu un ģenētiskajiem algoritmiem. Tika izpētīti un analizēti materiāli par citiem klasifikācijas problēmas risinājumiem, kuri izmanto ģenētiskos algoritmus. Tika izstrādāti autora ģenētisko algoritmu atribūtu selekcijas, klasifikācijas un ansambļa klasifikācijas risinājumi, kuri tika testēti uz vispārzināmām un publiski pieejamām datu kopām.

Darbā tiek piedāvātas unikālas risinājumu struktūras, kā arī tiek pielietotas jaunas autora idejas krustmijai, mutācijai, indivīdu derīguma funkcijai un saražoto klasifikācijas noteikumu sakombinēšanai.

Literatūras izpētes ziņā tika apskatīti vairāki citi jauni klasifikācijas problēmas risinājumi, kuri izmanto ģenētiskos algoritmus. Katrs apskatītais risinājums trešajā nodaļā tika analizēts, kā arī tika veikts analīzes kopsavilkums. Maģistra darba avoti sastāv no 16 zinātniskiem rakstiem, 2 grāmatām, 1 tehniskā dokumenta, 1 bakalaura darba, 1 maģistra kursa darba un 5 nerecenzētiem interneta avotiem.

Darbā aplūkotais ir aprakstīts pietiekami detalizēti. Tiek dots problēmas konteksts, galvenās citu risinājumu iezīmes un autora veiktā analīze, eksperimenti. Autora risinājumu ziņā jaunās autora idejas ir aprakstītas teksta formā, kā arī tiek piedāvāti ilustratīvi attēli. Izstrādātās programmatūras kods ir pieejams apskatei, kā arī ir aprakstīts, kā izmēģināt izstrādāto.

Darba izstrādes laikā tika izstrādāta programmatūra Java programmēšanas valodā, kā arī tika veikti eksperimenti ar to. Autora izstrādātais pirmkods satur apmēram 3500 rindu. Eksperimenti tika veikti ar pavisam 12 datu kopām.

Atribūtu selekcijas ziņā darba rezultāti ir saskanīgi ar citiem līdzīgiem rezultātiem literatūras avotos – ar ģenētisko algoritmu selekciju var atmest lielu daļu atribūtu, kā arī var iegūt dažu % precizitātes uzlabojumu. Autora ģenētisko algoritmu klasifikācijas un ansambļa klasifikācijas risinājumu ziņā citi līdzīgi rezultāti ir labāki, bet autora risinājumi atsevišķos gadījumos var sasniegt lielāku precizitāti. Izstrādāto testēja autors, pārbaudot katru funkciju. Izstrādātie risinājumi tika vērtēti ar 10 reižu šķērsvalidāciju un iepriekš neredzētām testēšanas kopām.

Darba noformējuma ziņā maģistra darba teksts ir pārskatīts un atrastās nepilnības ir izlabotas. Darbs tika pārbaudīts ar pareizrakstības pārbaudītāju. Oficiāli pieņemtā nozares

terminoloģija ir izmantota, kur tas ir iespējams. Autors izskatīja darba noformējuma kontrolsarakstu un izlaboja konstatētās nepilnības.

Izmantotie citu autoru avoti ir pareizi atzīmēti.

SATURS

APZĪMĒJUMU SARAKSTS

AdaBoost – klasifikācijas algoritms, kurš pakāpeniski būvē labākus klasifikatorus (*adaptive boosting*).

API – lietojumprogrammas saskarne. Funkcionalitāte, kuru var lietot ārēja programmatūra (*application programming interface*).

Bagging – algoritms, kurā ansambļa klasifikatorus trenē katru ar savu datu apakškopu, kur katrs ieraksts no oriģinālajiem datiem var parādīties vairākas reizes (*bootstrap aggregating*).

C4.5 – lēmumu koku algoritms, kuru izmanto klasifikācijā.

DBI indekss – klasterēšanas algoritmu metrika (*Davies-Bouldin index*).

GA – ģenētiskie algoritmi. Heiristika tuvinātu risinājumu atrašanai (*genetic algorithms*).

KNN – k tuvāko kaimiņu klasifikācijas algoritms, kurš objekta klasi nosaka pēc tā tuvākajiem k kaimiņiem (*k nearest neighbors*).

UCI – Kalifornijas Universitāte, kura atrodas Ērvinas pilsētā, Amerikas Savienoto Valstu Kalifornijas štatā (*University of California, Irvine*).

IEVADS

Darbā tiek pētīta klasifikācijas problēma. Specifiski tādi klasifikācijas problēmas risinājumi, kuri pielieto ģenētiskos algoritmus (GA). Klasifikācijas problēmā ir dati, kuri ir jāsagrupē. Tipiski grupēšana notiek klasēs. Mēdz izšķirt treniņa datus un testa datus. Parasti treniņa dati satur objektus, kuriem klase ir jau zināma, bet testa dati objektus, kuriem klase ir jānosaka. Līdz ar to no treniņa datiem ir jāiemācās grupēt, klasificēt testa datus. GA ir heiristika, kura var palīdzēt atrast labākus risinājumus klasifikācijas problēmai, tos kombinējot.

Darba mērķis ir izpētīt GA izmantošanas iespējas klasifikācijā.

Darba uzdevumi ir:

1. Izpētīt veiktos pētījumus par klasifikāciju, kuri izmanto GA;
2. Izstrādāt autora GA atribūtu selekcijas risinājumu klasifikācijas problēmai;
3. Izstrādāt autora GA klasifikācijas risinājumu;
4. Izstrādāt autora GA ansambļa klasifikācijas risinājumu;
5. Novērtēt izstrādāto uz publiski pieejamām datu kopām.

Maģistra kursa darbā autors veica literatūras apskati, izstrādāja un veica eksperimentus ar GA atribūtu selekcijas risinājumu klasifikācijas problēmai [2]. Maģistra darbā ar atribūtu selekcijas risinājumu tika veikti papildus eksperimenti (ar visu trenēšanas kopu), kā arī tika izveidoti abi klasifikācijas risinājumi un tika veikti tālāki eksperimenti ar tiem.

Problēma ir aktuāla, kad nepieciešams gūt labumu no rīcībā esošajiem datiem. Klasifikācija ir viens veids, kā veikt datu analīzi. Praktiski mūsdienās dati ir visur esoši un līdz ar to šai problēmai ir liela nozīme. To, ka problēma ir aktuāla un ka ir aktuāli to risināt, izmantojot GA, vislabāk parāda veiktā literatūras analīze, kuras laikā tika identificēti vairāki jauni pētījumi, kuri izmanto GA palīdzību klasifikācijas problēmas risināšanā.

Tēma galvenokārt tika izvēlēta, ņemot iedvesmu no maģistratūras kursa "Datizrace". Autors izpētīja, ka ir brīvi pieejamas atvērto datu kopas ar kurām var veikt eksperimentus un ka ir pietiekoši daudz jaunu materiālu, kurus apskatīt.

Protams, nozīmīga loma tēmas izvēlē bija arī autora bakalaura darbam par GA [1]. Veiktais pētījums bija par citu problēmu (ceļojošā tirgoņa problēma), bet tajā arī tika izmantoti GA. Maģistra darbā ir apskatīta cita problēma, kurai ir nepieciešams cits risinājums. No bakalaura darba maģistra darbā ir tikai aizņemtas pāris rindkopas, lai aprakstītu GA otrajā nodaļā [1].

Maģistra darbā izmantotās metodes ir sekojošas: literatūras izpēte dažādos avotos (raksti zinātniskos žurnālos un konferencēs, grāmatas, tehniskie dokumenti, internets), risinājumu izstrāde valodā Java un labi zināmu datu kopu izmantošana eksperimentos.

Darba struktūra sākas ar izklāstu par klasifikācijas problēmu. Tam seko ģenētisko algoritmu pārskats, veiktā literatūras analīze, izstrādātais GA atribūtu selekcijas risinājums klasifikācijai, izstrādātais GA klasifikācijas risinājums, izstrādātais GA ansambļa klasifikācijas risinājums, rezultāti, secinājumi, izmantotā literatūra un avoti, pielikumi (autora izstrādātais kods un tā darbināšanai nepieciešamais, koda piemēri) un dokumentārā lapa.

1. KLASIFIKĀCIJAS PROBLĒMA DATU ANALĪZEI

Tiek paredzēts, ka datu paliks tikai vairāk un vairāk [6, 19, 23]. Palielinās izveidotās informācijas daudzums un savienoto ierīču skaits. Datus veido ne tikai paši lietotāji, bet arī ierīces, uzņēmumi utt. [6, 19, 23]. Līdz ar to aktuāls ir jautājums, ko var darīt ar datiem, kā no tā gūt labumu. Klasifikācija ir viens variants, kā analizēt datus.

Vēlēšanās veikt klasifikāciju ir diezgan dabiska un visuresoša. Vienmēr varam veikt dažādus novērojumus. Piemēram, kvalitatīvi novērojumi: īss, garš, auksts, ass un biezs. Var būt arī kvantitatīvi novērojumi: 50, 30, 0.2, -8. Teiksim pēc visiem šiem novērojumiem ir veikta analīze un katrai entītijai ir papildus pievienots viens no diviem apzīmējumiem: bīstams vai drošs. Tam varētu sekot vēlme spēt pēc novērojumiem uzreiz pateikt, vai entītija ir bīstama vai droša. Atbilstoši bīstams un drošs ir iespējamās klases, kuras varētu klasificēt.

Klasifikāciju var izmantot dažādos veidos. Varbūt var klasificēt lietotājus pa to klasēm, vai mērījumi norāda uz kādas slimības eksistenci, vai klasificēt kādas darbības (pirkt vai nepirkt akcijas, dot vai nedot kredītu). Jo vairāk datu ir, jo vairāk iespēju paveras.

Klasifikatora, kas veic klasifikāciju definīciju piedāvā [15]. Klasifikatora formula ir redzama (1.1). Viens punkts datu telpā tiek pārveidots par klasi [15]. Protams, problēma ir spējā izveidot tādu formulu, kas strādā visos gadījumos – vienmēr piešķir pareizo klasi. Pat labi klasifikatori visticamāk nebūs 100% precīzi.

$$D : R^n \rightarrow \Omega, \qquad (1.1)[15]$$

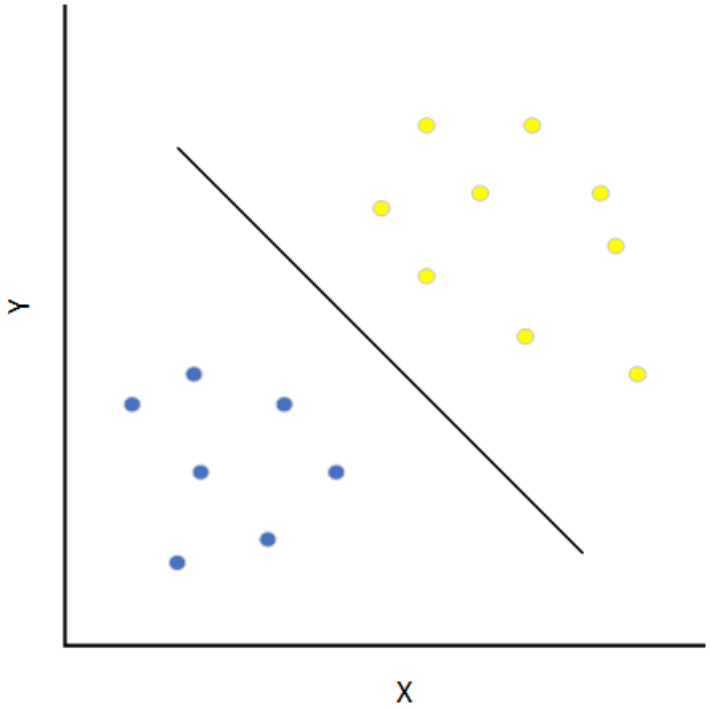
D – klasifikatora funkcija,

$R^n -$ reālo koordinātu telpa (vairākas reālas vērtības vienā punktā),

$\Omega -$ klašu kopa

Var izšķirt reģionus, kuros esošos punktus klasifikators klasificē konkrētai klasei. Protams, reģionā var nonākt punkti no nepareizās klases, kas rezultējas kļūdā. Bet teorētiski, ja vien nav identisku punktu, tad reģionus var sadalīt citādāk, lai kļūdu novērstu [15]. Te jāmin pārliekas pielāgošanās situācija, ja klasifikators dabū 100% precizitāti ar trenēšanas datiem, tad tas negarantē 100% precizitāti ar testēšanas datiem. Klasifikators ir iemācījies lokālu troksni un samazinājis spēju vispārināt nepieciešamās īpašības [14]. Tātad trenēšanas datos minētos punktus no citām klasēm nav nepieciešams vienmēr obligāti noklasificēt pareizi.

Vienkāršā gadījumā ir divi novērojumi x un y, kā arī divas klases k_1 un k_2. Kā redzams 1.1. att. **Klasifikācijas piemērs divās dimensijās un klasēs** ir dzeltenā un zilā klase, kur katru punktu raksturo tā x, y vērtības. Atbilstoši klases var nodalīt, novelkot taisni starp tām. Rezultātā sanāk divi klasificējošie reģioni. Ja pienāk jauni dati, tad klasificēšanu veic, paskatoties, kurā pusē taisnei punkts nonāk – ja tas ir virs taisnes, tad punkts pieder dzeltenajai klasei, bet ja zem, tad zilajai.



1.1. att. **Klasifikācijas piemērs divās dimensijās un klasēs:**

(X, Y) dimensijas, (dzeltenie, zilie punkti) klases

Jāievēro, ka katru punktu var raksturot vairāk nekā divi novērojumi, tad var rasties problēmas tos vizualizēt. Pie tam var būt vairāk par divām klasēm, tad tikai ar vienu taisni nepietiek.

Uz vienu no problēmas paveidiem norāda [15], kad entītijām nav apzīmējumu. Tas ir nav zināms, kādām klasēm pieder dati. Šajā gadījumā uzdevums ir ieraudzīt struktūru datos un noteikt iespējamās klases. Avots [15] norāda, ka šajā gadījumā rezultāta novērtējums ir tīri subjektīvs – nav klašu apzīmējumu pret kuriem pārbaudīt [15]. Autors uzskata, ka galvenais veiksmes faktors ir struktūras pamanīšana un secinājumu veikšana. Pat samērā vāja struktūra var uz kaut ko norādīt – varbūt uz kaut ko, ko ir vērts papētīt dziļāk.

Vēl viens problēmas paveids praktiski nāk no otras puses [10]. Katram objektam varētu būt vairāk par vienu klasi, kas tam pievienota. Grūtības sagādā vairākās apzīmējumu kombinācijas, iespējamās atkarības starp tiem un augsta dimensionalitāte. Nepieciešamas arī citādākas rezultātu izvērtēšanas metodes, jo tagad parādās iespēja klases noteikt daļēji pareizi,

nevis tikai pareizi vai nepareizi [10]. Šāda pieeja noder situācijās, kad objektam var būt vairākas birkas un ir vajadzība paredzēt visas šīs birkas (klases).

Iespējams arī, ka klases ir sadalītas hierarhiski – mazāka līmeņa klases ir zem augstāka līmeņa klasēm. Hierarhiskā struktūrā klases ir sadalītas orientētā grafā bez cikliem vai kokā, kuru veido "ir" attiecības (A klase ir arī B klase). Šīs attiecības ir asimetriskas, transitīvas, kā arī klase nevar būt attiecībā pati pret sevi. Pastāv saistība ar iepriekš minēto paveidu (vairākas klases objektam), jo katru lapu var klasificēt ar visām klasēm, kas hierarhiski atrodas virs tās [25]. Hierarhisku klasifikatoru paveidu ir daudz, bet šajā ziņā ir nepieciešams lietderīgi izmantot datu struktūru, piemēram, izvēloties kādā dziļumā veikt klasificēšanu, vai kā labāk problēmu sadalīt mazākās problēmās.

2. ĢENĒTISKIE ALGORITMI KLASIFIKĀCIJAS UZLABOŠANAI

Šajā nodaļā autors sniedz īsu aprakstu par ģenētiskajiem algoritmiem (GA). Vairāk par tiem var uzzināt autora bakalaura darbā [1].

Šī nodaļa ir sadalīta divās apakš nodaļās. Pirmā apakš nodaļa sniedz ieskatu GA pamata darbībā, bet otrā apakš nodaļa apraksta, kā GA var palīdzēt ar klasifikācijas rezultātu uzlabošanu.

2.1. Ģenētisko algoritmu pamata darbības apraksts

Autors vispirms šeit piedāvā dažas rindkopas no sava bakalaura darba, kuras apraksta GA pamata idejas [1]:

"Algoritmā parasti vispirms uzģenerē nejaušus indivīdus (problēmas risinājumus) un tad mēģina tos uzlabot, atrast labākus. Programmēšanas ziņā visticamāk, ka indivīds ir objekts, kas satur iespējamu, bet ne optimālu problēmas risinājumu, kā arī tā derīgumu (*fitness*), kas ir novērtējums, cik labs, optimāls šis risinājums ir. GA izmantoto indivīdu kopumu sauc par populāciju. Indivīdus var dažādi reprezentēt. Piemēram, indivīds var būt binära virkne, naturālu skaitļu virkne, koks, matrica utt. Reprezentācijas izvēle ir atkarīga no izvirzītās problēmas.

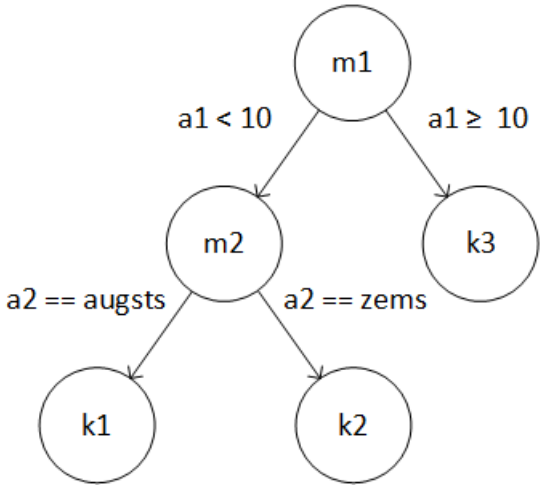
Balstīšanās uz naturālo selekciju tiek nodrošināta, ka vairākos ciklos (paaudzēs) vienmēr tiek izmantota selekcija (indivīdu atlase), lai izvēlētos indivīdus uz ģenētiskiem pārveidojumiem, kurus nodrošina mutācijas un krustmijas (*crossover*) operatori, balstoties uz indivīdu derīgumu. Selekcija ir mehānisms, kas izvēlas indivīdus, bet mutācija un krustmija ir operatori, kas veic izmaiņas indivīdos. Pārveidotie indivīdi nokļūst nākamajā paaudzē.

Katram indivīdam arī katrā paaudzē rēķina tā derīgumu. Tas ir pamata rādītājs, kurš izsaka, ko ar indivīdu darīs iekš GA. Derīgumu parasti izsaka kā reālu skaitli. Tiesa, kā tieši iegūst šo skaitli ir stipri atkarīgs no izvēlētās problēmas. Derīgumam ir jāatspoguļo, cik tuvu dotais risinājums ir nepieciešamajam risinājumam, ja šis rādītājs ir nepareizi izteikts, tad GA var nedot nekādu labumu. Respektīvi tas ir kritiski svarīgs." [1]

Tālāk tiks aprakstīts, kā GA pamata principi varētu izskatīties klasificēšanas problēmā, kura ir apskatīta maģistra darbā. Šajā gadījumā izveidotie GA indivīdi ir pielietojami, lai noteiktu datu ierakstu iespējamās klases.

Klasifikācijas risinājums var tikt izteikts ar lēmumu koku (lēmumi mezglos noved līdz klasei lapā) [15]. Šādu risinājumu var izmantot GA, lai veidotu jaunus, labākus risinājumus (lēmumu kokus) vairākos ciklos, izpildot GA operatorus. Attiecīgi gala rezultāts būtu lēmumu koks ar lielāko derīgumu [20].

Iespējama klasifikācijas indivīda piemērs ir dots 2.1. att. **Klasifikācijas indivīds lēmumu koka formā**. Šī lēmumu koka (indivīda) mezglos ir nosacījumi, kuri nosaka, kur iet pēc konkrētām atribūtu a vērtībām. Tie var būt dažādi, piemēram, skaitliski ($a_1 < 10$) vai kategoriski ($a_2 == augsts$). Lapās ir klases (k_1, k_2, k_3), kuras nosaka attiecīgos lēmumus piešķirt ierakstiem datos konkrētas klases. Tātad, lai noteiktu datu ieraksta klasi pietiek izstaigāt koku, sākot no tā saknes, līdz kādai lapai (lēmumam). Piemēram, datu ieraksts ar atribūtiem $a_1 = 5$ un $a_2 = augsts$ 2.1. att. **Klasifikācijas indivīds lēmumu koka formā** tiktu klasificēts klasē k_1.

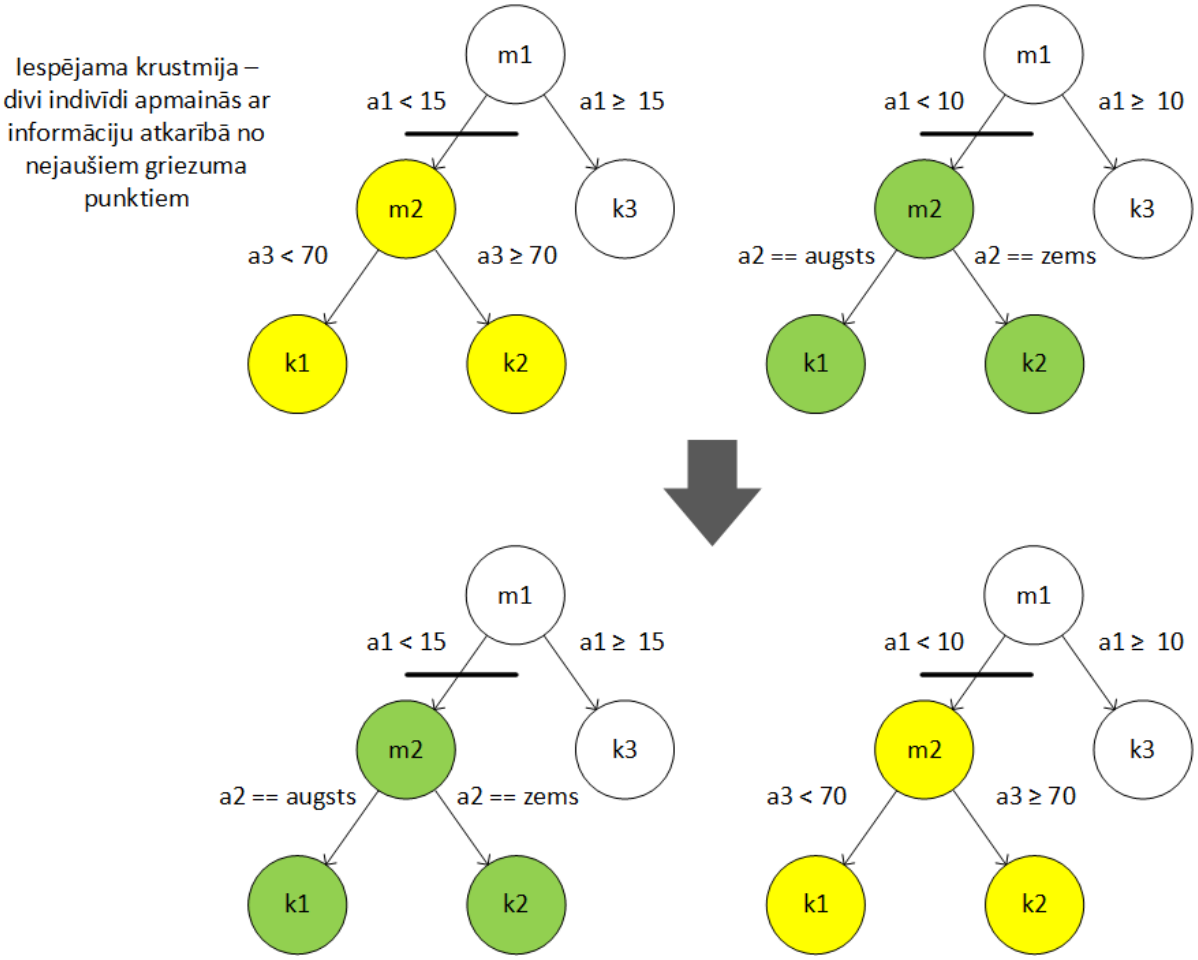


2.1. att. **Klasifikācijas indivīds lēmumu koka formā:**

(k) iespējamās klases koka lapās, (m) mezgli, (a) atribūti

Šāda indivīda derīgums varētu būt izteikts ar pareizi klasificēto datu ierakstu daļu attiecībā pret visiem datu ierakstiem treniņa kopā jeb precizitāti. Tādā gadījumā GA selekcijas operators pārsvarā atlasītu indivīdus ar augstu precizitāti, lai ar tiem veiktu krustmiju un mutāciju [20].

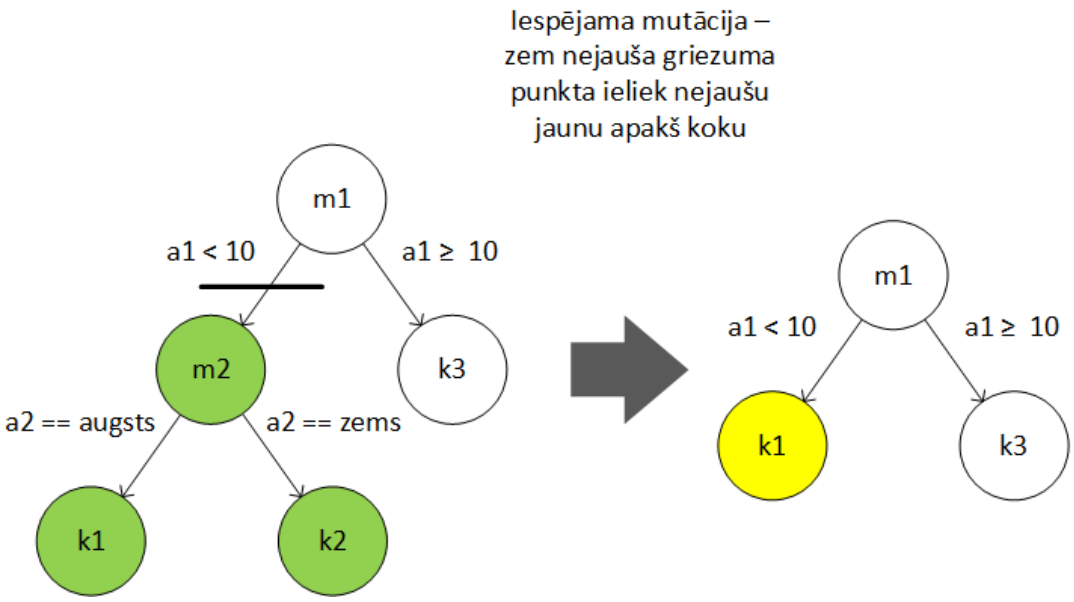
Krustmija varētu ņemt divus lēmumu koku indivīdus un izvēlēties kādu šķautni katrā no tiem. Tad attiecīgi, lai izpildītu krustmijas operāciju un sakombinētu šos indivīdus, viss, kas atrodas zem šīm šķautnēm (griezuma punktiem), tiktu samainīts vietām, tā izveidojot divus jaunus lēmumu kokus [20]. Šīs krustmijas piemērs ir redzams 2.2. att. **Krustmijas piemērs ar griezuma punktiem lēmumu koka šķautnēs**.



2.2. att. **Krustmijas piemērs ar griezuma punktiem lēmumu koka šķautnēs:**

(k) iespējamās klases koka lapās, (m) mezgli, (a) atribūti

Mutācija no viena lēmumu koka izveidotu citu lēmumu koku, kurš atšķirtos no sākotnējā. Tātad mutācija lēmumu kokā varētu izvēlēties vienu nejaušu šķautni, izdzēst visu, kas atrodas zem tās, un tukšajā vietā ielikt nejauši izveidotu citu apakš koku. Attiecīgi varētu būt arī noderīgi ģenerēt tukšus apakš kokus, lai simulētu dzēšanu. [20]. Mutācijas piemērs ir redzams 2.3. att. **Mutācijas piemērs ar griezuma punktu lēmumu koka šķautnē**.



2.3. att. **Mutācijas piemērs ar griezuma punktu lēmumu koka šķautnē:**

(k) iespējamās klases koka lapās, (m) mezgli, (a) atribūti

GA savas izpildes laikā veidotu indivīdu populāciju ar iepriekš minētajām operācijām vairākos ciklos, katrā ciklā populāciju izveidojot par jaunu. Tādā veidā tiek īstenota indivīdu ar lielāku derīgumu meklēšana [20].

2.2. Ģenētisko algoritmu sniegtās klasifikācijas uzlabošanas iespējas

Pirmkārt, kā bija redzams iepriekšējajā apakš nodaļā, GA klasificēšanu var veikt paši. Attiecīgi GA varētu uzdot meklēt lēmumu kokus, kuriem ir lielāka precizitāte. Citādi GA var veikt uzlabojumus jau eksistējošos risinājumos, vai kombinēt abas pieejas.

Redzams, ka šādi lēmumu koki var būt vairāki, tā reprezentējot iespējamu meklēšanas telpu. Lielas meklēšanas telpas pilnīga izstaigāšana var nebūt vēlama, vai pat iespējama. GA ir piemēroti paralēlai apstrādei, kas ļauj ātrāk pārmeklēt lielu meklēšanas telpu. Tas attiecīgi var būt noderīgi klasifikācijas procesam [20].

GA algoritmi var uzlabot neironu tīklus, piemēram, optimizējot to svarus. Avots [16] min, ka GA, kurš optimizē neironu tīklus, var paātrināt iespējamo risinājumu atrašanu [16].

Pamatā klasifikācijas algoritmiem ir savi parametri, kuri ietekmē to darbību un līdz ar to arī to rezultativitāti. GA var pielietot šo parametru optimizācijai. Protams, tas ir atkarīgs no regulējamo parametru daudzuma un konkrētās situācijas, bet to iespējamās kombinācijas var veidot apjomīgu meklēšanas telpu. Piemēram, pētījums [9] ir veicis secinājumu, ka nejaušo mežu algoritma parametri kopā ar klašu dekompozīcijas iestatījumiem veido lielu meklēšanas

telpu un līdz ar to, ka GA izmantošana ir labs risinājums. Tiek arī argumentēts, ka GA labi strādā situācijā ar daudziem lokāliem optimumiem [9].

Ir iespējami arī dažādi citi klasifikācijas uzlabošanas varianti ar GA. Sekojošajā 3. nodaļā ir analizēti vairāki pētījumi, kuri izmanto GA klasifikācijas procesā.

3. ĢENĒTISKO ALGORITMU IZMANTOŠANA KLASIFIKĀCIJĀ

Šeit ir apskatīti vairāki pētījumi par GA izmantošanu klasifikācijā. Pētījumi ir sadalīti pa četrām apakšnodaļām atkarībā no to tā, kā tie izmanto GA: lēmumu koku optimizēšanai, ansambļa tipa struktūras atbalstīšanai, neironu tīklu optimizēšanai un atribūtu selekcijai. Šīs nodaļas piektajā apakš nodaļā ir dots kopsavilkums par veikto literatūras analīzi.

Tiek vērsta uzmanība uz to, kādi dati tika klasificēti, kā risinājumos tika izmantoti GA, risinājumu galvenās iezīmes un kādi bija rezultāti.

Autors, analizējot atsevišķos pētījumus, ievēro vienotu formātu: pirmā rindkopa raksturo pētījumu, bet otrā rindkopa satur autora domas par pētījumu.

3.1. Lēmumu koku optimizēšana ar GA

Šajā apakš nodaļā ir aprakstīti risinājumi, kuri izmanto GA, lai optimizētu lēmumu kokus. Lēmumu koks ir hierarhiska struktūra, kuru tipiski attēlo koka grafa veidā. Pamatā klasifikācijas ziņā lēmumi mezglos ved līdz kādai iespējamai klasei lapā. Lēmumi ir tie, kas šķeļ datus dažādās kopās pēc to atribūtiem līdz tiek sasniegts gala rezultāts [15].

3.1.1. Mobilo lietotāju klasificēšana, optimizējot izvilktos noteikumus no C4.5

Mobilos lietotājus klasificē pētījumā [18]. Tiek piedāvāts risinājums, kas no C4.5 algoritma izveidotā lēmumu koka izvelk noteikumus, kurus uzlabo GA, lai pēc tam labākos no tiem izmantotu klasificēšanā. Tiek piedāvāts risinājumu kodējums šim nolūkam, kā arī labuma izvērtēšanas funkcija, kas balstīta uz četrām metrikām. Izveidotais risinājums sniedz labākus rezultātus nekā oriģinālie C4.5 un atbalsta vektora mašīnas algoritmi gan uz mobilajiem lietotājiem, gan “Iris” un “Breast-cancer” datu kopām. Papildus augstākai precizitātei tiek iegūti vieglāk saprotami, cilvēk-lasāmi likumi, kas veic klasificēšanu.

Izvēlētā pieeja parāda iespēju izmantot jau gatavus kokus un kā tos novērtēt. Indivīdu vērtēšanas metode veic vairākus būtiskus mērījumus. Vienkāršie likumi dot iespēju veikt papildus analīzi, piemēram, kuri atribūti ir nozīmīgākie klasificēšanā. Autors domā, ka trūkums pētījumā ir tikai C4.5 veidoto lēmumu koku izmantošanā. Lai gan raksta autori norāda uz iespējām izmantot citus lēmumu kokus, tas netiek pārbaudīts. Varētu izvērtēt arī

iespēju izmantot vairākus lēmumu kokus vienlaicīgi. Nebija skaidrs, kas tiek darīts ar populācijas izmēru, jo tas netika minēts kā parametrs un algoritmā šajā ziņā ir tikai teikts, ka inicializē populāciju. Izskatās, ka varētu vai nu populācijas izmēru pielāgot C4.5 atrasto likumu skaitam, vai otrādi – C4.5 atrast tik likumu, lai aizpildītu noteikto populāciju. Šis jautājums rakstā netika izskatīts [18].

3.1.2. Tirgošanās sistēmas noteikumu noteikšana ar lēmumu kokiem

Tirgošanās sistēmas noteikumus analizē pētījumā [11]. Pēc datiem attiecīgi tiek klasificētas pirkt, pārdot vai turēt darbības. Tiek izmantoti vēsturiskie ASV IT sektora 5 akciju dati (2007.-2015. gads trenēšanai un 2016. gads testēšanai), lai simulētu tirgošanos, kas attiecīgi ļauj izvērtēt GA indivīdu (lēmumu koku) derīgumu. GA tiek izmantoti, lai optimizētu pirkt/pārdot/turēt lēmumu kokus. Katrā jaunajā GA paaudzē tiek noņemti liekie koku mezgli, kur var būt neizpildāmas prasības (piemēram, ja a > 30 uzreiz seko a < 15). Īpaša nozīme tiek pievērsta tirgus dinamiskumam. Pašos kokos izmanto rādītājus, kas skatās uz d iepriekšējām dienām (piemēram, augstākā, zemākā vērtība laika periodā). Tos izsaka ar 5 graudainības pakāpēm ("very low, low, medium, high, very high"), kas balstītas uz iepriekšējajiem datiem (pašā zemākajā pakāpē būs vēsturiski zemākie dati). Rezultātā tiek gūts mazs gadskārtējs guvums un cilvēk-saprotami tirgošanās likumi.

Autors domā, ka pētījuma galvenais pienesums ir tirgus dinamikas izmantošana ar rādītāju palīdzību. Raksta autori arī norāda, ka graudainības pakāpes var izmantot arī citos lietojumos, ne tikai akciju tirgū – tā kā šo ir vērts atcerēties. Galvenā pieejas kritika ir nepieciešamībā labot izveidotos kokus – rodas situācija, ka bez labojumiem izvērtēšana patērē vairāk laika, bet arī paša labošana prasa ievērojamu laiku. Arī tas, ka tiek izmantoti tikai 3 rādītāji (paši raksta autori arī uzskata, ka varētu atrast vairāk rādītāju [11]). Visbeidzot, autors komentē, ka rakstā bija minēts, ka galējais lēmums (pirkt/pārdot/turēt) vēl ir atkarīgs no pozīciju pārmaiņu tabulas (pozīcijas – īss, garš, ārpus pozīcijas), bet raksta autori to piemin tikai vienreiz un nekomentē, kā viņi ir nonākuši pie tādas tabulas, kā arī šī tabula nav uzrādīta galvenā algoritma aprakstā [11].

3.1.3. Klasifikācijas noteikumu attīstīšana ar vairāku populāciju GA

Vairāku populāciju GA klasificēšanai tiek piedāvāti pētījumā [24]. Tiek klasificēts ievērojams skaits datu kopu (pavisam 10 [24]). GA tiek izmantoti, lai atrastu klasifikācijas

noteikumus ar kuriem tiek veikta paša klasificēšana. Pamata ideja pieejai ir palielināt GA risinājumu dažādību, kā arī nodrošināt iespēju atsevišķās indivīdu populācijas attīstīt paralēli. Pētījumā sākumā tiek izveidotas 5 populācijas [24], kuras vispirms tiek attīstītas atsevišķi, tad tiek izmantots migrācijas operators, lai pārvietotu indivīdus starp populācijām un visbeidzot savienošanas operators sakopo noteikumu kopumu no visām populācijām, izmetot liekos noteikumus. Papildus tam realizētie pamata GA (iekš katras populācijas) izmanto drūzmēšanās tehniku, kurā populācija katrā paaudzē, nevis tiek veidota par jaunu, bet tiek mainīta, jaunajiem indivīdiem no populācijas izmetot sliktākos un tiem līdzīgākos indivīdus. Šajā ziņā līdzīgumu balsta uz izmantoto atribūtu skaita, kas ir vienādi starp indivīdiem. Migrācijas operators ir samērā vienkāršs – starp populācijām tiek samainīti labākie indivīdi, ja vien tas dot uzlabojumu. GA paaudžu beigās savienošanas operators visus atrastos noteikumus ieliek vienā populācijā, bet izmet tos, kas ir kāda cita noteikuma apakškopa. Rezultātā tika noskaidrots, ka izveidotais risinājums pārspēj parastos GA ar drūzmēšanās tehniku, kā arī tika piedāvāti optimālie migrācijas parametri.

Pētījums parāda, ka populācijas dalīšana vairākās var palīdzēt ar ne tikai īsāku izpildes laiku, bet arī ar labāku precizitāti. Pētījuma trūkums ir salīdzināšana tikai ar vienu citu GA pieeju un arī tas, ka citi nozīmīgi parametri netiek analizēti (piemēram, populāciju daudzums). Drūzmēšanās GA paši par sevi ir domāti, lai palielinātu populācijas dažādību, bet redzams, ka kombinācijā ar vairākām populācijām rezultāts ir vēl labāks. Tas liek domāt, ka indivīdu dažādībai ir jāpievērš liela uzmanība. Tas arī norāda uz to, ka būtu bijis noderīgi pētījumā veikt arī salīdzināšanu ar parastajiem GA bez drūzmēšanās. Autors arī norāda, ka nebija skaidrs, kas tiek darīts situācijā, ja izvēlētie noteikumi dot dažādus rezultātus. Tādā ziņā, ka viens noteikums saka, ka ievades objekta klase ir X, bet cits noteikums saka, ka tā paša objekta klase ir Y – tad kādu klasi piešķirt objektam. Autors komentē, ka lai paralēli darbinātu GA nav obligāti nepieciešams veidot vairākas populācijas – pietiek paralēli rēķināt GA operatoru darbības. Līdz ar to ir redzams, ka piedāvātajai arhitektūrai ir nozīme precizitātes palielināšanā [24].

3.2. GA izmantošana, lai atbalstītu ansambļa tipa struktūras

Šajā apakš nodaļā ir apskatīti risinājumi, kuri izmanto GA, lai atbalstītu klasifikatoru ansambļa tipa struktūras. Ansambļa tipa struktūrā ir iekļauti vairāki klasifikatori, kuri tiek kombinēti, lai iegūtu gala rezultātu. Cerība ir gala rezultātā iegūt labāku precizitāti apmaiņā pret lielāku sarežģītību [15].

3.2.1. Pret-vēža peptīdu klasificēšana ar divām ansambļa tehnikām

Pret-vēža peptīdus klasificē pētījumā [3]. Pret-vēža peptīdi pašlaik ir izstrādes stadijā, bet cerība ir, ka tie varētu būt kā cilvēka ķermenim mazāk traucējoša alternatīva klasiskajām procedūrām. Analizējamie dati satur proteīnu virknes no divām klasēm (ir/nav pret-vēža peptīds). Tika izņemtas līdzīgās virknes (90% un vairāk). Īpašību vektors tiek sakombinēts kopā no trim dažādām sekvences reprezentācijām, tad tas tiek padots pieciem klasifikatoriem (atbalsta vektora mašīna, varbūtisks neironu tīkls, nejaušie meži, vispārinātās regresijas neironu tīkls, k tuvākie kaimiņi), kas veido ansambli. Ansamblis gala rezultātu izvēlas no divām tehnikām, pirmā ir vienkārša balsošana (izvēlas to klasi par kuru visvairāk balso) un otrā ir GA ansamblis, kas klasifikatoru rezultātiem piekārto svarus (izvēlas maksimālo vērtību no rezultātu un svaru reizinājumiem). Rezultātā izstrādātā pieeja pārspēja citus uz šīs datu kopas veiktos eksperimentus. Individuālo algoritmu ziņā labus rezultātus uzrādīja atbalsta vektoru mašīna, bet kombinēto algoritmu ziņā GA parādīja savu pārākumu.

Pētījumā ir izveidota diezgan apjomīga struktūra, kā arī testēšana ir veikta gan ar visu struktūru kopumā, gan tās atsevišķajiem elementiem. Autors domā, ka pētījums parāda dažādu klasisku algoritmu hibridizācijas nozīmi. Jāievēro arī, ka nejaušo mežu iekļaušana klasifikatoros nozīmē, ka ir izveidota ansambļa metode, kas iekļauj citu ansambļa metodi. Tātad ansambļa metodes var būvēt vienu uz otras. Autors arī norāda, ka rakstā ir minēts, ka gala rezultāts tiek kombinēts ar GA un vienkāršo balsošanu, bet nav specificēts, kādā veidā tas notiek. Ņemot vērā, ka ir divi rezultāti, visticamāk, ka balsošanas starp tiem nav, bet rezultātiem varētu būt vai nebūt piekārtoti svari [3].

3.2.2. Nejaušo mežu algoritma uzlabošana, optimizējot tā parametrus un veicot klašu dekompozīciju

Apjomīgu klasifikāciju (22 datu kopas) veica pētījumā [9]. Šajā gadījumā GA tika pielietoti, lai optimizētu nejaušo mežu algoritma parametrus (koku skaits, izmantoto īpašību skaits) un klašu dekompozīciju (izmantoto klasteru daudzums katrai klasei). Klašu dekompozīcija klases sadala apakšklasēs (klasteri konkrētās klases datos), kas uzlabo datu diversifikāciju. Te jāmin, ka 1 klasteris nozīmē, ka diversifikācija nenotiek – līdz ar to tā izstrādātajā risinājumā nebija vienmēr obligāta. Izvēlētās pieejas darba plūsma ir lielā mērā identiska parastajam GA ar izmaiņām risinājumu izvērtēšanā – risinājums ir minētie parametri

un lai tos izvērtētu tiek palaista dekompozīcija (K-vidējo algoritms) un pēc tām nejaušo mežu algoritms uz sadalītajiem datiem. Risinājums pārspēja nejaušo mežu algoritmu, kā arī AdaBoost algoritmu. Papildus tika parādīts, ka ir labums no klašu dekompozīcijas.

Izvēlētā pieeja parāda parametru optimizēšanas nozīmi. Lai gan klašu dekompozīcija parādīja labākus rezultātus, tie nebija ļoti pārliecinoši – pieeja pārspēja variantu, kas tikai optimizē parametrus nejaušo mežu algoritmam, 12 reizēs no 22 reizēm. Autors komentē, ka rakstā tika uzsvērta labas derīguma funkcijas nepieciešamība, bet pašā tekstā nav ne reizi minēts kādu derīguma funkciju izmantoja paša raksta autori. Citādi pētījumā ir veikta apjomīga un nozīmīga analīze [9].

3.2.3. Uzņēmumu bankrota klasificēšana ar klasifikatoru ansambli divos soļos

Uzņēmumu iespējamo bankrotu klasificē pētījumā [26]. Klasificēšana notiek divās klasēs "bankrots" un "sekmīgs". Analizējamā datu kopa sastāv no 912 Krievijas uzņēmumiem [26]. Šajā pētījumā GA izmanto divās fāzēs – vispirms, lai atlasītu atribūtus individuālajiem klasifikatoriem, tad lai optimizētu svarus izveidoto klasifikatoru ansamblim (no klasifikatoriem pirmajā solī). Pirmā soļa individuālie klasifikatori ir sekojoši: k tuvākie kaimiņi, naivais Beijess, loģistiskā regresija, lēmumu koki un atbalsta vektora mašīna. GA indivīds šajā solī satur visus atribūtus, kuriem pretī ir 1 (atribūtu izmanto) vai 0 (atribūtu neizmanto). Katru klasifikatoru trenē ar vairākām datu kopas apakškopām (rezultāts ir vidējais starp visām apakškopām). Otrajā solī GA indivīds ir reālās svaru vērtības katram klasifikatoram. Svaru un klasifikatoru rezultātu reizinājumu summa nosaka noteikto klasi. Šajā solī trenē uz vienas trenēšanas kopas. Rezultātā tika parādīts, ka atribūtu selekcija uzlabo katra individuālā klasifikatora rezultātu, kā arī izveidotais ansamblis pārspēja lielu daudzumu citu metožu (17 [26]), kas izmantotas bankrota klasificēšanai. Papildus tam tika parādīta pieejas spēja izvēlēties nozīmīgus atribūtus, novērtējot tos pēc tā, cik klasifikatoros atribūts tika iekļauts. Tika secināts, ka tie atribūti, kuri bija nozīmīgi tikai 1 vai 2 klasifikatoriem, nedot labāku klasifikācijas rezultātu [26].

Pētījumā tika veikta ievērojama salīdzināšana ar citām metodēm un tika parādīta atribūtu selekcijas nozīme – tā palīdzēja pilnīgi visiem izvēlētajiem klasifikatoriem (bija savākti gan finansiāli, gan ārēji rādītāji, kā arī ekonomisko situāciju un uzņēmumu raksturojoši rādītāji). Autors norāda, ka pirmajā solī visi indivīdi tiek iniciēti vienādi (visi atribūti tiek izmantoti – visi vieninieki). Labāk būtu iniciēt nejauši vai citādāk, jo izveidotā sākotnējā populācija ir pilnīgi viendabīga. Ticami, ka tas arī ietekmēja atribūtu selekciju, jo

nav tādu atribūtu, kuru neviens klasifikators neizmanto. Pētījumā tika izmantoti visi pieci klasifikatori. Autors šajā ziņā domā, ka nav noderīgi, piemēram, tērēt resursus tādam klasifikatoram, kuram ir piekārtots svars 0.013 (loģistiskā regresija) [26]. Nākamais lielākais svars ir par gandrīz 10 reizēm lielāks (0.122) [26]. Praktiski šis klasifikators ir pielīdzināms troksnim ansamblī, kas neko nozīmīgu nedot, bet aprēķini klasifikācijai ir jāveic pilnā apmērā [26].

3.3. Neironu tīklu optimizēšana ar GA

Šajā apakš nodaļā ir aprakstīti risinājumi, kuri izmanto GA, lai uzlabotu klasifikāciju ar neironu tīkliem. Neironu tīkli imitē dabā pastāvošus neironu tīklus jeb smadzenes. Tāpat kā smadzenes neironu tīkls sastāv no vairākiem savienotiem neironiem. Neironi izmanto aktivācijas funkcijas, lai apstrādātu pienākošo informāciju [15].

3.3.1. Struktūras sagrūšanas paredzēšana, GA optimizējot vairākus mērķus

Betona struktūras klasificē pētījumā [5]. Tiek izmantota datu kopa ar 150 ēkām, kur attiecīgi ir nepieciešams klasificēt divās klasēs: ir struktūras sagrūšanas un nav struktūras sagrūšanas. Klasificēšanai izmanto neironu tīklu, kura svarus optimizē GA. Izmantotais neironu tīkls ir klasisks vairāk slāņu perceptrona priekšējās plūsmas tīkls. GA ziņā tiek optimizēti divi mērķi, nevis viens. Tātad tipiski ir viena derīguma vērtība, kuru optimizēt, bet šajā gadījumā tiek optimizēta saknes vidējā kvadrāta kļūda un maksimālā kļūda. Līdz ar to atlase GA tiek veikta, grupējot indivīdus pēc Pareto dominēšanas (x dominē y, ja y visi mērķi (kļūdas) ir vienādi vai sliktāki par x un eksistē vismaz viens y mērķis, kas ir sliktāks). Protams, priekšroku dot tiem indivīdiem, kuri netiek dominēti. Pētījumā izveidotais risinājums pārspēja klasisko neironu tīklu un neironu tīklu risinājumu, kurš ir optimizēts ar daļiņu spieta metodi. Tiesa pētījumā izstrādātais laika ziņā strādāja apmēram trīs reizes ilgāk, nekā minētie divi konkurējošie risinājumi.

Pētījums parāda, ka neironu tīklu optimizēšanu var veikt, izmantojot vairākus mērķus. Autors norāda, ka klasiskais neironu tīkls tika ievērojami pārspēts (piemēram, precizitāte bija 80% klasiskajam neironu tīklam pret izstrādātā risinājuma 93% [5]). Daļiņu spieta optimizētais variants gan atpalika tikai par dažiem procentiem (precizitāte bija 90% pret 93% [5]). Autors domā, ka tas paceļ jautājumu, cik tad daudzus mērķus būtu visoptimālāk

izvēlēties. Pārāk daudzi mērķi varētu novest pie tā, ka daļa no tiem ir lieka, bet izskatās, ka lielāks mērķu daudzums ļauj labāk izteikt, kas ir nepieciešams no GA [5].

3.3.2. Elektrokardiogrammu signālu klasificēšana ar neironu tīkliem, veicot dimensiju redukciju

Elektrokardiogrammu signāli tiek klasificēti pētījumā [16]. Šo signālu klasificēšana ir nozīmīga sirds slimību diagnozei. Signāli tiek pavisam klasificēti 6 klasēs ar neironu tīklu, kas izmanto kļūdu atgriezeniskās izplatīšanas metodi. Pētījumā tiek izmantota sirds aritmijas datu kopa, kā arī simulēti signāli. Sākumā tiek veikta iegūto signālu apstrāde, lai mazinātu troksni datos, tad no signāliem tiek izvilkti atribūti ar viļņu pakešu dekompozīcijas statisko metodi. Tikai tad seko GA optimizācija, kas izvēlas atribūtu kopu no iepriekšējā solī izvilktajiem atribūtiem, kā arī optimizē neironu tīkla svarus un noslieces. Visbeidzot neironu tīkls veic klasificēšanu. Rezultātā tika iegūta lielāka precizitāte, kā identiskam neironu tīklam bez GA, atbalsta vektoru mašīnai un GA optimizētai atbalsta vektoru mašīnai. Arī tika pārspētas citas pieejas, kas tika izmantotas uz sirds aritmijas datu kopas – gan precizitātē, gan klašu daudzumā (piedāvātā pieeja signālus klasificē vairāk klasēs, nekā citas pieejas).

Redzams, ka pieejai papildus ir nepieciešams izvilkt atribūtus no elektrokardiogrammu signāliem pirms GA var sākt optimizēšanu (papildus apstrāde). Veiktā optimizēšana ir pamatīga, jo GA tiek pielietoti divās vietās, lai izvēlētos atribūtus un optimizētu neironu tīkla svarus un noslieces. Tas parāda GA nozīmi klasificēšanā. Autors komentē, ka pētījumā tika izmantota signālu simulēšana no ierīces, lai iegūtu datus, bet šie dati tika izmantoti tikai lai salīdzinātu izstrādāto pieeju ar identisku neironu tīklu bez GA. Nav saprotams, kāpēc simulētos datus neizmantoja arī lai salīdzinātu ar citām pieejām. Pētījumā lielākā daļa veiktās analīzes tiek balstīta uz aritmijas datu kopas [16].

3.4. Atribūtu selekcija ar GA tālākai klasifikācijai

Šajā apakš nodaļā ir aprakstīti risinājumi, kuri izmanto GA, lai izvēlētos klasificēšanā izmantojamos atribūtus, tā samazinot izmantojamo datu apjomu.

Autors norāda, ka šajā apakš nodaļā ir iekļauti pētījumi, kuru galvenais fokuss ir atribūtu selekcija (GA tiek izmantoti tikai atribūtu selekcijai). Maģistra darbā ir apskatīti arī risinājumi, kuri GA izmanto dažādiem nolūkiem. Pētījums [26] izmanto GA gan atribūtu

selekcijai, gan klasifikatoru ansambļa struktūras atbalstīšanai [26]. Pētījums [16] izmanto GA gan atribūtu selekcijai, gan neironu tīklu optimizēšanai [16].

3.4.1. Viļņu ciparu noteikšana biomasas degradācijas pētīšanai

Biomasas degradācija tiek pētīta [22]. Pētījumā izmanto kukurūzas sakņu datu kopu. Infrasarkanā spektrometrija izmanto infrasarkano starojumu, lai novērotu vibrācijas biomasā, kas ir degradācijas procesā. Vibrācijas biomasā mēra ar cm^{-1}, kas ir viļņa cipars (šajā gadījumā tiek pētīts intervāls no 800 cm^{-1} līdz 1800 cm^{-1} [22]). Šīs vibrācijas var palīdzēt noteikt molekulāro struktūru. Attiecīgi ir jāatrod tādi cm^{-1}, kas ir noderīgi infrasarkanajai spektrometrijai (vislabāk atšķir īpašības). GA izmanto atribūtu selekcijai tālākai klasifikācijai, lai izvēlētos labākos viļņu ciparus (no tiem sastāv GA indivīds). Attiecīgi pēc tam datus no viļņu ciparіem izmanto, lai kukurūzas saknes sagrupētu pēc to degradācijas stadijām. Pētījumā GA tiek pielietota derīguma funkcija, kas ir balstīta uz naivo Beijesa klasifikatoru – kombinācija no labākās nosacītās varbūtības un Beijesa kļūdas. Beijesa kļūda uzrāda zemāko iespējamo kļūdu klasifikācijas problēmai, tāpēc ja GA uzdot optimizēt pēc iespējas mazāku Beijesa kļūdu, tad tiek meklēts tāds sadalījums, kur klases labi atdalās. GA iegūto rezultātu izmanto galveno komponenšu analīzē, lai noteiktu, cik labi izvēlētie viļņu cipari atdala kukurūzas saknes pēc to degradācijas pakāpēm. Iegūtais klašu sadalījums tika arī izmērīts ar Duna (Dunn) indeksu. Rezultātā tika secināts, ka izstrādātais risinājums pārspēj divus citus GA risinājumus – ar Fišera lineārā diskriminanta derīguma funkciju un DBI indeksa derīguma funkciju.

Pētījums parāda GA ar naivā Beijesa klasifikatora kombinēšanas iespēju, izmantojot derīguma funkciju. Abi algoritmi, nevis tiek darbināti vienlaicīgi, bet GA tiek uzdots meklēt to, ko meklētu naivā Beijesa klasifikators. Autors norāda, ka izstrādātajā risinājumā atrodamo viļņu ciparu skaits ir viens no nepieciešamajiem parametriem, kurš tika noteikts eksperimentu rezultātā katrai testētajai pieejai atsevišķi. Autors komentē, ka, atļaujot indivīdiem būt dažādos garumos, šādi eksperimenti nebūtu jāveic, piemēram, kad parādītos nepieciešamība analizēt citu datu kopu [22]. Balstoties uz [5], noderīgi varētu būt arī izmantot visas trīs izstrādātās derīguma funkcijas vienā klasifikatorā, jo kā tika parādīts no tā var gūt labāku precizitāti [5].

3.4.2. Krūts vēža noteikšana ar K tuvāko kaimiņu algoritmu, optimizējot izvēlētos atribūtus un parametrus

Krūts vēzis tiek klasificēts pētījumā [21]. Atbilstoši tiek izmantota datu kopa ar krūts vēža prognozēšanas datiem. GA tiek izmantoti, lai izvēlētos atribūtus k tuvāko kaimiņu klasifikatoram. GA indivīds ir vieninieku un nuļļu virkne, kas nosaka, vai atribūtu izmantot, vai neizmantot. Pavisam kopā ir 32 atribūti no kuriem izvēlēties. K tuvāko kaimiņu implementācija objektu salīdzināšanai izmanto Manhetenas attālumu, kurš iekļauj tikai horizontālo un vertikālo distanci, nevis diagonālo. Papildus izmantojamajiem atribūtiem eksperimentu rezultātā tiek atrasts arī kaimiņu daudzums, izmantojamais datu kopas apjoms (cik % datu izmantot kā trenēšanas kopu – k tuvāko kaimiņu algoritms izmanto trenēšanas kopu, lai klasificētu neredzētus objektus), krustmijas un mutācijas proporcija. Tiek noteikta verifikācijas stadija, kas pārbauda iegūtos labākos rezultātus, lai atrastu starp tiem trīs labākos. Rezultātā tika noteikts tāds atribūtu un parametru kopums, kas par gandrīz 2% palielina vidējo precizitāti.

Pētījumā tika veikta pamatīga testēšana atribūtu un parametru ziņā, bet ievērojams trūkums ir, ka netika veikta salīdzināšana ar citām pieejām. Pētījums parāda to, ka k tuvāko kaimiņu klasifikatoram nav obligāti izmantot pēc iespējas vairāk datu, lai veiktu klasificēšanu, jo labākais rezultāts tika iegūts tikai ar pārsteidzošiem 30% visu datu [21]. Pētījuma autori saka, ka, izmantojot 90% datu kā trenēšanas kopu (pārējie 10% testēšanai), bieži vien ir liela standart novirze un maza minimālā precizitāte, kas mudināja ieviest parametru, kurš regulē izmantojamo datu kopas apjomu [21]. Autoram šķiet, ka mazākas datu kopas izmantošana, lai iegūtu labāku rezultātu, ir pretēji intuīcijai. Tas paceļ jautājumu, vai arī citiem klasifikatoriem ir noderīgi pamēģināt izmantot dažādu datu apjomu trenēšanas kopām. Autors arī komentē, ka papildus ieguvums ir mazāks laika patēriņš – mazākai datu kopai ir jāpatērē mazāk resursu, lai veiktu modeļa trenēšanu uz tās. Pētījumā būtu bijis noderīgi testēt pieeju arī uz citām datu kopām, jo ir iespējams, ka izvēlētajā datu kopā ir daudz maldinošu datu un ka pārsvarā lielāks trenēšanas datu apjoms nozīmē lielāku precizitāti [21].

3.5. Kopsavilkums par GA izmantojumu klasifikācijā

Šī apakšnodaļa sniedz veiktās literatūras analīzes kopskatu. Tā parāda, kas tiek pētīts GA izmantošanas klasifikācijā jomā, kā arī tiek veikti secinājumi par galvenajām GA izmantošanas pieejām un idejām.

Veiktā literatūras analīze galvenokārt parāda, ka ir aktuāli pielietot GA klasifikācijas problēmas risināšanai, jo ir veikti vairāki jauni pētījumi, kuri savos risinājumos izmanto GA. Pie tam apskatītie pētījumi ir sasnieguši savus mērķus un ieguvuši uzlabotus risinājumus. Tas nozīmē, ka ir vērts veikt pētījumus GA ziņā. GA klasifikācijas jomā ir plašs pielietojums, kā arī GA dot iespējas veidot efektīvākus risinājumus.

Redzams, ka literatūrā GA ir dažādi pielietojumi. Pirmām kārtām GA var izmantot, lai optimizētu lēmumu kokus, kuri nosaka objektu klases. Šajā ziņā GA var konstruēt visu koku [11], vai būtībā izvēlēties koku sadalīt vairākās mazākās daļās – noteikumos (ja izpildās šie nosacījumi, tad objektam ir klase X, citādi šīs klases nav) [18, 24]. Autors domā, ka ērtāk ir izmantot noteikumus, jo pētījumā [11] izveidotie lēmumu koki bija patstāvīgi jālabo. Pie tam noteikumi ir cilvēkiem vieglāk saprotami, nekā koki. No otras puses lēmumu kokiem ir konkrēta kārtība, kā notiek klasificēšana (iet pa koku), bet noteikumiem ir jānosaka sava darba kārtība. Zināms, ka [18] tika izvēlēts ņemt vērā derīgākus noteikumus pirms nederīgākiem (ar lielāku derīguma funkcijas vērtību) [18]. Autors domā, ka varētu būt risinājums, kurš veido kaut ko līdzīgu noteikumu ansamblim. Teiksim, ja divi noteikumi apgalvo, ka A klase ir X, bet viens apgalvo, ka A klase ir Y, tad var klasificēt A klasi kā X. Visādā ziņā GA var arī pielietot kopā ar ansambļa tipa klasifikāciju, kurā ir vairāki klasifikatori un kurā dažādie klasifikācijas spriedumi tiek kontrolēti ar kādu metodi (piemēram, balsošanu).

GA var izmantot, lai atbalstītu klasifikatoru ansambļa struktūru. Naturāli ir pielikt svarus klasifikatoriem, tā kontrolējot to nozīmi risinājumā [3, 26]. Cita ideja ir ņemt jau gatavu ansambli un optimizēt tā parametrus [9]. Lielā mērā redzams, ka GA citu algoritmu parametrus varētu optimizēt daudzos pielietojumos (ne tikai ansambļa). Autors klasifikatoru ansambļu ziņā domā, ka ir grūti izvēlēties no tieši kādiem klasifikatoriem tas sastāvēs – savā ziņā varētu teikt, ka precīzā izvēle ir gandrīz kā aksioma. Protams, klasifikatoru svari regulē ko cik lielā mērā izmantot, bet diez vai testēšana tikai ar ~5 klasifikatoriem dot optimālo ansambli, kā arī empīriski apsvērumi dot tikai daļēju optimizāciju. Noteikti varētu apsvērt iespēju iekļaut vairāk klasifikatoru kā tas tiek norādīts [26], lai iegūtu lielāku dažādību. Tos

klasifikatorus, kuriem tiek piekārtoti mazi svari gala risinājumā varētu neizmantot. Protams, var arī detalizētāk analizēt kādi svari tika iegūti.

Gana bieži sastopama ideja ir pielietot GA, lai optimizētu neironu tīklus. Redzams, ka apskatītajos pētījumos tika optimizēti neironu tīklu svari [5, 16]. Autors domā, ka šajos pētījumos netiek ievērota ideja paplašināt optimizējamos parametrus. Piemēram, neironu skaits, slāņu skaits un aktivizācijas funkciju veidi utt. visi ir parametri, kuru noteikšana nav triviāla. Protams, neironu tīkls ir arī noderīgs klasifikators, kuru ievietot iekš ansambļa, kā tas tiek darīts pētījumā [3].

GA var pielietot atribūtu selekcijai tālākai klasifikācijai. GA dot iespēju veikt vienkārši realizējamu un efektīvu atribūtu selekciju, kas ir nepieciešama tehnika vairākos gadījumos. Veiktā literatūras analīze liecina, ka GA tiek bieži lietoti atribūtu selekcijai, piemēram, pētījumi [16, 26] praktizē GA pielietošanu divās vietās – gan atribūtu selekcijai, gan tālākai optimizācijai [16, 26]. Autors domā, ka, risinot klasifikācijas uzdevumus, vienmēr var apsvērt GA pielietošanu. Lai gan tas nav patiesi pilnīgi visos gadījumos, ir ticams, ka atribūtu būs daudz un ka ne visi atribūti ir nepieciešami klasifikācijai.

Visbeidzot, var secināt, ka GA pētījumos tiek pielietoti, lai optimizētu lēmumu kokus, neironu tīklus, atbalstītu klasifikatoru ansambļa struktūru un lai veiktu atribūtu selekciju.

4. IZSTRĀDĀTAIS ĢENĒTISKO ALGORITMU ATRIBŪTU SELEKCIJAS RISINĀJUMS KLASIFIKĀCIJAS PROBLĒMAI

Šajā nodaļā ir aprakstīts autora maģistra kursa darbā izstrādātais GA atribūtu selekcijas risinājums klasifikācijas problēmai un attiecīgie veiktie eksperimenti [2]. Ir sniegtas dažādas idejas, kā arī ir aprakstīts, kas tika paveikts.

Maģistra darbā tika veikti papildus testi. Testēšanas rezultāti tika iegūti, veicot trenēšanu ar visiem trenēšanas kopas datiem, nevis labākās šķērsvalidācijas kopas datiem, kā iepriekš. Šie rezultāti ir redzami 4.4. tabulā, kā arī attiecīgajās vietās pēc tabulas un 4.4. apakš nodaļā ir pievienoti papildus komentāri par iegūtajām izmaiņām. Tika arī izlabota kļūda, kas samazināja Pilsētas zemes tipu datu kopas [12, 13, 17] testēšanas rezultātus.

Pirmā apakšnodaļa definē sākotnējās risinājuma pamata idejas. Otrā apakšnodaļa izklāsta izstrādāto risinājumu. Trešā apakšnodaļa ir par veiktajiem eksperimentiem.

4.1. Galvenās idejas risinājuma izstrādei

Šeit ir iezīmētas idejas kāds varētu izskatīties gala risinājums un ar ko varētu sākt izstrādi.

Klasificējamo datu ziņā var izmantot atvērto UCI mašīnmācīšanās repozitoriju [17]. Šis repozitorijs ir labi zināms un ir pielietots citos pētījumos, kā arī ir brīvi pieejamas vairākas datu kopas no dažādām nozarēm [17].

Izstrādājamajā risinājumā var izmantot funkcionalitāti no WEKA [8], kuru var izsaukt no Java koda. Piemēram, var pielietot jau gatavus algoritmus (C4.5, Naivais Beijess), vai pielietot testēšanas funkcionalitāti. Līdz ar to risinājums tiktu izstrādāts valodā Java. Risinājuma struktūru un GA izstrādātu autors, izsaucot WEKA API, kur tas būtu noderīgi [8].

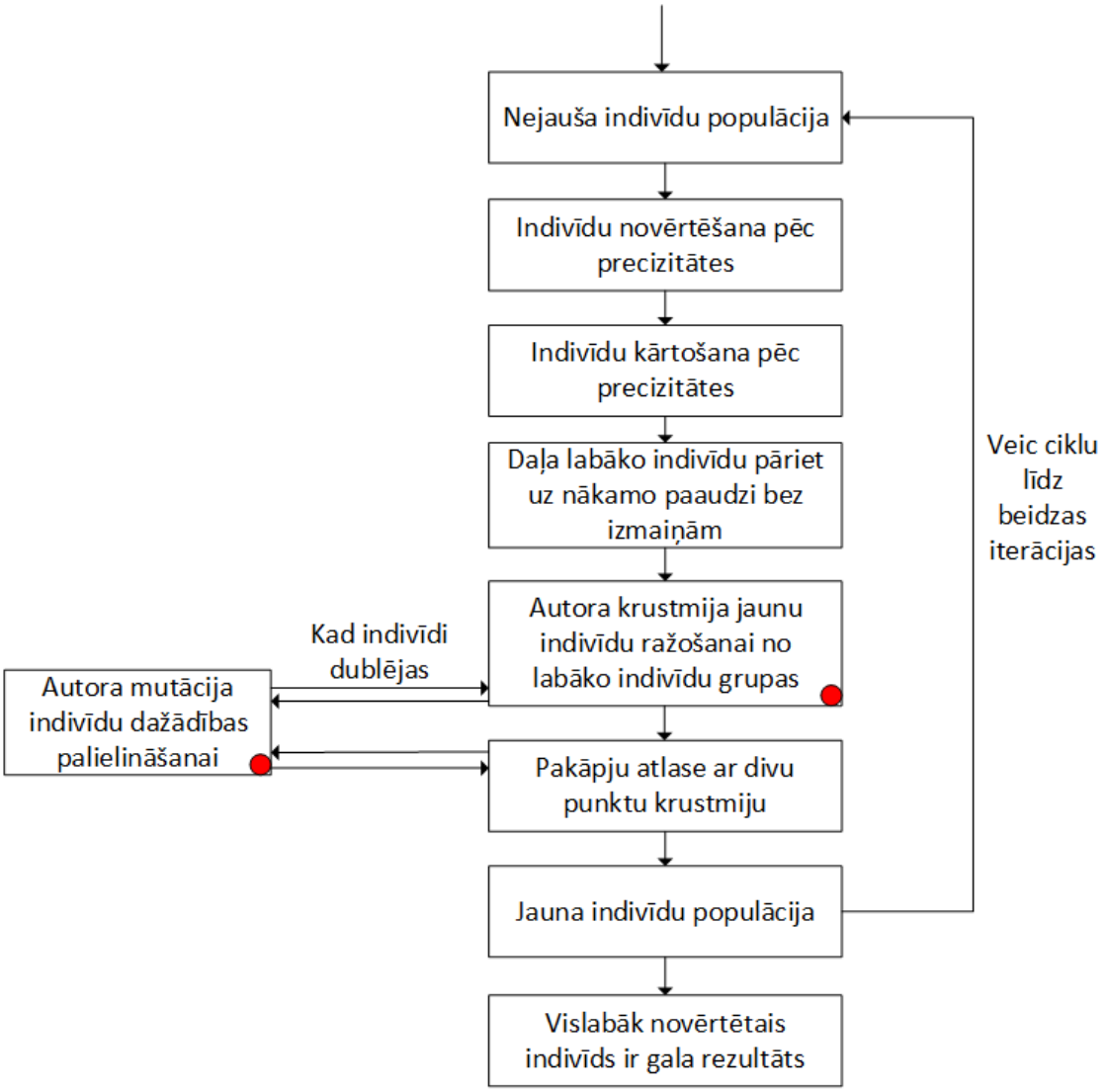
Autors domā, ka izstrādi varētu sākt ar atribūtu selekciju. Tas ir varētu izstrādāt GA, kas veic atribūtu selekciju kādam no WEKA API pieejamajiem klasifikatoriem [8]. Ticams, ka atribūtu selekcija jau sniegtu kādu uzlabojumu, kā arī tas izveidotu pamatu tālākām idejām.

4.2. Izstrādātā ģenētisko algoritmu atribūtu selekcijas risinājuma apraksts

Šeit ir aprakstīts izstrādātais ģenētisko algoritmu atribūtu selekcijas risinājums valodā Java, kurš pielieto WEKA [8]. 1. pielikumā var apskatīt informāciju par autora izstrādātā koda apskatīšanu un izmēģināšanu. Izstrādātā selekcija tiek pielietota klasifikācijas procesā, lai to uzlabotu.

WEKA API tiek izmantots, lai lietotu k tuvāko kaimiņu klasifikatoru: padotu tam parametrus (kaimiņu skaits, pielietojamie atribūti), klasificētu un nolasītu tā rezultātus. WEKA API arī tiek lietots, lai veiktu operācijas ar datiem: to nolasīšana, dalīšana daļās un nejauša jaukšana [8].

Risinājuma kopskats, kur sarkanie punkti apzīmē izstrādātās autora jaunās idejas, ir redzams 4.1. att. **Sākotnējais izstrādātais autora risinājums ar jaunajām autora idejām**. Sīkāks apraksts ir pieejams šajā apakšnodaļā.



4.1. att. **Sākotnējais izstrādātais autora risinājums ar jaunajām autora idejām:**

(sarkanie punkti) jaunās autora idejas

Sākotnējās populācijas ziņā tiek izmantota nejauša inicializācija. Indivīds ir Būla vērtību masīvs. Patiesas vērtības nosaka, kurus atribūtus izmantot klasifikācijas algoritmam. Klases atribūts indivīdos netiek reprezentēts, jo tas vienmēr tiek pievienots atribūtiem, kad tos nosaka klasifikācijas algoritmam. Risinājumā ir iebūvēta arī X pirmo atribūtu ignorēšana gadījumam, kad pirmie atribūti ir objektu ID un tam līdzīgi. GA indivīda piemērs ir redzams 4.2. att. **Autora pielietotās indivīda hromosomas piemērs**. Indivīda objektā ir iekļauta arī tā derīguma vērtība, kas ir tā precizitātes novērtējums, kāds % objektu tika pareizi klasificēts. Pati trenēšana ir sīkāk aprakstīta tālāk šajā apakšnodaļā.

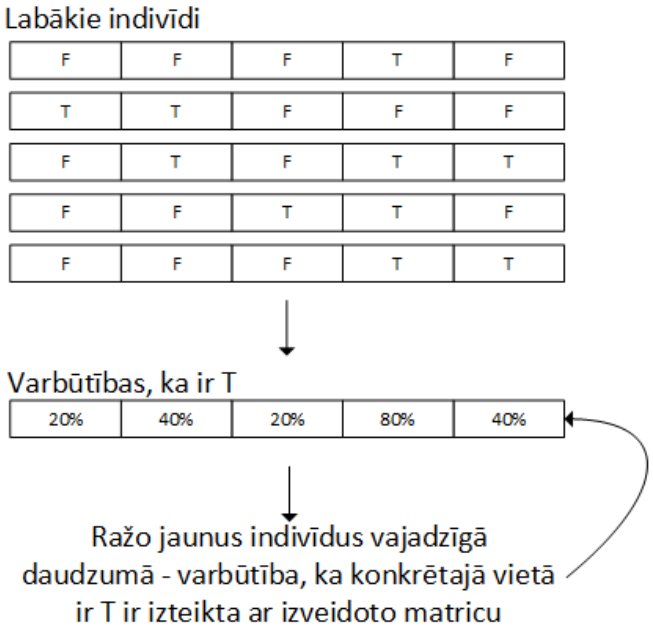
F	F	T	F	T	T	F	T

4.2. att. **Autora pielietotās indivīda hromosomas piemērs:**

(T) atribūtu izmantot, (F) atribūtu neizmantot

Indivīdu selekcijas ziņā tie katrā GA paaudzē tiek sakārtoti pēc to derīguma dilstošā secībā. Gadījumā, ja derīgums ir vienāds, tad augstāku prioritāti dot indivīdiem, kuri izmanto

mazāku atribūtu skaitu. Tas ļauj identificēt labāko indivīdu grupu. Daļa labāko indivīdu vienmēr nonāk nākamajā paaudzē bez izmaiņām tajos (to nosaka krustmijas proporcija). Tad cita daļa labāko indivīdu ir atbildīgi par jaunu indivīdu izveidi. Šiem labākajiem indivīdiem tiek noteiktas to varbūtības. Tādā ziņā, ka varbūtība izsaka, cik % iespējams, ka labākajos indivīdos šajā vietā būs patiess. Tad no šīm varbūtībām uzražo vajadzīgo indivīdu skaitu. Pamata ideja šai autora pieejai ir izveidot krustmiju, kura saražo vairākus indivīdus no mazāka indivīdu skaita tā lai tie būtu līdzīgi labākajiem indivīdiem (tātad no x indivīdiem uztaisa y indivīdus). Pieejas skaidrojums ir redzams 4.3. att. **Autora izveidotā krustmija jaunu indivīdu ražošanai no labāko indivīdu grupas, izmantojot to vērtības un vērtību varbūtības**, kā arī autora izstrādātās krustmijas Java funkciju var apskatīt 2. pielikumā.

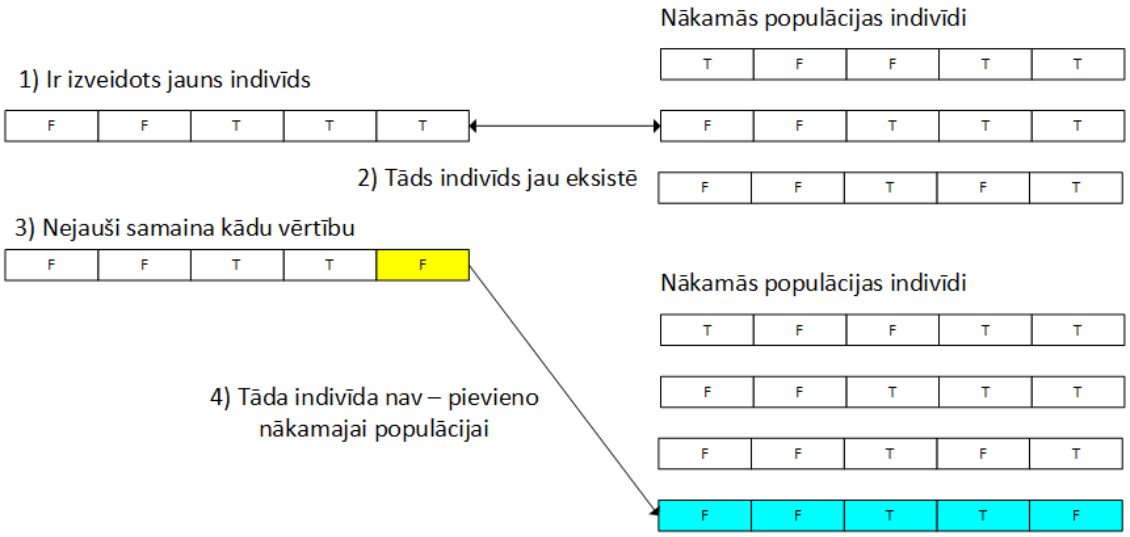


4.3. att. **Autora izveidotā krustmija jaunu indivīdu ražošanai no labāko indivīdu grupas, izmantojot to vērtības un vērtību varbūtības:**
(T) atribūtu izmantot, (F) atribūtu neizmantot

Labākie indivīdi un iepriekšminētā pieeja neuztaisa visu populāciju. Atlikušo daļu uztaisa ar pakāpju selekciju un divu punktu krustmiju.

Mutācijas ziņā autors nolēma, ka mutācijas proporcijas parametru var nepielietot, ja mutāciju sasaista ar indivīdu dažādības uzturēšanu. Tātad mutāciju pielieto, nevis tad, kad ir iestājies mutācijas parametrs, bet tad, kad ir noteikts, ka ir uztaisīti divi identiski indivīdi. Līdz ar to mutāciju, kas nejauši samaina, kādu vienu vērtību hromosomā, indivīdam pielieto tik ilgi, kamēr tas kļūst unikāls. Lai realizētu šo, nākamās paaudzes indivīdu populācija tiek glabāta jaucējtabulā, kurā katrs ievietotais indivīds ir unikāls. Šim nolūkam tiek izmantota Java 8 pieejamā jaucējfunkcija Būla vērtību masīviem. Tātad mutāciju pielieto atkarībā no ieviestās jaucējtabulas satura un Būla vērtību masīvu vērtībām. Pamatojums šādai autora

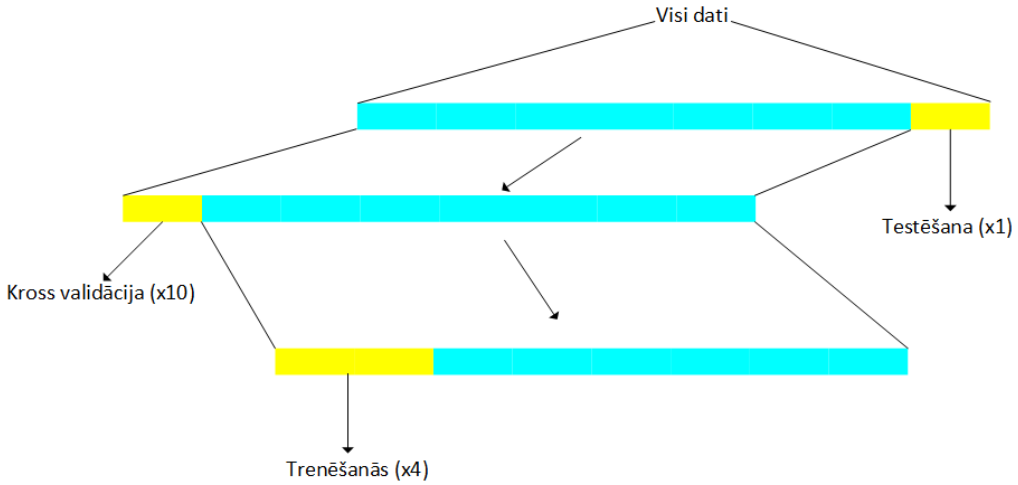
pieejai ir, ka viens indivīds vairākos eksemplāros nedot nekādu papildus informāciju un līdz ar to indivīdu dublēšana ir lieka. Piemēru šai autora pieejai var apskatīt 4.4. att. **Autora izveidotā mutācija indivīdu dažādības palielināšanai**, kā arī izstrādātās mutācijas Java funkciju var apskatīt 3. pielikumā.



4.4. att. **Autora izveidotā mutācija indivīdu dažādības palielināšanai:**

(T) atribūtu izmantot, (F) atribūtu neizmantot

Trenēšanas laikā tiek veikta atkārtota indivīdu precizitātes izvērtēšana. Tiek izmantota ideja no [26], kas ir līdzīga šķērsvalidācijai, tikai trenēšanas laikā. Tātad indivīda galējā precizitāte ir vidējais no 4 trenēšanas reizēm, kur pārbaudei katru reizi izmanto $\frac{1}{4}$ no visiem trenēšanas datiem un attiecīgi trenēšanai pārējās $\frac{3}{4}$. Pamata ideja ir samazināt pārlieku pielāgotību, katru reizi pārbaudot uz neredzētiem datiem [26]. Izveidotā piemēra pārbaudīšanai arī izmanto šķērsvalidāciju. Datu lietojuma kopskats ir redzams 4.5. att. **Risinājuma datu dalīšana un to izmantojums pa testēšanas, šķērsvalidācijas un trenēšanas reizēm**. Tātad vispirms nodala datus testēšanai, tad uz palikušajiem datiem veic šķērsvalidāciju 10 reizes, kur katrā reizē dati tiek atkal dalīti 4 daļās.



4.5. att. **Risinājuma datu dalīšana un to izmantojums pa testēšanas, šķērsvalidācijas un trenēšanas reizēm**

Attiecīgi izveidotais risinājums tika pārbaudīts ar UCI mašīnmācīšanās repozitorija palīdzību [17]. Veiktie eksperimenti ir aprakstīti nākamajā apakšnodaļā.

4.3. Veiktie eksperimenti ar izstrādāto atribūtu selekcijas risinājumu

Pirms eksperimentiem visi dati tika sadalīti trenēšanas un testēšanas datu kopās. Testēšanai tiek izmantoti 20% visu datu. Pie tam dati tiek arī nejauši sajaukti. Vienīgais izņēmums ir pilsētas zemes tipu datu kopa, kurā trenēšanas un testēšanas kopas bija jau iepriekš sadalītas [12, 13, 17]. Šķērsvalidācija tiek veikta uz visiem trenēšanas datiem. Risinājuma svarīgākie parametri, skaidrojumi un to vērtības ir redzamas sekojošajā 4.1. tabulā. Procentu vērtības ir domātas kā procenti no visas populācijas.

4.1. tabula

Autora atribūtu selekcijas risinājuma galvenie parametri, to skaidrojumi un vērtības

Parametrs	Skaidrojums	Vērtība
Populācijas izmērs	Cik daudz indivīdu risinājumu ir	50
Paaudzes	Cik daudzās iterācijās atkārto algoritmu	1000
Labāko indivīdu grupas lielums	Cik liels % labāko indivīdu piedalās jaunu indivīdu veidošanā	5%
Labāko indivīdu grupas indivīdi	Cik liels % indivīdu tiek uzražots no labāko indivīdu grupas	20%
Krustmijas proporcija	Cik liels % indivīdu tiek izmainīti uz nākamo paaudzi – atlikušais % labāko indivīdu pāriet uz nākamo paaudzi bez izmaiņām	95%
Kaimiņu skaits	WEKA KNN algoritmam [8], cik daudzi kaimiņi tiek izmantoti, lai veiktu klasificēšanu	Pēc datu kopas

Izstrādātais risinājums veic atribūtu selekciju WEKA KNN klasifikācijas algoritmam [8]. Risinājums ar autora izstrādāto atribūtu selekciju tiek salīdzināts pret WEKA versiju bez

atribūtu selekcijas. Datu kopas tiek ņemtas no UCI mašīnmācīšanās repozitorija – īsi to apraksti ir pieejami sekojošajā 4.2. tabulā [17].

4.2. tabula

Eksperimentos izmantotās datu kopas, to atribūti, instanču skaits un apraksti

UCI datu kopa (atribūti-instances)	**Īss apraksts**
Jonosfēra (35-351) [17]	Binārs klasifikācijas uzdevums (labi/slikti mērījumi) – radara dati, kuri pēta brīvos elektronus jonosfērā [17]
Viskonsinas krūts vēža prognozēšana (34-198) [17]	Binārs klasifikācijas uzdevums (vēzis ir atkārtots vai nav pēc ārstēšanas) – atribūti ir no iegūtajiem pacientu attēliem [17]
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	Binārs klasifikācijas uzdevums (vēzis ir labdabīgs vai ļaundabīgs) – atribūti ir no iegūtajiem pacientu attēliem [17]
Pilsētas zemes tipi (148-675) [12, 13, 17]	9 klašu klasifikācijas uzdevumus (dažādi zemes tipi) – atribūti ir no aero attēliem, maz piemēru katrai klasei (14-30) [12, 13, 17]
Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	Binārs klasifikācijas uzdevums (satur vai nesatur diabētiskas retinopātijas pazīmes) – atribūti ir no attēliem [4, 7, 17]
Aritmija (280-452) [17]	16 klašu klasifikācijas uzdevumus (ir vai nav sirds aritmijas, kā arī 15 sirds aritmijas grupas) [17]
Muskuss (versija 1) (169-476) [17]	Binārs klasifikācijas uzdevums (ir vai nav muskuss) – atribūti raksturo molekulu uzbūvi [17]

Veikto eksperimentu rezultāti ir redzami sekojošajā 4.3. tabulā. Atribūtu daudzuma novērtējumi iekļauj klases atribūtu. Instanču daudzums iekļauj gan trenēšanas, gan testēšanas kopas. Katrai datu kopai tiek norādīts arī pielietotais tuvāko kaimiņu skaits, kas ir viens no WEKA KNN klasifikācijas algoritma parametriem [8]. Tā vērtību nosacīja autors pēc validācijas rezultātu analīzes. Dažviet atribūti tika izņemti pirms eksperimentu veikšanas, gadījumā ja tie bija noteikti lieki (kā objekta ID, nosaukums).

4.3. tabula

Veiktie eksperimenti ar autora izstrādāto atribūtu selekciju, iegūtās precizitātes un izmantotie atribūti

UCI datu kopa (atribūti-instances)	Kaimiņi	Rezultāti – precizitātes un atribūti			
		Mērījums	WEKA k tuvāko kaimiņu rezultāti	Autora + WEKA k tuvāko kaimiņu rezultāti	Iegūtais autora uzlabojums
Jonosfēra (35-351) [17]	3	**Validācija**	85,7512%	89,6798%	+3,9286%
		Testēšana	88,5714%	90%	+1,4286%
		Atribūti	35	10	71,4286%
Viskonsinas krūts vēža prognozēšana (34-198) [17]	5	**Validācija**	78,058%	75,6232%	-2,4348%
		Testēšana	62,5%	67,5%	+5%
		Atribūti	32	14	56,25%
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	3	**Validācija**	96,9556%	95,4444%	-1,5112%
		Testēšana	94,7368%	93,8596%	-0,8772%
		Atribūti	31	15	51,6129%
Pilsētas zemes tipi (148-675) [12, 13, 17]	3	**Validācija**	77,0833%	79,5833%	+2,5%
		Testēšana	24,6548%	26,43%	+1,7752%
		Atribūti	148	60	59,4595%
Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	31	**Validācija**	64,3829%	66,5603%	+2,1774%
		Testēšana	62,1739%	63,0435%	+0,8696%
		Atribūti	20	6	70%
Aritmija (280-452) [17]	3	**Validācija**	57,7632%	61,038%	+3,2748%
		Testēšana	57,7778%	57,7778%	0%
		Atribūti	280	72	74,2857%
Muskuss (versija 1) (169-476) [17]	1	**Validācija**	84,7571%	87,1188%	+2,3617%
		Testēšana	84,2105%	85,2632%	+1,0527%
		Atribūti	167	67	59,8802%

Autora kursa darbā testēšanas rezultāti tika iegūti, veicot trenēšanu ar to šķērsvalidācijas datu kopu, kura ieguva vislielāko precizitāti validācijas laikā, nevis ar visiem trenēšanas datiem [2]. Autors maģistra darbā veica papildus eksperimentus, kuros testēšana tika veikta vēlreiz ar tiem pašiem parametriem, bet trenēšanai tika izmantoti visi trenēšanas dati. Rezultāti ir redzami 4.4. tabulā, kur salīdzināšanai vecie rezultāti ir ielikti iekavās.

4.4. tabula

Veiktie eksperimenti ar autora izstrādāto atribūtu selekciju, trenējot ar visu trenēšanas kopu testēšanai, iegūtās precizitātes, izmantotie atribūti un vecās vērtības

UCI datu kopa (atribūti-instances)	Kaimiņi	Rezultāti – precizitātes un atribūti, vecās vērtības ir ievietotas iekavās			
		Mērījums	WEKA k tuvāko kaimiņu rezultāti	Autora + WEKA k tuvāko kaimiņu rezultāti	Iegūtais autora uzlabojums
Jonosfēra (35-351) [17]	3	**Testēšana**	90% (88,5714%)	94,2857% (90%)	+4,2857% (+1,4286%)
		Atribūti	35 (35)	8 (10)	77,1429% (71,4286%)
Viskonsinas krūts vēža prognozēšana (34-198) [17]	5	**Testēšana**	55% (62,5%)	65% (67,5%)	+10% (+5%)
		Atribūti	32 (32)	16 (14)	50% (56,25%)
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	3	**Testēšana**	94,7368% (94,7368%)	93,8596% (93,8596%)	-0,8772% (-0,8772%)
		Atribūti	31 (31)	15 (15)	51,6129% (51,6129%)
Pilsētas zemes tipi (148-675) [12, 13, 17]	3	**Testēšana**	70,0197% (24,6548%)	71,2032% (26,43%)	+1,1835% (+1,7752%)
		Atribūti	148 (148)	53 (60)	64,1892% (59,4595%)
Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	31	**Testēšana**	62,1739% (62,1739%)	64,3478% (63,0435%)	+2,1739% (+0,8696%)
		Atribūti	20 (20)	6 (6)	70% (70%)
Aritmija (280-452) [17]	3	**Testēšana**	60% (57,7778%)	58.89% (57,7778%)	-1,11% (0%)
		Atribūti	280 (280)	82 (72)	70,7143% (74,2857%)
Muskuss (versija 1) (169-476) [17]	1	**Testēšana**	85,2632% (84,2105%)	83,1579% (85,2632%)	-2,1053% (+1,0527%)
		Atribūti	167 (167)	55 (67)	67,0659% (59,8802%)

Pilsētas zemes tipu kopas [12, 13, 17] ziņā testēšanas rezultāts ļoti palielinājās. Problēma bija tajā, ka testēšanas kopas datnē klases nebija uzrādītas identiskā secībā, kā trenēšanas kopas datnē, tāpēc iepriekš testēšanas rezultāts bija tik mazs. Veicot trenēšanu ar labāko šķērsvalidācijas kopu, WEKA risinājums iegūst 69,428% testēšanas precizitāti [8], bet

autora risinājums iegūst 70,0197% testēšanas precizitāti uz Pilsētas zemes tipu kopas [12, 13, 17], tāpēc arī šajā gadījumā trenēšana ar visu trenēšanas datu kopu dot precizitātes uzlabojumu.

Izvēlētā iepriekšējā trenēšanas pieeja ar labāko šķērsvalidācijas kopu ietaupa laiku (nav jāveic trenēšana ar visu trenēšanas kopu) un pēc rezultātiem redzams, ka ne visur tiek iegūti uzlabojumi, tomēr kopumā, veicot trenēšanu ar visu trenēšanas kopu, testēšanas rezultāti ir palikuši labāki. Testēšanas precizitātes palika sliktākas tikai Viskonsinas krūts vēža prognozēšanas datu kopai [17] (abiem risinājumiem) un Muskusa (versijas 1) datu kopai [17] (autora risinājumam). Testēšanas precizitātes uzlabojās pavisam 8 gadījumos, pasliktinājās 3 gadījumos un neizmainījās 3 gadījumos. Līdz ar to autors tālākos testos testēšanas rezultātus iegūst, trenējot ar visu trenēšanas kopu.

Kopumā redzams, ka izstrādātais risinājums vienmēr spēj samazināt izmantojamo atribūtu skaitu un ievērojamā daļā gadījumu arī spēj iegūt lielāku precizitāti par dažiem %.

Atribūtu skaita samazinājums ir ievērojams – visos gadījumos tika izmesti vairāk par 50% atribūtu (trenējoties ar visiem trenēšanas datiem, mazākais rādītājs ir tieši 50%). Pie tam trijos gadījumos tika izmesti 70% atribūtu vai vairāk. Pat ja nav precizitātes uzlabojuma, tas dot iespēju strādāt ar mazāka apjoma datiem.

Precizitātes ziņā attiecīgie uzlabojumi vai zudumi ir dažos procentos (5% uzlabojums ir vislielākais sākotnējā variantā, bet 10% uzlabojums tika sasniegts, trenējoties ar visiem trenēšanas datiem). Sākotnējā variantā četrās datu kopās uzlabojumi tika sasniegti gan validācijas, gan testēšanas rezultātos: Jonosfēras [17], Pilsētas zemes tipu [12, 13, 17], Debrecenas diabētiskas retinopātijas [4, 7, 17], un Muskusa (versijas 1) [17] (trenējoties ar visiem trenēšanas datiem, nebija testēšanas uzlabojuma Muskusa (versijas 1) datu kopā [17]). Pārējās trīs datu kopās nebija uzlabojumu abos mērījumos vai attiecīgi bija zudumi.

4.4. Atribūtu selekcijas risinājuma rezultātu salīdzinājums ar citu līdzīgu pētījumu rezultātiem

Šeit ir parādīti līdzīgi atribūtu selekcijas rezultāti no citiem pētījumiem un dots autora novērtējums. Sekojošajā 4.5. tabulā ir pieejams rezultātu salīdzinājuma kopsavilkums par izmesto atribūtu procentiem.

4.5. tabula

Rezultātu salīdzinājuma kopsavilkums par autora un apskatīto pētījumu izmesto atribūtu procentiem

Pētījums	Izmesto atribūtu procents
Autora pētījums	~50%-~70%
Krūts vēža noteikšana ar K tuvāko kaimiņu algoritmu, optimizējot izvēlētos atribūtus un parametrus [21]	65,625% [21]
Uzņēmumu bankrota klasificēšana ar klasifikatoru ansambli divos soļos [26]	38,1818% [26]
Elektrokardiogrammu signālu klasificēšana ar neironu tīkliem, veicot dimensiju redukciju [16]	~50% [16]

Atribūtu selekcijas eksperimentus ar Viskonsinas krūts vēža prognozēšanas datu kopu [17] veica pētījums [21]. Šajā pētījumā testa kopas nebija, bet tika veikta 10 reižu šķērsvalidācija [21]. Labākais vidējais rezultāts bija 77.57% precizitāte [21]. Tika iegūts gandrīz 2% uzlabojums pār lietoto k tuvāko kaimiņu algoritmu [21]. Pētījumā tika atrasta 11 atribūtu kopa (izmesti 65,625% atribūtu) [21]. Attiecīgi autors uz Viskonsinas krūts vēža prognozēšanas datu kopas [17] ieguva 75,6232% vidējo precizitāti (-2,4348% zudums, salīdzinot ar WEKA versiju [8]) un atrada 14 atribūtu kopu (izmesti 56,25% atribūtu), bet te jāmin, ka uz testēšanas kopas tika iegūts 5% uzlabojums (10% uzlabojums, trenējoties ar visu trenēšanas kopu – 16 atribūti, izmesti 50% atribūtu).

Pētījumā par bankrota klasificēšanu arī tika veikta atribūtu selekcija k tuvāko kaimiņu algoritmam [26]. Tika iegūts 4,1% uzlabojums vidējajā precizitātē [26], kā arī tika izmesti 38,1818% atribūtu [26].

Pētījumā par elektrokardiogrammu signālu klasificēšanu tika veikta atribūtu selekcija neironu tīklam [16]. Lai gan izstrādātā selekcija atsevišķi netika testēta, tā spēja izmest ap 50% atribūtu [16].

Kopumā var teikt, ka rezultāti ir saskanīgi. Ar GA atribūtu selekciju var atmest lielu daļu atribūtu, kā arī iegūt dažu % precizitātes uzlabojumu. Autora risinājums ne vienmēr varēja iegūt labāku precizitāti, bet tajos gadījumos, kad uzlabojums tika iegūts, tas arī bija par dažiem % (5% uzlabojums bija vislielākais ar labāko validācijas kopu, 10% uzlabojums bija vislielākais ar visu trenēšanas kopu). Autora izmesto atribūtu procenti bija aptuvenā 50%-70% intervālā, tātad dažviet tika izmests diezgan liels atribūtu daudzums.

5. IZSTRĀDĀTAIS ĢENĒTISKO ALGORITMU KLASIFIKĀCIJAS RISINĀJUMS

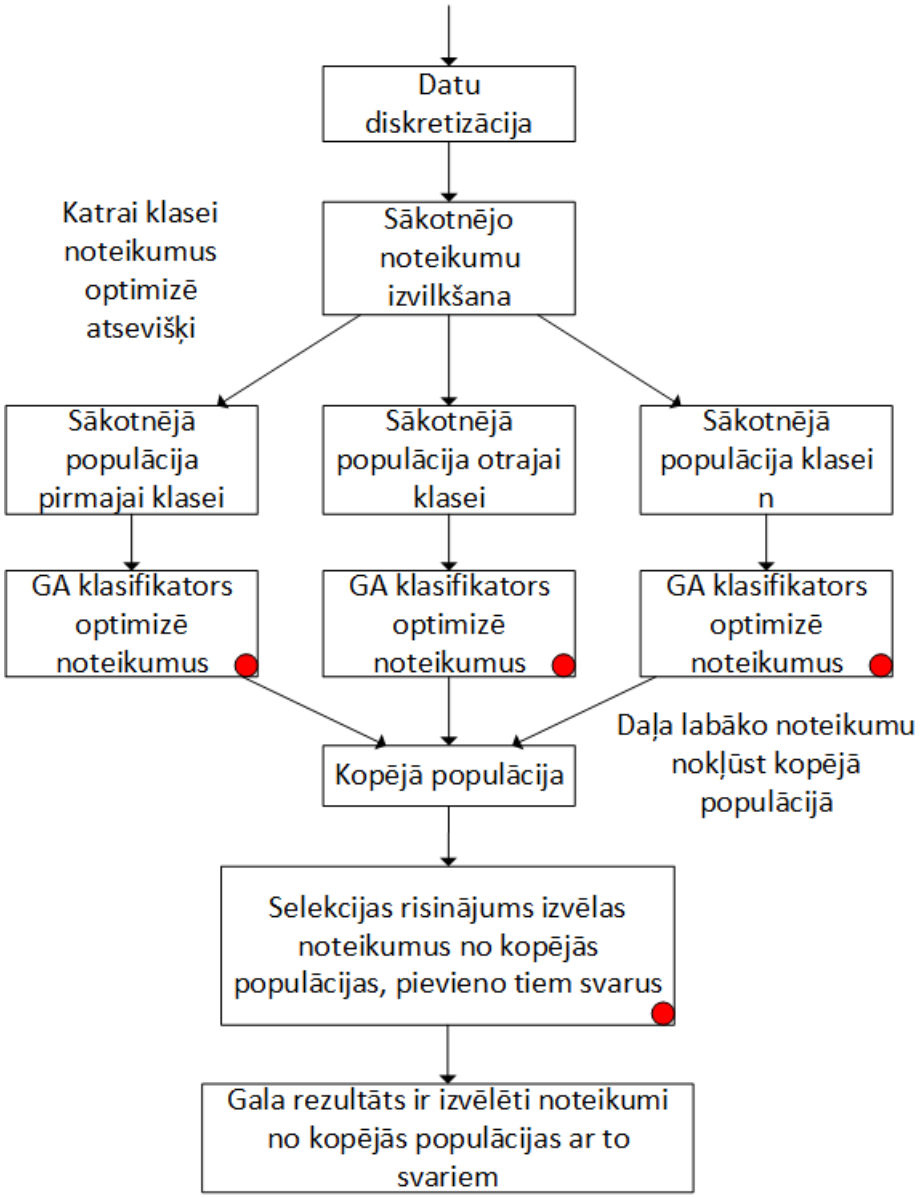
Šajā nodaļā ir aprakstīts izstrādātais GA klasifikācijas risinājums. Tas pielieto 4. nodaļā aprakstīto atribūtu selekcijas risinājumu klasifikācijas noteikumu izvēlei. Nodaļā ir aprakstīts paveiktais, veiktie eksperimenti un analīze.

5.1. Izstrādātā ģenētisko algoritmu klasifikācijas risinājuma apraksts

Tāpat, kā izstrādātais atribūtu selekcijas risinājums, GA klasifikācijas risinājums ir izstrādāts valodā Java un pielieto WEKA [8]. 1. pielikumā var atrast informāciju par autora izstrādātā koda apskatīšanu un izmēģināšanu. No WEKA API papildus tiek lietots J48 klasifikators, kā arī pieejamās operācijas ar datiem, piemēram, datu diskretizācija [8].

Klasifikācija tiek veikta ar noteikumu palīdzību. Klasifikācijas noteikumi ir līdzīgi ja-tad nosacījumiem. Tas ir, ja ieraksta atribūtiem ir šādas vērtības, tad tas pieder šādai klasei. Noteikumi kopumā veido ansambli, kurā ir iespējams veikt balsošanu. Izstrādātā atribūtu selekcija tiek pielietota, lai izvēlētos noteikumus, nevis atribūtus. Noteikumi paši var netieši noteikt, kurus atribūtus nelietot, vienkārši tos neiekļaujot.

Vispārīgu autora izstrādātā GA klasifikācijas risinājuma diagrammu var redzēt 5.1. att. **Autora izstrādātais klasifikācijas risinājums ar ieviestajām autora jaunajām idejām**. Vietas, kurās ir izmantotas autora jaunās idejas, ir apzīmētas ar sarkaniem punktiem. Noteikumu optimizēšanai autors ir izstrādājis savu indivīdu derīguma funkciju, kā arī ir izstrādāta autora pieeja saražoto klasifikācijas noteikumu sakombinēšanai.

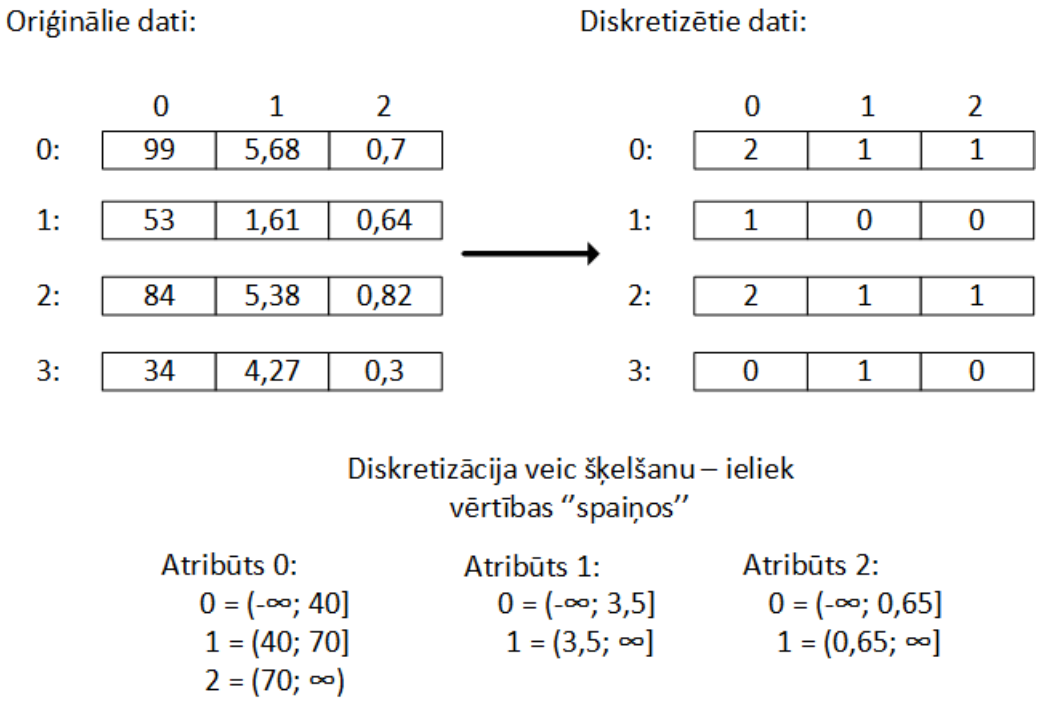


5.1. att. **Autora izstrādātais klasifikācijas risinājums ar ieviestajām autora jaunajām idejām:** (sarkanie punkti) jaunās autora idejas

Izstrādātajā risinājumā dati tiek apstrādāti pirms to lietošanas. Tas ir tie tiek diskretizēti. Piemērs datu diskretizācijas rezultātam ir redzams 5.2. att. **Piemērs iespējamam datu diskretizācijas rezultātam**. Pārsvarā tiek izmantota WEKA API pieejamā uzraudzītā datu diskretizācija [8]. Viskonsinas krūts vēža prognozēšanas datu kopai [17] tiek izmantota WEKA API neuzraudzītā datu diskretizācija [8], jo uzraudzītais diskretizācijas variants šajā gadījumā visus izmantotos atribūtus diskretizē uz vienu vērtību. Līdz ar to tie kļūst neizmantojami.

Ja datos, piemēram, atribūts ir ar reālo skaitļu vērtībām, tad iespējams ir bezgalīgs vērtību skaits šim atribūtam. Diskretizācija nodrošina, ka šīs vērtības tiek pārnestas uz galīgu vērtību telpu. Dati, kuri jau sākotnēji ir diskrēti, protams, netiek diskretizēti. Līdz ar to klasifikācijas noteikumi, lai pārbaudītu ieraksta atbilstību sev, novērtē, vai konkrētā vērtība

datos ir vienāda ar vērtību noteikumā (tieši vienāda, nevis, piemēram, mazāka, vai lielāka). Tātad šajā gadījumā diskretizācija atvieglo GA operācijas ar datiem, jo nav jāveic lēmumi, kurās vietās šķelt datu telpu (tas praktiski ir jau izdarīts iepriekš).

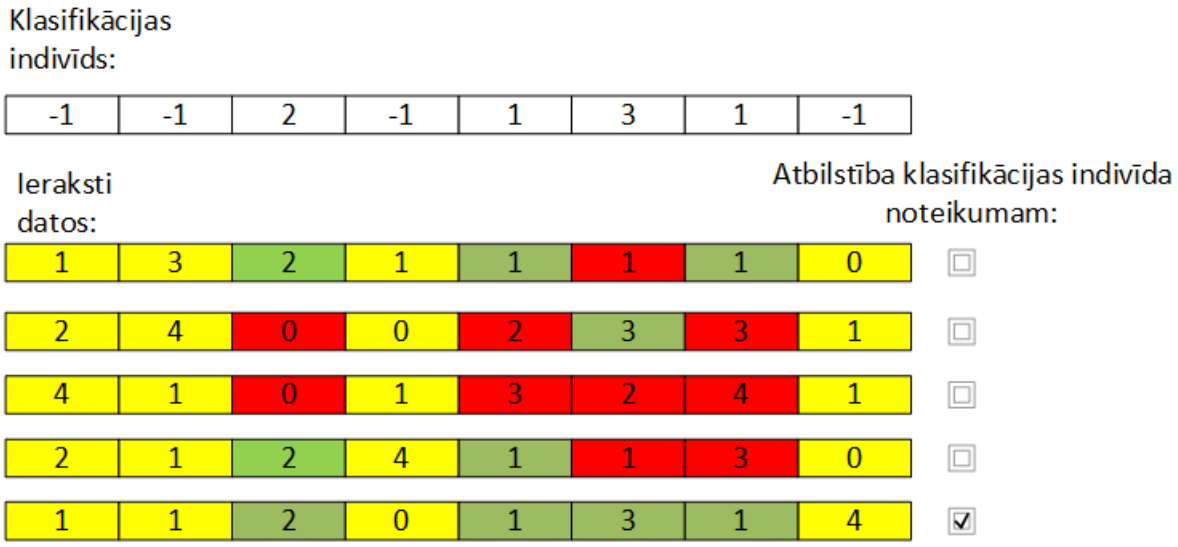


5.2. att. **Piemērs iespējamam datu diskretizācijas rezultātam**

Šajā gadījumā GA klasifikācijas indivīda hromosoma ir naturālu skaitļu virkne, kura nosaka iespējamu klasifikācijas noteikumu. Piemērs GA klasifikācijas indivīda hromosomai ir redzams 5.3. att. **Autora izmantotā klasifikācijas indivīda hromosomas piemērs ar tās pielietojumu klasificēšanas procesā**. Katrs skaitlis atbilst vienam atribūtam un norāda uz jau sākotnēji diskrētu vērtību vai kādu "spaini" jeb skaitļu intervālu, piemēram, (40; 70], pret kuru vērtības tiek pārbaudītas. Attiecīgi 5.2. att. **Piemērs iespējamam datu diskretizācijas rezultātam** atribūta 0 iespējamās vērtības, kuras var parādīties GA klasifikācijas indivīdā, ir: {0,1,2}. Indivīdā ir iespējama arī vērtība "-1", kura norāda uz to, ka atribūts netiek pārbaudīts attiecīgajā noteikumā. Katram noteikumam ir arī klase, kura neietilpst hromosomā, jo to GA operācijas nemaina. Tātad, ja ieraksts datos pilnībā atbilst indivīda hromosomā atrodamajam, izņemot vietas ar "-1" vērtībām, tad attiecīgais indivīda klasifikācijas noteikums ir izpildījies, nosakot, ka ieraksta klase ir klase, kura ir pielikta konkrētajam indivīdam.

Autors ir izvēlējies, ka indivīda klase netiek iekļauta hromosomā. Šajā ziņā ir jānorāda, ka datos klases atribūts pretēji tiek iekļauts. Šāda izvēle tika izdarīta, lai GA operācijas būtu vienkāršāk izpildīt. Attiecīgi izstrādātajā risinājumā klases atribūts datos tiek izlaists, kad tas ir nepieciešams. Maģistra darba piemēros klases atribūts datos neeksistē vienkāršības dēļ, bet

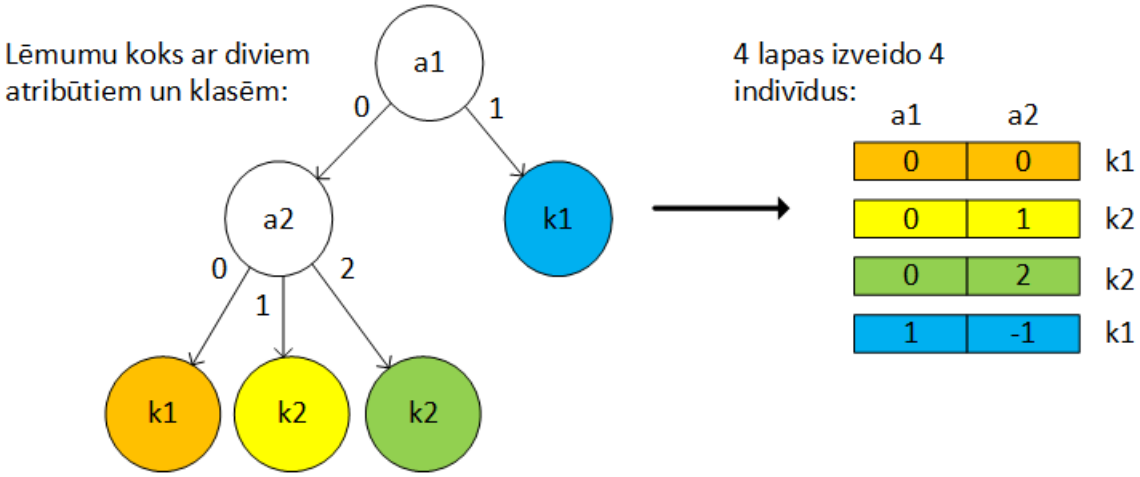
reāli vērtības vietām būtu par vienu pozīciju nobīdītas, piemēram, otrā pozīcija indivīdā atbilstu trešajai pozīcijai datos un tā tālāk.



5.3. att. **Autora izmantotā klasifikācijas indivīda hromosomas piemērs ar tās pielietojumu klasificēšanas procesā:**

(-1) pārbaudes nav, citādi pārbauda atbilstību

Līdzīgi kā [18], sākotnējā populācija tiek inicializēta ar C4.5 izveidotā lēmumu koka palīdzību [18]. Sākotnējie indivīdi, noteikumi tiek iegūti no WEKA API J48 klasifikatora [8]. Tas tiek palaists uz pieejamajiem trenēšanās datiem un uztaisa savu lēmumu koku, kurš tiek pārveidots par sākotnējiem indivīdiem [8]. Izveidotais indivīdu skaits ir identisks lapu skaitam. Attiecīgi katrs indivīds attēlo vienu iespējamo ceļu līdz kādai koka lapai [18]. Indivīdu veidošana no lēmumu koka ir redzama 5.4. att. **Sākotnējo indivīdu iegūšana no lēmumu koka**. Tā kā dati tika jau iepriekš diskretizēti arī J48 izveidotais lēmumu koks šķautnēs satur tikai salīdzināšanas, kuras pārbauda vienādību ar kādu vērtību [8].



5.4. att. **Sākotnējo indivīdu iegūšana no lēmumu koka:**

(a1) atribūts ar divām iespējamām vērtībām, (a2) atribūts ar trim iespējamām vērtībām, (k) noteiktās klases

Sākotnējā populācija tiek dalīta tik populācijās, cik datos ir klašu. Katra populācija satur indivīdus, kuri reprezentē noteikumus vienai un tai pašai klasei. Attiecīgi sākotnēji iegūtie indivīdi tiek izdalīti pa to atbilstošajām populācijām. Atlikušās brīvās vietas populācijās tiek aizpildītas ar nejaušiem indivīdiem. Šajos indivīdos tiek izvēlēts viens nejaušs atribūts, kuru izmantot, jo noteikumi pārsvarā lielāko daļu atribūtu neizmanto.

Dalīšana populācijās pa to klasēm tiek veikta divu iemeslu dēļ. Pirmkārt, divu dažādu klašu indivīdu jaukšana kopā visdrīzāk nekādu labumu nedos. Pie tam tad arī nav skaidrs, kādai klasei vajadzētu piederēt uzražotajiem indivīdiem. Otrkārt, piemēram, ja viena no klasēm k1 labi atdalās no pārējām – ir viegli nosakāma, bet pārējās klases tik labi neatdalās, tad ir iespējams, ka indivīdi, kuri atbilst klasei k1 izkonkurēs pārējos indivīdus, jo to derīgums būs ievērojami lielāks par citu klašu indivīdiem. Attiecīgi, ja klasei nav noteikumu, tad nav iespējams korekti klasificēt ierakstus no šīs klases.

Indivīdu derīguma ziņā autors izveidoja savu indivīdu derīguma funkciju, kura ir redzama (5.1). Izstrādāto Java funkciju, kura nosaka derīgumu klasifikācijas indivīdiem var apskatīt 4. pielikumā. Pamata motivācija šādai funkcijai ir, ka, meklējot labākus indivīdus, daļa datu varētu tikt izlaista, ja derīguma funkcija uzrāda, ka šos datus ir "neizdevīgi" klasificēt. Tas ir varētu rasties situācija, ka risinājums nezina, kā klasificēt datus, jo attiecīgie noteikumi nav izveidoti. Izstrādātā funkcija galvenokārt dot lielāku derīgumu tiem indivīdiem, kuri pareizi klasificē datus, kurus neklasificē citi indivīdi. Attiecīgi izstrādātā funkcija dot mazāku derīgumu tiem indivīdiem, kuri pareizi klasificē datus, kurus pareizi klasificē arī citi indivīdi.

$$F(ID) = \begin{cases} -\infty, & ID_{PP} \leq 0 \\ -\infty, & (ID_{PP} + ID_{PN}) \leq 1 \\ -\infty, & (ID_{NN} + ID_{NP}) \leq 1 \\ \left(\sum_{i\ \varepsilon\ ID_{PP}=1}^{n} 1 - \frac{i_p}{Ps}\right) * \frac{ID_{PP}}{ID_{PP}+ID_{PN}}, & citādi \end{cases}, \qquad (5.1)$$

F(ID) – indivīda derīguma funkcija,

ID_{PP} - ierakstu skaits, kurus indivīds klasificēja, kā atbilstošus sev un kuri bija atbilstoši,

ID_{PN} - ierakstu skaits, kurus indivīds klasificēja, kā atbilstošus sev, bet kuri bija neatbilstoši,

ID_{NN} – ierakstu skaits, kurus indivīds klasificēja, kā neatbilstošus sev un kuri bija neatbilstoši,

ID_{NP} – ierakstu skaits, kurus indivīds klasificēja, kā neatbilstošus sev, bet kuri bija atbilstoši,

n – kopējais ierakstu skaits datos,

i_p – cik reizes visa indivīdu populācija ir klasificējusi ierakstu i, kā atbilstošu sev un šis spriedums ir bijis korekts,

Ps – visas indivīdu populācijas izmērs

Pirmā rinda funkcijā (5.1) saka, ka visnederīgākie ir tie indivīdi, noteikumi, kuriem nekas neatbilst. Lai noteikums dotu vismaz minimālu pienesumu precizitātes uzlabošanā vismaz viens ieraksts ir jāklasificē, kā atbilstošs sev un pareizi. Otrā un trešā rinda ir samērā līdzīgas. Otrā rinda neļauj radīt situāciju, ka noteikumam atbilst pārāk maz ierakstu. Šajā ziņā pareizībai nav nozīmes – galvenais, lai indivīds, noteikums teiktu, ka tam kaut kas atbilst. Trešā rinda ir pretējais, kas saka, ka visnederīgākie ir indivīdi, kuriem viss atbilst.

Visbeidzot, ja indivīds netiek atzīts par visnederīgāko, tad tiek rēķināts tā derīgums. Derīgums tiek palielināts par katru pareizi klasificēto ierakstu, kurš atbilda noteikumam (summa). Šis palielinājums tiek modificēts atkarībā no tā, cik citiem indivīdiem ieraksts ir korekti bijis atbilstīgs. Ja ir maz šādu citu indivīdu, tad palielinājums ir lielāks un pretēji.

Viss līdz šim dot lielāku derīgumu par pareiziem balsojumiem (lielāks derīgums par pareizi klasificētiem ierakstiem, kuri atbilst noteikumam), bet nav sodu par nepareiziem balsojumiem (mazāks derīgums par nepareizi klasificētiem ierakstiem, kuri tika uzskatīti par atbilstošiem). Līdz ar to dotais derīgums vēl tiek sareizināts ar korekto daļu no visiem balsojumiem (atbilstībām noteikumam). Redzams, ka autors indivīdu derīgumu pamatā balsta uz to balsojumiem, jo attiecīgi indivīdi klasificēšanu veic balsojot.

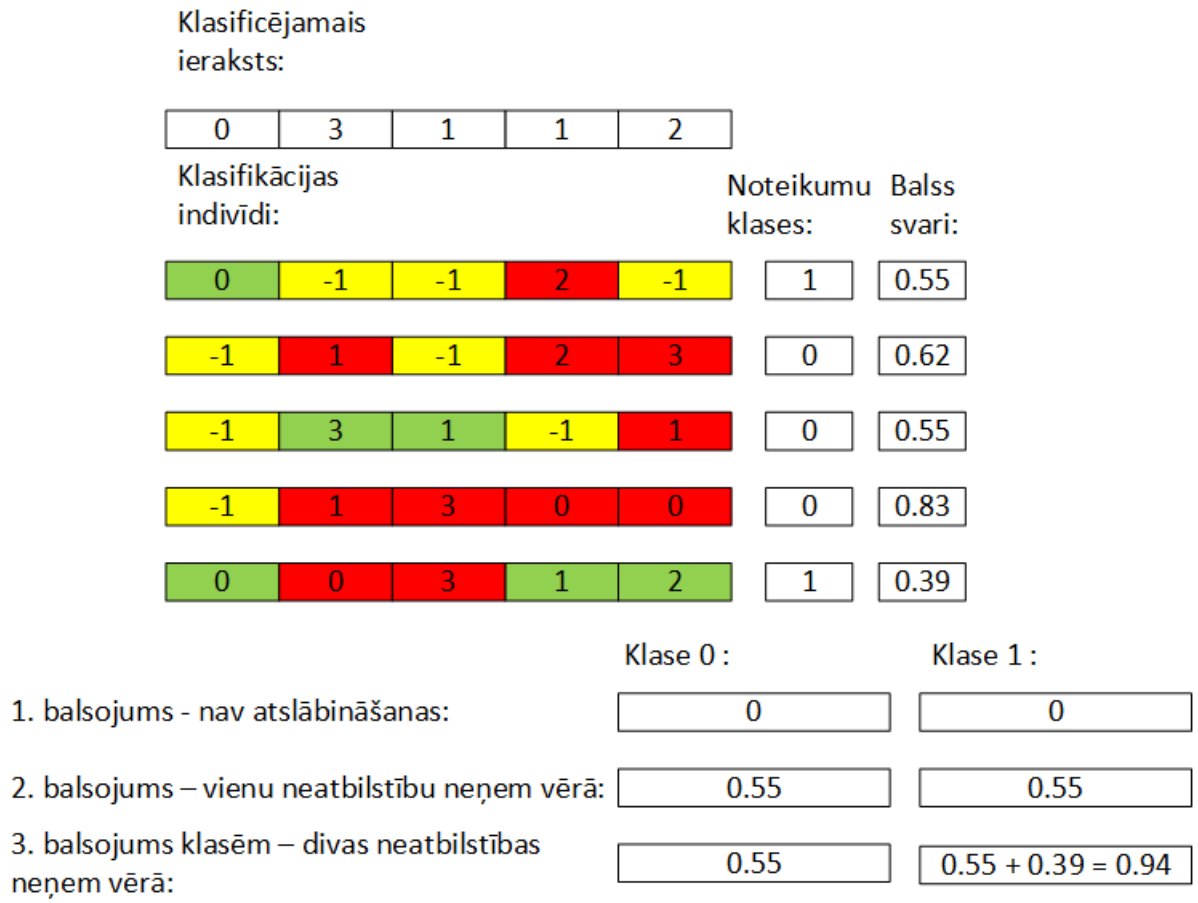
Autora GA klasifikācijas risinājuma operatori ir līdzīgi autora GA selekcijas risinājuma operatoriem. GA selekcijas risinājuma operatoriem ir veikta pielāgošana, lai tie strādātu ar klasifikācijas indivīdiem, bet pamata koncepti nav mainīti. Lielā mērā operatoriem ir pielikta papildus dimensija. Piemēram, mutācija izvēlas gan nejaušu vietu hromosomā, gan arī nejaušu vērtību uz kuru veikt pārmainīšanu (50%, ka tiek izvēlēts "-1", ka atribūtu neizmanto,

un 50%, ka tiek nejauši izvēlēta cita iespējama vērtība), jo izveidotie diskretizētie intervāli var būt vairāki. Jaucējfunkcijas ziņā tiek pielietota Java 8 pieejamā funkcija naturālo skaitļu masīviem. Autora jaunu indivīdu ražošana no labāko indivīdu grupas saglabā un ģenerēšanā izmanto varbūtības pa izveidotajiem diskretizētajiem intervāliem.

Pēc GA veiktās noteikumu optimizēšanas tiek veidota kopējā populācija. Ne visi uzražotie indivīdi nonāk šajā populācijā. Galvenokārt tas tā tiek darīts, lai risinājumam būtu augstāka precizitāte bez noteikumu selekcijas. Citādāk indivīdi ar mazāku derīgumu var traucēt gala rezultātam. Līdz ar to kopējā populācijā ievietojamo indivīdu skaitu no vienas apakš populācijas (populācijas, kura klasificē konkrētu klasi) regulē ar ieviestu parametra vērtību, kura izsaka, cik % indivīdu saglabāt kopējā populācijā. Tas tā tiek darīts, lai kopējā populācijā pārsvarā nonāktu indivīdi, kuri ir vērtīgi un nav mainīti pēdējās paaudzēs. Sākotnējā testēšana parādīja, ka, iekļaujot pilnīgi visus indivīdus, atsevišķos gadījumos bez likumu selekcijas rezultāti bija par dažiem desmitiem procentu sliktāki. Redzams, ka šādai indivīdu neiekļaušanai ir diezgan liela nozīme.

Tālāk tiek izšķirti pirmie divi iespējamie struktūras varianti. Pirmais vienkāršākais variants paredz, ka tiek izmantoti visi indivīdi no kopējās populācijas. Otrais variants paredz, ka tiek veikta noteikumu selekcija ar izstrādāto autora GA selekcijas risinājumu.

Abiem struktūras variantiem ir dažas kopīgas iezīmes. Pirmām kārtām abos gadījumos notiek svērta balsošana. Situācijā, kad balsošanas rezultāts ir neizšķirts, notiek noteikumu vienreizēja atslābināšana un atkārtota balsošana. Ja arī tad ir neizšķirts, tad notiek noteikumu otrreizēja atslābināšana un balsošana. Šis process tiek turpināts līdz vai nu tiek izlemts par ko balsot, vai arī tālāka atslābināšana nav iespējama (jau ir veikta atslābināšana pieejamo atribūtu skaitā). Piemērs pielietotajai balsošanai ar atslābināšanu ir redzams 5.5. att. **Piemērs balsošanai ar atslābināšanu, kura notiek līdz tiek sasniegts rezultāts**. Gadījumā, ja balsošana arī pēc atslābināšanas ir neizšķirta, tad klasifikators klasi izlemj nejauši. Vienreizēja atslābināšana nozīmē, ka, ja ieraksts neatbilst vienam atribūtam noteikumā, tad tas netiek ņemts vērā (tiek uzskatīts, ka ieraksts atbilst). Otrreizēja atslābināšana pieļauj jau divas neatbilstības un tā tālāk. Redzams, ka atslābināšana pieejamo atribūtu skaitā veido situāciju, ka visi noteikumi atbilst.



5.5. att. **Piemērs balsošanai ar atslābināšanu, kura notiek līdz tiek sasniegts rezultāts:**

(-1) pārbaudes nav, citādi pārbauda atbilstību vai atslābina

Tātad liela noteikumu atslābināšana veido situāciju, ka balso pilnīgi visi noteikumi. Autors norāda, ka viena no svērtās balsošanas priekšrocībām ir, ka samazinās neizšķirtu skaits (pretstatā nesvērtai balsošanai), kas bija iemesls tās ieviešanai abos variantos. Līdz ar to, lai gan neizšķirti var iestāties, to skaitam pārsvarā nevajadzētu būt lielam.

Pirmajā struktūras variantā svērtajā balsošanā piedalās visi indivīdi no kopējās populācijas. Autors nolēma, ka labāk ir uzticēties noteikumiem, kuri trenēšanās laikā ir veikuši mazāk kļūdu, tāpēc indivīdu svari ir vienādi ar to pareizajiem balsojumiem (noteikums saka, ka atbilst un reāli atbilst) pret to pareizo un nepareizo balsojumu (noteikums saka, ka atbilst, bet reāli neatbilst) summu. Šos svarus autors nokopē no attiecīgās vietas autora indivīdu derīguma formulā (5.1). Rezultātā ir samērā vienkārši iegūti noteikumu svari, kuri mazina neizšķirtu balsojumu skaitu.

Otrajā struktūras variantā svērtajā balsošanā piedalās izvēlētie noteikumi. Noteikumu izvēli veic autora izstrādātais selekcijas risinājums. Attiecīgi selekcijas risinājumā ir ieviesta svaru apstrāde, kuri tiek izmantoti pirmā varianta svaru vietā. Līdz ar to varētu teikt, ka izmainītais autora selekcijas risinājuma indivīds sastāv no divām hromosomām: pirmā ir Būla vērtību masīvs, kurš nosaka noteikumu izmantojumu un otrā ir reālo vērtību masīvs, kurš

nosaka svarus. Sākumā svari tiek uzstādīti uz 1.0. Autors nolēma tos galvenokārt mainīt ar mutācijas operatora palīdzību.

Autora krustmija jaunu indivīdu ražošanai no labāko indivīdu grupas izmainītajā selekcijas risinājumā atceras mazāko un lielāko sastapto svaru vērtību pie katras vietas Būla vērtību masīvā. Attiecīgi svari tiek ģenerēti nejauši novērotajos intervālos.

Mutācijas ziņā mutācijas operators izmainītajā selekcijas risinājumā izvēlas vai nu pamainīt kādu svaru, vai apgriezt kādu Būla vērtību. Attiecīgās varbūtības veikt vai nu vienu vai otru ir 50%. Svaru maiņas gadījumā tiem tiek pieskaitīts vai atņemts reāls skaitlis intervālā [0;1), ievērojot nosacījumu, ka svari nevar palikt negatīvi. Negatīvas vērtības gadījumā svarus uzstāda uz 0.01. Jaucējfunkcijas ziņā izmanto Java 8 pieejamo funkciju reālo vērtību masīviem. Tai tiek padots svaru vērtību masīvs, bet lai rezultātu ietekmētu arī Būla vērtību masīvs, svaru vērtību masīvā jaucējfunkcijai pielieto 0, kur Būla vērtību masīvā ir nepatiesas vērtības. Šajā ziņā indivīdus uzskata par vienādiem līdzīgā veidā, ja to svari ir vienādi, ar nosacījumu, ka, veicot salīdzināšanu, svaru masīvos tiek pielietots 0 vietās, kur Būla vērtību masīvā ir nepatiesas vērtības. Tas tā tiek darīts, lai, piemēram, netiktu veidots viens labākais indivīds vairākās kopijās ar atšķirīgiem svariem atribūtos, kurus nemaz nepielieto balsošanā.

Izmainītā selekcijas risinājuma indivīda derīgums ir vienāds ar tā precizitāti. Papildus tam autors nolēma ieviest sodu par klasifikācijām, kuras neseko pamata kārtībai. Tas ir, pirmkārt, indivīdi tiek sodīti par veiktām noteikumu atslābināšanām. Labāk ir izmantot noteikumus, kuri nav jāatslābina. Jo vairāk atslābināšanu veic, jo mazāk uzticams ir gala rezultāts. Otrkārt, indivīdi tiek sodīti par nejaušiem minējumiem. Nejaušu minējumu skaitu ir nepieciešams samazināt, jo tie var mazināt klasifikācijas precizitāti. Pie tam gadījumos, kad nejauši minējumi ir korekti, tie var mākslīgi palielināt indivīdu derīgumu. Attiecīgi, lai ieviestu sodu selekcijas indivīdu derīgums tiek reizināts ar inverso vērtību nevēlamo klasifikāciju proporcijai.

Autors min, ka varētu eksistēt vēl daži citi risinājuma varianti. Pirmkārt, svērto balsošanu varētu aizstāt ar vienkāršu balsošanu bez svariem. Otrkārt, balsošanu varētu neveikt vispār. Šādā ziņā vienkārši pirmais saņemtais balsojums ir klasificētā klase. Šajā variantā, lai izvairītos no situācijas, ka datu ieraksta klasi nosaka neuzticams indivīds, indivīdi pirms klasifikācijas veikšanas tiek sakārtoti pēc to derīgumiem. Tātad vispirms balsot ļauj visderīgākajam indivīdam, tad otrajam derīgākajam indivīdam un tā tālāk.

Risinājuma datu izmantojums ir līdzīgs 4.2. apakš nodaļā aprakstītajam (šķērsvalidācija un testēšana). Testēšanas rezultātu iegūšanai trenēšanu veic ar visiem trenēšanas kopas

datiem. Izņēmumi ir datu diskretizācija un derīguma funkciju datu izmantojums (indivīdu derīgums nav vidējais no 4 reizēm, kā [26] – tas tiek rēķināts ar 1 reizi).

5.2. Veiktie eksperimenti ar izstrādāto klasifikācijas risinājumu

Šajā apakš nodaļā ir aprakstīti nākamie veiktie eksperimenti. Tiek pārbaudīta autora izstrādātā GA klasifikācijas risinājuma spēja veikt klasificēšanu ar UCI repozitorijā pieejamajām datu kopām [17]. Dati pirms to izmantošanas tiek diskretizēti. Datu diskretizācija notiek gan autora risinājuma variantiem, gan WEKA API pieejamajam J48 klasifikatoram [8].

Tika veikti tālāki eksperimenti ar autora risinājuma galvenajiem parametriem, kurus var apskatīt 5.1. tabulā. Parametrus noteica autors pēc validācijas rezultātu analīzes. Šim nolūkam gan tika veikti testi ar manuāli izvēlētām parametru vērtībām, gan testi ar nejauši uzģenerētām vērtībām. Vērtības nejauši tika ģenerētas labāko indivīdu grupas lielumam, labāko indivīdu grupas indivīdiem, krustmijas proporcijai un saglabājamo indivīdu proporcijai.

Tika atklāts, ka šajā gadījumā ir nozīmīgi, ka ir pietiekoši daudz indivīdu. It īpaši tad, ja daļa no tiem nenonāk kopējā populācijā. Mazākā Īrisa kopa [17] ir izņēmums populācijas ziņā ar izmēru 100, jo ar lielākām populācijām rodas problēma nodrošināt, ka indivīdi ir unikāli.

Indivīdu veidošanas no labākās grupas ziņā jāmin, ka noderīgi ir veidot no lielākas indivīdu grupas mazliet mazāku indivīdu grupu. Krustmijas proporcijai ir vēlams būt virs 50% un saglabājamo indivīdu proporcija var būt ap 50%.

5.1. tabula

Autora GA klasifikācijas risinājuma pamata parametri, to skaidrojumi un vērtības

Parametrs	Skaidrojums	Vērtība
Populācijas izmērs	Cik daudz indivīdu risinājumu ir	200 (Īrisa kopai 100 [17])
Paaudzes	Cik daudzās iterācijās atkārto algoritmu	300
Labāko indivīdu grupas lielums	Cik liels % labāko indivīdu piedalās jaunu indivīdu veidošanā	45%
Labāko indivīdu grupas indivīdi	Cik liels % indivīdu tiek uzražots no labāko indivīdu grupas	25%
Krustmijas proporcija	Cik liels % indivīdu tiek izmainīti uz nākamo paaudzi – atlikušais % labāko indivīdu pāriet uz nākamo paaudzi bez izmaiņām	65%
Saglabājamo indivīdu proporcija	Cik liels % labāko indivīdu tiek saglabāti, kad kādas apakš populācijas indivīdi tiek pārvietoti uz kopējo populāciju	55%

Eksperimentos tiek lietotas UCI datu kopas, kuras tika izmantotas arī 4.3. apakš nodaļā [17]. Papildus tām tiek pievienotas vēl citas UCI datu kopas, kuras ir redzamas 5.2. tabulā [17]. Arī šo datu kopu dati tāpat tiek nejauši sajaukti un sadalīti testēšanas (20% datu) un trenēšanas kopās. Par datu kopām norādītā informācija atbilst iepriekšējiem pieņēmumiem (atribūtu skaits iekļauj klasi, norādītais instanču daudzums ir visiem datiem) [17].

5.2. tabula

Eksperimentiem pievienotās datu kopas ar to atribūtiem, instancēm un aprakstiem

UCI datu kopa (atribūti-instances)	Īss apraksts
Īriss (5-150) [17]	3 klašu klasifikācijas uzdevums (īrisa stādu veidi), atribūti ir stādu fiziskie mērījumi [17]
Dermatoloģija (35-366) [17]	6 klašu klasifikācijas uzdevums (dažādas slimības), atribūti ir iegūti no pacientiem (pamatā no ādas paraugiem) [17]
Sēnes (23-8124) [17]	2 klašu klasifikācijas uzdevums (ēdama vai neēdama sēne), atribūti raksturo sēņu paraugus [17]
Desas (10-958) [17]	2 klašu klasifikācijas uzdevums (“X” spēlētājs uzvarēja vai neuzvarēja), atribūti raksturo, kas ir iezīmēts lauciņos [17]
Lapu bloku klasifikācija (11-5473) [17]	5 klašu klasifikācijas uzdevums (dažādi tipi, piemēram, teksta bloks), atribūti raksturo dažādus bloku mērījumus [17]

Autors izšķir pavisam 6 dažādus sava GA klasifikācijas risinājuma variantus. Tos var identificēt pēc sekojošās 5.3. tabulas. Risinājumi ir atšķirīgi ar saviem balsošanas veidiem un noteikumu atlasi.

5.3. tabula

Autora GA klasifikācijas risinājuma iespējamie varianti

Autora risinājuma varianta nosaukums	Balsošanas veids	Atlasītie noteikumi
Autora 1.	Svērtā balsošana	Visi noteikumi
Autora 2.	Svērtā balsošana	Selekcijas izvēlēti
Autora 3.	Nesvērtā balsošana	Visi noteikumi
Autora 4.	Nesvērtā balsošana	Selekcijas izvēlēti
Autora 5.	Balsošanas nav	Visi noteikumi
Autora 6.	Balsošanas nav	Selekcijas izvēlēti

Katrs autora risinājuma variants tika pārbaudīts ar vairākām datu kopām. Iegūtie validācijas un testēšanas rezultāti ir redzami 5.4. tabulā. Testēšanas rezultātu iegūšanai trenēšanu veic ar visiem trenēšanas kopas datiem.

5.4. tabula

Autora GA klasifikācijas risinājuma variantu iegūtās validācijas un testēšanas precizitātes

UCI datu kopa (atribūti-instances)	Autora klasifikācijas risinājuma variantu rezultāti – vispirms validācijas, tad testēšanas precizitāte					
	Autora 1.	**Autora 2.**	**Autora 3.**	**Autora 4.**	**Autora 5.**	**Autora 6.**
Jonosfēra (35-351) [17]	90,39% 90%	91,48% 94,29%	90,39% 90%	92,52% 94,29%	76,49% 77,14%	88,26% 88,57%
Viskonsinas krūts vēža prognozēšana (34-198) [17]	75,59% 65%	68,72% 60%	76,26% 65%	76.23% 62,5%	77,16% 72,5%	71,83% 70%
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	93,42% 92,98%	95,4% 94,74%	93,44% 94,74%	96,27% 92,11%	61,2% 68,42%	94,36% 91,23%
Pilsētas zemes tipi (148-675) [12, 13, 17]	80,63% 76,53%	81,46% 76,73%	80,42% 75,15%	79,38% 74,75%	68,54% 69,82%	76,04% 67,65%
Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	62% 63,04%	68,18% 63,91%	62,43% 62,61%	68,07% 64,35%	53,2% 52,61%	66,99% 65,22%
Aritmija (280-452) [17]	72,09% 68,89%	71,54% 70%	69,61% 73,33%	72,4% 68,89%	61,1% 66,67%	63,58% 66,67%
Muskuss (versija 1) (169-476) [17]	86,07% 70,53%	87,38% 80%	86,08% 78,95%	86,86% 80%	57,78% 57,89%	80,06% 74,74%
Īriss (5-150) [17]	95% 93,33%	94,17% 93,33%	93,33% 96,67%	95% 93,33%	90,83% 96,67%	93,33% 93,33%
Dermatoloģija (35-366) [17]	97,68% 95,89%	95,99% 95,89%	96,99% 94,52%	97,24% 97,26%	94,27% 91,78%	96,68% 93,15%
Sēnes (23-8124) [17]	92,06% 91,82%	98,92% 98,83%	92,2% 94,4%	98,91% 98,83%	79,96% 75,08%	97% 96,49%
Desas (10-958) [17]	75,16% 73,44%	91,12% 94,27%	71,42% 71,35%	82,53% 83,33%	65,63% 64,06%	81,42% 84,38%
Lapu bloku klasifikācija (11-5473) [17]	95,48% 94,79%	96,37% 95,34%	94,95% 93,79%	96,21% 94,89%	89,74% 89,13%	95,36% 95,07%

Pirmkārt, redzams, ka noteikumu selekcija pamatā uzlabo iegūto precizitāti, lai gan pastāv izņēmumi, piemēram, Viskonsinas krūts vēža prognozēšanas kopā [17] noteikumu selekcija precizitāti samazina.

Lai gan autora 5. risinājuma variants ieguva vislabāko rezultātu Viskonsinas krūts vēža prognozēšanas kopā [17], tā rezultāti, salīdzinot ar pārējiem autora variantiem, pārsvarā ir vissliktākie. Tomēr autora 6. variants parāda to, ka balsošanas neveikšana var būt

konkurētspējīga precizitātes ziņā. Noteikumu selekcija šajā gadījumā var palielināt precizitāti pat par dažiem desmitiem procentu, piemēram, Viskonsinas krūts vēža diagnosticēšanas datu kopā [17] autora 5. variants ieguva 61,2% validācijas precizitāti, kas autora 6. variantā tiek uzlabota līdz pat 94,36%. Redzams, ka, neveicot balsošanu, pareizo noteikumu izvēle ir ļoti nozīmīga.

Veicot autora risinājuma variantu salīdzināšanu, kuri izmanto noteikumu selekciju (varianti 2., 4. un 6.), redzams, ka vismazāko precizitāti pārsvarā iegūst autora 6. variants. Autora 2. un 4. varianti precizitātes ziņā ir līdzīgi. Dažos gadījumos labāku rezultātu iegūst autora 2. variants, bet citos autora 4. variants. Redzams, ka šajā gadījumā svērtās balsošanas izmantošana nedot viennozīmīgu ieguvumu precizitātes ziņā.

Papildus precizitātēm tika izmērīts arī nestandarta klasifikāciju (minēšana, vai klasificēšana ar noteikumu atslābināšanu) apjoms autora risinājuma variantiem. Šie mērījumi ļauj spriest, kuri risinājuma varianti uz nestandarta klasifikācijām paļaujas vairāk vai mazāk. Attiecīgos mērījumus var apskatīt 5.5. tabulā. Autors norāda, ka pirmā vērtība ir kopējais nestandarta klasifikāciju skaits 10 validācijas reizēs, kamēr otrā vērtība ir nestandarta klasifikāciju skaits 1 testēšanas reizē. Līdz ar to pirmā vērtība var būt ievērojami lielāka par otro vērtību.

5.5. tabula

Autora GA klasifikācijas risinājuma variantu izmantotais minēšanu un klasifikāciju ar atslābināšanu apjoms

UCI datu kopa (atribūti-instances)	**Autora klasifikācijas risinājuma variantu rezultāti – vispirms kopējais minēšanu un klasifikāciju ar atslābināšanu apjoms 10 validācijas reizēm, tad kopējais apjoms testēšanai**					
	Autora 1.	**Autora 2.**	**Autora 3.**	**Autora 4.**	**Autora 5.**	**Autora 6.**
Jonosfēra (35-351) [17]	0 0	2 0	1 0	3 3	0 0	4 0
Viskonsinas krūts vēža prognozēšana (34-198) [17]	0 0	0 0	2 0	8 6	0 0	1 0
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	0 0	0 0	1 0	5 4	0 0	10 3
Pilsētas zemes tipi (148-675) [12, 13, 17]	13 34	25 106	21 64	39 138	16 42	36 162

5.5. tab. turpinājums

Autora GA klasifikācijas risinājuma variantu izmantotais minēšanu un klasifikāciju ar atslābināšanu apjoms

Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	0 0	0 0	16 0	1 0	0 0	1 0
Aritmija (280-452) [17]	5 1	11 1	25 1	11 4	5 2	29 6
Muskuss (versija 1) (169-476) [17]	0 0	0 0	43 6	3 1	0 0	25 6
Īriss (5-150) [17]	0 0	0 0	4 6	1 0	0 0	1 0
Dermatoloģija (35-366) [17]	1 1	13 2	2 4	13 1	1 0	12 3
Sēnes (23-8124) [17]	0 0	0 0	40 0	0 0	0 0	0 0
Desas (10-958) [17]	0 0	0 0	22 5	36 6	0 0	2 0
Lapu bloku klasifikācija (11-5473) [17]	0 0	9 1	22 16	33 2	0 0	34 12

Nestandarta klasifikāciju ziņā ir redzams, ka vairākos gadījumos tās nemaz netiek pielietotas. Pilsētas zemes tipu datu kopā [12, 13, 17] pielietotais nestandarta klasifikāciju skaits ir vislielākais.

Interesanti, ka pārsvarā noteikumu selekcija noved pie lielāka nestandarta klasifikāciju skaita, bet arī lielākas precizitātes. Tas vedina domāt, ka iespējama metode šo klasifikāciju skaita samazināšanai ir daudzi noteikumi, bet daudzi noteikumi arī samazina kopējo precizitāti. Var spriest, ka slikti noteikumi balso nepareizi, bet palīdz izšķirt strīdus.

Tiesa autora 3. un 4. variantu ziņā dažviet ir novērojams pretējais, ka noteikumu selekcija samazina nestandarta klasifikāciju skaitu. Attiecīgi ir jānorāda, ka autora 3., 4. un 6. variantos nestandarta klasifikāciju skaits ir vislielākais. Tātad noteikumu selekcija var arī samazināt nestandarta klasifikāciju skaitu, ja tas jau sākotnēji ir diezgan liels. Attiecīgi nesvērta balsošana (3. un 4. variants) var palielināt nestandarta klasifikāciju skaitu.

Autora 1., 2. un 5. varianti izmanto vismazāko nestandarta klasifikāciju skaitu. 1. un 5. variants šajā ziņā ir pārsvarā vienādi. Autora risinājuma 2. variantā nestandarta klasifikāciju ir mazliet vairāk.

Tā kā autora risinājuma 2. un 4. varianti ir līdzīgi iegūtās precizitātes ziņā, bet 4. variantā nestandarta klasifikāciju skaits ir lielāks nekā 2. variantā, tad var secināt, ka autora risinājuma 2. variants ir labākais, salīdzinot ar pārējiem pieciem.

Tālāk tiek salīdzināts autora izstrādātā risinājuma 2. variants ar WEKA J48 [8]. Attiecīgi 5.6. tabulā var redzēt WEKA J48 [8] un autora klasifikācijas risinājuma 2. varianta iegūtos rezultātus uz maģistra darbā izmantotajām UCI datu kopām [17].

5.6. tabula

Autora klasifikācijas risinājuma otrā varianta rezultātu salīdzinājums ar WEKA J48, iegūtās precizitātes validācijas un testēšanas kopām

UCI datu kopa (atribūti-instances)	Rezultāti – vispirms validācijas, tad testēšanas precizitāte		
	WEKA J48 rezultāti	Autora klasifikācijas risinājuma 2. varianta rezultāti	Iegūtais autora uzlabojums vai zaudējums
Jonosfēra (35-351) [17]	90,04% 88,57%	91,48% 94,29%	+1,44% +5,72%
Viskonsinas krūts vēža prognozēšana (34-198) [17]	77,16% 72,5%	68,72% 60%	-8,44% -12,5%
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	95,62% 97,37%	95,4% 94,74%	-0,22% -2,63%
Pilsētas zemes tipi (148-675) [12, 13, 17]	79,79% 68,84%	81,46% 76,73%	+1,67% +7,89%
Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	69,27% 64,78%	68,18% 63,91%	-1,09% -0,87%
Aritmija (280-452) [17]	72,68% 71,11%	71,54% 70%	-1,14% -1,11%
Muskuss (versija 1) (169-476) [17]	88,97% 76,84%	87,38% 80%	-1,59% +3,16%
Īriss (5-150) [17]	95,83% 93,33%	94,17% 93,33%	-1,66% 0%
Dermatoloģija (35-366) [17]	95,27% 94,52%	95,99% 95,89%	+0,72% +1,37%
Sēnes (23-8124) [17]	100% 100%	98,92% 98,83%	-1,08% -1,17%
Desas (10-958) [17]	83,01% 82,81%	91,12% 94,27%	+8,11% +11,46%
Lapu bloku klasifikācija (11-5473) [17]	96,64% 96,35%	96,37% 95,34%	-0,27% -1,01%

Kopumā redzams, ka WEKA J48 [8] pārsvarā pārspēj autora klasifikācijas risinājuma 2. variantu, bet ne viennozīmīgi visos gadījumos. Tas ir WEKA J48 [8] autora risinājuma 2. variantu pārspēj 7 gadījumos no 12 (abos mērījumos). Muskuss (versija 1) datu kopā [17] WEKA J48 [8] ir labāks validācijas rezultāts, bet autora risinājuma 2. variantam ir labāks testēšanas rezultāts.

Novērojamās atšķirības pamatā ir dažos procentos. Vislielākās atšķirības ir vērojamas apmēram 10% apmērā. Viskonsinas krūts vēža prognozēšanas datu kopā [17] WEKA J48 [8] pārspēj autora risinājuma 2. variantu par 8,44% validācijas ziņā un 12,5% testēšanas ziņā. Desu kopā [17] autora klasifikācijas risinājuma 2. variants pārspēj WEKA J48 [8] par 8,11% validācijas ziņā un 11,46% testēšanas ziņā. Ievērojami ir arī autora risinājuma 2. varianta testēšanas rezultāti Jonosfēras [17] un Pilsētas zemes tipu datu kopās [12, 13, 17], kuri ir par 5,72% un 7,89% labāki nekā attiecīgie WEKA J48 [8] rezultāti.

5.3. Klasifikācijas risinājuma rezultātu salīdzinājums ar citiem līdzīgiem rezultātiem

Šeit ir apskatīti līdzīgi rezultāti no citiem pētījumiem. Attiecīgi ir dots autora novērtējums. Salīdzinājums tiek veikts, apskatot tās UCI datu kopas, kuras parādās autora maģistra darbā [17].

Pētījumā [18] tika veikti eksperimenti ar UCI Īrisa datu kopu [17]. Tika iegūta 72,2% testēšanas precizitāte [18], kas ir salīdzinoši maza, apskatot pārējos rezultātus. Autora risinājuma 2. variants ieguva 93,33% testēšanas precizitāti. Vēl lielāku testēšanas precizitāti, 96,67%, ieguva autora risinājuma 3. un 5. varianti.

Pētījumā [21] tika veikti eksperimenti ar UCI Viskonsinas krūts vēža prognozēšanas kopu [17]. Iegūtā labākā validācijas precizitāte bija 77,57% [21]. Autora risinājuma 2. variants ieguva mazāku 68,72% validācijas precizitāti, lai gan lielākā autora iegūtā validācijas precizitāte ir 77,16% (autora 5. variants), kas ir tikai mazliet mazāka.

Pētījumā [24] tika veikta risinājuma šķērsvalidācija uz Sēņu [17], Īrisa [17] un Desu datu kopām [17], kuras ir pieejamas UCI repozitorijā [17]. Šajā ziņā ir jānorāda, ka pētījumā [24] tika izmantota 10 reižu šķērsvalidācija Sēņu [17] un Desu datu kopām [17] un 2 reižu šķērsvalidācija Īrisa kopai [17, 24]. 5.7. tabulā ir redzami pētījumā [24] piedāvātā risinājuma rezultāti (vairāku populāciju GA) [24] un attiecīgie maģistra darba autora iegūtie rezultāti. No autora rezultātiem tiek pamatā piedāvāti autora klasifikācijas risinājuma 2. varianta rezultāti,

kā arī labākie autora iegūtie rezultāti gadījumā, ja kāds cits autora variants ieguva labāku rezultātu.

5.7. tabula

Autora rezultātu salīdzinājums ar vairāku populāciju GA pētījuma rezultātiem, iegūtās validācijas precizitātes

UCI datu kopa (atribūti-instances)	Vairāku populāciju GA validācijas precizitāte	Autora iegūtās validācijas precizitātes
Īriss (5-150) [17]	98,67% [24]	94,17% (autora 2. variants) 95% (autora 1. un 4. varianti)
Sēnes (23-8124) [17]	98,16% [24]	98,92% (autora 2. variants)
Desas (10-958) [17]	93,01% [24]	91,12% (autora 2. variants)

Redzams, ka autora klasifikācijas risinājuma 2. variants ieguva lielāku precizitāti Sēņu datu kopā [17], bet citur pētījuma [24] rezultāti ir lielāki [24]. Jāmin, ka atšķirības šajā ziņā ir samērā mazas (mazākas par 5%) [24].

Pētījumā [9] tika veikta desmitkārtīga testēšana uz UCI Viskonsinas krūts diagnosticēšanas [17], Dermatoloģijas [17], Debrecenas diabētiskas retinopātijas [4, 7, 17], Jonosfēras [17], Īrisa [17] un Lapu bloku klasifikācijas datu kopām [9, 17]. 5.8. tabulā ir redzami pētījumā [9] piedāvātā nejaušo mežu GA risinājuma rezultāti (vidējais rezultāts no desmitkārtīgas testēšanas) [9] un attiecīgie autora iegūtie rezultāti. No autora rezultātiem tiek izmantoti klasifikācijas risinājuma 2. varianta validācijas rezultāti, kā arī labākie autora citu variantu rezultāti, ja tie pārspēj 2. varianta rezultātus.

5.8. tabula

Autora iegūto validācijas rezultātu salīdzinājums ar nejaušo mežu algoritma uzlabošanas pētījuma desmitkārtīgās testēšanas vidējās precizitātes rezultātiem

UCI datu kopa (atribūti-instances)	Nejaušo mežu GA desmitkārtīgas testēšanas precizitāte	Autora iegūtās validācijas precizitātes
Jonosfēra (35-351) [17]	93,73% [9]	91,48% (autora 2. variants) 92,52% (autora 4. variants)
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	98,14% [9]	95,4% (autora 2. variants) 96,27% (autora 4. variants)

5.8. tab. turpinājums

Autora iegūto validācijas rezultātu salīdzinājums ar nejaušo mežu algoritma uzlabošanas pētījuma desmitkārtīgās testēšanas vidējās precizitātes rezultātiem

Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	68,26% [9]	68,18% (autora 2. variants)
Īriss (5-150) [17]	97,04% [9]	94,17% (autora 2. variants) 95% (autora 1. un 4. varianti)
Dermatoloģija (35-366) [17]	98,36% [9]	95,99% (autora 2. variants) 97,68% (autora 1. variants)
Lapu bloku klasifikācija (11-5473) [17]	97,34% [9]	96,37% (autora 2. variants)

Autora klasifikācijas risinājuma rezultāti ir sliktāki par pētījuma [9] rezultātiem apskatītajās 6 UCI datu kopās [9, 17]. Arī šajā gadījumā atšķirības nav lielas (mazākas par 5%) [9].

Kopumā autora klasifikācijas rezultāti, salīdzinot ar citiem līdzīgiem pētījumiem ir sliktāki. Tomēr atsevišķos gadījumos autora klasifikācijas risinājums spēj sniegt labāku rezultātu un novērotās atšķirības pārsvarā nav lielas.

6. IZSTRĀDĀTAIS ĢENĒTISKO ALGORITMU ANSAMBĻA KLASIFIKĀCIJAS RISINĀJUMS

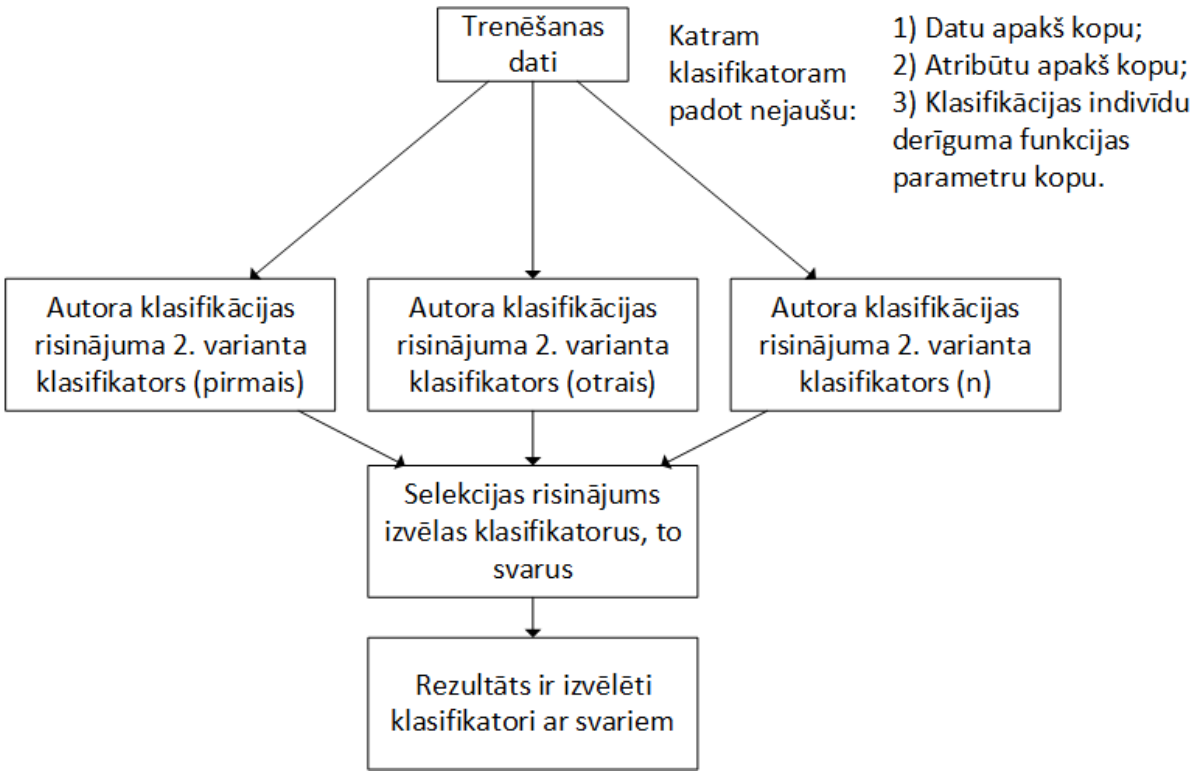
Šajā nodaļā ir aprakstīts autora izveidotais GA ansambļa klasifikācijas risinājums un veiktie eksperimenti ar to. Ansambli veido vairāki autora klasifikācijas risinājuma 2. varianta klasifikatori. Attiecīgais autora klasifikācijas risinājums ir aprakstīts 5. nodaļā. Tiek izmantota arī autora izstrādātā selekcija rezultātu apvienošanai, kura ir aprakstīta 4. nodaļā.

Pirmā apakš nodaļa apraksta risinājumu. Otrā apakš nodaļa ir par veiktajiem eksperimentiem, bet trešā apakš nodaļa sniedz salīdzinājumu ar citiem līdzīgiem rezultātiem.

6.1. Izstrādātā ansambļa risinājuma apraksts

Autora GA ansambļa klasifikācijas risinājums tāpat kā iepriekšējās nodaļās aprakstītie autora risinājumi (4. un 5. nodaļa) ir izveidots valodā Java un izmanto WEKA [8]. Informāciju par autora izstrādātā koda apskatīšanu un izmēģināšanu var iegūt 1. pielikumā. Šī risinājuma pamata ideja ir iepriekš izstrādāto (4. un 5. nodaļā) kombinēt ar ansambļa izveides algoritmu (*bagging*) [15], tā izveidojot klasifikatoru ansambli [15].

Klasifikāciju veic ar autora klasifikācijas risinājuma 2. varianta klasifikatoriem. Klasifikatorus sakombinē ar autora selekcijas risinājuma palīdzību, kurš, izvēlas, kurus uzbūvētos klasifikatorus lietot un nosaka to svarus. Attiecīgi ir iespējams variants bez klasifikatoru selekcijas un ar klasifikatoru selekciju. Risinājuma kopskats ir redzams 6.1. att. **Autora izstrādātais GA ansambļa klasifikācijas risinājums**.



6.1. att. **Autora izstrādātais GA ansambļa klasifikācijas risinājums**

Vairāku vienādu klasifikatoru salikšana vienā ansamblī visticamāk, ka uzlabojumu nedos, jo tie attiecīgi balsos vienādi. Autors līdz ar to izmanto algoritmu, kuru pielieto nejaušo mežu algoritmā, lai ieviestu izmaiņas klasifikatoros (*bagging*) [15]. Tajā attiecīgi vairākus klasifikatorus uzbūvē ar nejauši izvēlētām oriģinālo trenēšanas datu apakš kopām, bet viens ieraksts apakš kopās var būt iekļauts vairākas reizes. Tātad daļa ierakstu tiks iekļauti vairākas reizes, bet daļa ierakstu vispār netiks iekļauti pat ja apakš kopas izmērs būs vienāds ar sākotnējās trenēšanas kopas izmēru [15].

Tiek izmantota arī ideja, ka klasifikatorus var būvēt arī ar atribūtu apakškopām un katram no tiem var nejauši pamainīt parametrus [15].

Atribūtu ziņā viens un tas pats atribūts vairākas reizes neko nedot. Attiecīgi autors nosaka atribūtu izmantošanas parametru, cik procentu no visiem atribūtiem izmantot. Šajā ziņā risinājumā iet cauri visiem iespējamajiem atribūtiem un tiek piemērota attiecīgā parametra varbūtība atribūtu iekļaut, vai neiekļaut. Tātad izmantoto atribūtu skaits katram klasifikatoram var būt dažāds, bet tas būs aptuveni līdzīgs noteiktajam atribūtu izmantošanas parametram. Izņēmuma gadījumos, kad, piemēram, atribūti vispār netiek iekļauti, vai tiek iekļauti nepietiekošā daudzumā (nevar uzražot pietiekoši daudz unikālu indivīdu noteiktās populācijas izmērā ar dotajiem atribūtiem), atribūtu izvēles process tiek atkārtots.

Algoritma parametru ziņā autors izvēlējās pamainīt vērtības iekš klasifikācijas indivīdu derīguma funkcijas (5.1). Tiek noteikts parametrs, kas ir procenti no kopējā trenēšanas datu

apjoma, kas nosaka maksimālo uzģenerēto ierakstu skaita vērtību. Tas ir maksimālais iespējamais pieprasītais indivīdu skaits ir atbilstošs noteiktajam procentam no visiem trenēšanas datiem (naturālam skaitlim). Katram klasifikatoram tiek uzģenerētas divas vērtības intervālā no 0 līdz vērtībai, kuru nosaka minētais parametrs. Pirmā vērtība nosaka, cik daudzi ieraksti ir nepieciešami, kurus noteikums pareizi klasificē. Otrā vērtība nosaka, cik daudzi ieraksti ir nepieciešami, kurus noteikums atzīmē, kā atbilstošus vai neatbilstošus sev. Autors norāda, ka šajā ziņā neatkarīgi no uzģenerētās vērtības noteikumi, kuri neizdara nevienu pareizu klasifikāciju, vai saka, ka visi ieraksti ir atbilstoši vai neatbilstoši sev joprojām netiek iekļauti, jo pietiek ar uzģenerētu vērtību 0, lai tas iestātos (derīguma funkcijā tiek pārbaudīts uz vienādu vai mazāku).

Visbeidzot, lai nodrošinātu, ka neveiksmīgi izveidoti klasifikatori netiek iekļauti un ka labiem klasifikatoriem ir lielāka ietekme uz rezultātu, tiek izmantota izstrādātā autora selekcija no maģistra darba 4. nodaļas, lai izvēlētos, kurus klasifikatorus iekļaut vai neiekļaut un kādus svarus tiem piešķirt.

Variantā bez selekcijas joprojām tiek izmantota ideja, ka klasifikatoru svari ir vienādi ar to pareizības procentu, cik procentu trenēšanas ierakstu klasifikators klasificēja pareizi. Protams, variantā bez selekcijas tiek izmantoti visi uzģenerētie klasifikatori.

6.2. Veiktie eksperimenti ar autora izstrādāto ansambļa klasifikācijas risinājumu

Tālāk seko nākamie veiktie eksperimenti ar UCI repozitorija kopām [17]. Šajā gadījumā galvenokārt tiek pārbaudīta autora ansambļa risinājuma rezultativitāte. Datu izmantojums ir identisks 5.2. apakš nodaļā minētajam, kā arī tiek izmantotas tās pašas datu kopas, kuras tika izmantotas 5. nodaļā.

Galvenie parametri ir identiski 5.1. tabulā minētajiem, izņemot populācijas izmēru un paaudžu skaitu. Papildus tiem nāk daži ansamblim specifiski parametri. Parametrus noteica autors pēc validācijas rezultātu analīzes (testi ar nejauši un manuāli izvēlētām vērtībām galvenokārt uz Viskonsinas krūts vēža diagnosticēšanas datu kopas [17]). Vērtības nejauši tika ģenerētas izmantoto atribūtu proporcijai un maksimālā pieprasītā ierakstu skaita proporcijai. Izvēlētos jaunos un izmainītos parametrus var redzēt sekojošajā 6.1. tabulā.

Populācijas izmērs un paaudžu skaits tika samazināti, lai ansambļa izpildes laiks nebūtu pārāk liels. Izmantoto datu proporcija tika noteikta kā 100%, kā minēts aprakstā no [15]. Autors min, ka tas tik un tā nozīmē, ka daļa datu ierakstu netiks izmantota visos

klasifikatoros, jo viens un tas pats ieraksts var parādīties vairākas reizes [15]. Tiek izmantota liela daļa atribūtu, bet maksimālā pieprasītā ierakstu skaita proporcija ir samērā maza.

6.1. tabula

Autora ansambļa risinājumam galvenie specifiskie parametri, to apraksti un noteiktās vērtības

Parametra nosaukums	Parametra apraksts	Parametra vērtība
Populācijas izmērs	Izveidoto noteikumu daudzums populācijā	50
Paaudžu skaits	GA iterāciju skaits	100
Klasifikatoru skaits	Cik daudzi autora klasifikācijas risinājuma 2. varianta klasifikatori veido ansambli	100
Izmantoto atribūtu proporcija	Cik liela proporcija atribūtu tiek izmantoti uzbūvētajiem klasifikatoriem	70%
Izmantoto datu proporcija	Cik liela proporcija datu tiek izmantota uzbūvētajiem klasifikatoriem	100%
Maksimālā pieprasītā ierakstu skaita proporcija	Cik daudzi ieraksti var tikt maksimāli pieprasīti, ģenerējot klasifikācijas indivīdu derīguma funkcijas vērtības – parametrs ir izteikts kā naturāls skaitlis, kas iegūts no attiecīgās proporcijas	7%

Veiktie eksperimenti ar autora ansambļa klasifikācijas risinājumu ir redzami sekojošajā 6.2. tabulā. Tabulā vispirms ir redzams validācijas rezultāts, kam seko iegūtais testēšanas rezultāts. Papildus autora rezultātiem 6.2. tabulā ir redzami arī WEKA nejaušo mežu algoritma rezultāti [8].

6.2. tabula

Autora ansambļa klasifikācijas risinājuma un WEKA nejaušo mežu algoritma iegūtie validācijas un testēšanas rezultāti

UCI datu kopa (atribūti-instances)	Rezultāti – vispirms validācijas, tad testēšanas precizitāte		
	Autora ansambļa klasifikācijas risinājums bez selekcijas	Autora ansambļa klasifikācijas risinājums ar selekciju	WEKA nejaušo mežu algoritms
Jonosfēra (35-351) [17]	92,17% 94,29%	92,51% 88,57%	92,17% 94,29%
Viskonsinas krūts vēža prognozēšana (34-198) [17]	77,16% 72,5%	72,72% 70%	78,49% 72,5%
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	96,73% 95,61%	96,29% 96,49%	97,16% 96,49%
Pilsētas zemes tipi (148-675) [12, 13, 17]	84,58% 79,68%	82,29% 75,94%	91,25% 80,28%

6.2. tab. turpinājums

Autora ansambļa klasifikācijas risinājuma un WEKA nejaušo mežu algoritma iegūtie validācijas un testēšanas rezultāti

Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	66,99% 62,61%	68,4% 63,91%	68,4% 65,22%
Aritmija (280-452) [17]	72,09% 68,89%	76,24% 71,11%	74,56% 74,44%
Muskuss (versija 1) (169-476) [17]	90,53% 75,79%	90,01% 78,95%	93,16% 83,16%
Īriss (5-150) [17]	95,83% 90%	95% 90%	95% 93,33%
Dermatoloģija (35-366) [17]	98,69% 97,26%	96,71% 97,26%	98,97% 95,89%
Sēnes (23-8124) [17]	98,94% 96,92%	99,05% 98,89%	100% 100%
Desas (10-958) [17]	76,87% 76,04%	77% 77,08%	94,52% 94,27%
Lapu bloku klasifikācija (11-5473) [17]	93,19% 92,24%	94,56% 93,61%	96,87% 96,44%

Kopumā redzams, ka rezultāti ar vai bez selekcijas ir diezgan līdzīgi. Mazu pārsvaru tomēr gūst variants ar klasifikatoru selekciju. Variants bez selekcijas pārspēj variantu ar selekciju 4 gadījumos (labāks gan validācijas, gan testēšanas rezultāts), bet variants ar selekciju pārspēj variantu bez selekcijas 5 gadījumos. Pārējos trīs gadījumos (Jonosfēra [17], Viskonsinas krūts vēža diagnosticēšana [17] un Muskuss (versija 1) [17]) vienam variantam ir labāks validācijas rezultāts, bet otram testēšanas rezultāts. Var secināt, ka klasifikatoru selekcija var būt noderīga daļā gadījumu.

WEKA nejaušo mežu algoritms [8] pārspēj autora GA ansambļa klasifikācijas risinājuma abus variantus, bet pastāv atsevišķi mērījumi, kuros autora GA ansambļa klasifikācijas risinājuma varianti ieguva labākus rezultātus. Autora GA ansambļa klasifikācijas risinājums ar klasifikatoru selekciju ieguva labākus validācijas rezultātus Jonosfēras [17] un Aritmijas [17] datu kopās un labāku testēšanas rezultātu Dermatoloģijas [17] datu kopā. Autora ansambļa variants bez selekcijas ieguva labāku validācijas rezultātu Īrisa datu kopā [17], kā arī labāku testēšanas rezultātu Dermatoloģijas [17] datu kopā. Pamatā atšķirības rezultātos ir dažos procentos, bet lielākās atšķirības ir vērojamas Pilsētas zemes tipu [12, 13, 17], Muskusa (versijas 1) [17] un Desu [17] datu kopās.

Nestandarta klasifikācijas (minējumi, noteikumu atslābināšana) netika iekļautas rezultātos, jo tās praktiski netiek veiktas. Sanāk, ka tās var notikt iekš individuālajiem klasifikatoriem, bet svērtajā ansamblī nestandarta klasifikācijas jau ir liels retums.

Autora GA ansambļa klasifikācijas risinājuma variants ar klasifikatoru selekciju tiek salīdzināts ar autora GA klasifikācijas 2. varianta (5. nodaļa) risinājumu. Attiecīgie rezultāti ir redzami 6.3. tabulā.

6.3. tabula

Autora GA ansambļa klasifikācijas risinājuma varianta ar klasifikatoru selekciju salīdzinājums ar autora GA klasifikācijas risinājuma 2. variantu, iegūtās validācijas un testēšanas precizitātes

UCI datu kopa (atribūti-instances)	Rezultāti – vispirms validācijas, tad testēšanas precizitāte	
	Autora ansambļa klasifikācijas risinājums ar selekciju	Autora klasifikācijas risinājuma 2. variants
Jonosfēra (35-351) [17]	92,51% 88,57%	91,48% 94,29%
Viskonsinas krūts vēža prognozēšana (34-198) [17]	72,72% 70%	68,72% 60%
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	96,29% 96,49%	95,4% 94,74%
Pilsētas zemes tipi (148-675) [12, 13, 17]	82,29% 75,94%	81,46% 76,73%
Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	68,4% 63,91%	68,18% 63,91%
Aritmija (280-452) [17]	76,24% 71,11%	71,54% 70%
Muskuss (versija 1) (169-476) [17]	90,01% 78,95%	87,38% 80%
Īriss (5-150) [17]	95% 90%	94,17% 93,33%
Dermatoloģija (35-366) [17]	96,71% 97,26%	95,99% 95,89%
Sēnes (23-8124) [17]	99,05% 98,89%	98,92% 98,83%
Desas (10-958) [17]	77% 77,08%	91,12% 94,27%
Lapu bloku klasifikācija (11-5473) [17]	94,56% 93,61%	96,37% 95,34%

Salīdzinot autora ansambļa risinājumu ar autora klasifikācijas risinājumu, redzams, ka autora ansambļa risinājums pārspēj autora klasifikācijas risinājumu 6 gadījumos (abos mērījumos), bet autora klasifikācijas risinājums pārspēj autora ansambļa risinājumu 2 gadījumos (abos mērījumos). 4 gadījumos vienam risinājumam ir labāks validācijas rezultāts, bet otram testēšanas. Tātad ansambļa izmantošana var sniegt uzlabojumus precizitātē.

Lielākās atšķirības precizitātēs ir vērojamas Viskonsinas krūts vēža prognozēšanas [17] datu kopā, kur labāku rezultātu sniedz autora ansambļa klasifikācijas risinājums (10% labāks testēšanas rezultāts) un Desu [17] datu kopā, kur ievērojami labāku rezultātu iegūst autora klasifikācijas risinājums (apmēram par 15% labāks abos mērījumos).

6.3. Ansambļa klasifikācijas risinājuma rezultātu salīdzinājums ar citiem līdzīgiem rezultātiem

Šajā apakš nodaļā iegūtie rezultāti tiek salīdzināti ar rezultātiem citos pētījumos. 5.3. apakš nodaļā ir precīzāk uzrakstīts, kas tieši tiek salīdzināts. Attiecīgos autora un citu pētījumu rezultātus var redzēt 6.4. tabulā, kur pie katra rezultāta ir komentāri, kas tas ir par rezultātu. Autora rezultātu ziņā ir uzrādīti GA ansambļa klasifikācijas abu variantu rezultāti gan bez klasifikatoru selekcijas, gan ar klasifikatoru selekciju.

6.4. tabula

Autora GA ansambļa klasifikācijas rezultātu salīdzinājums ar citu pētījumu rezultātiem

UCI datu kopa (atribūti-instances)	**Autora iegūtās precizitātes (vispirms klasifikācija bez klasifikatoru selekcijas, tad klasifikācija ar klasifikatoru selekciju)**	**Citu pētījumu iegūtās precizitātes**
Jonosfēra (35-351) [17]	92,17% (validācija) 92,51% (validācija)	93,73% (10 reižu testēšana) [9]
Viskonsinas krūts vēža prognozēšana (34-198) [17]	77,16% (validācija) 72,72% (validācija)	77,57% (validācija) [21]
Viskonsinas krūts vēža diagnosticēšana (32-569) [17]	96,73% (validācija) 96,29% (validācija)	98,14% (10 reižu testēšana) [9]
Debrecenas diabētiskas retinopātijas datu kopa (20-1151) [4, 17], bāzēta uz "Messidor" attēlu datu kopas, kuru laipni nodrošina "Messidor" programmas partneri (skatīt http://www.adcis.net/en/DownloadThirdParty/Messidor.html) [7]	66,99% (validācija) 68,4% (validācija)	68,26% (10 reižu testēšana) [9]

6.4. tab. turpinājums

Autora GA ansambļa klasifikācijas rezultātu salīdzinājums ar citu pētījumu rezultātiem

Īriss (5-150) [17]	95,83% (validācija) 90% (testēšana) 95% (validācija) 90% (testēšana)	72,2% (testēšana) [18] 98,67% (validācija) [24] 97,04% (10 reižu testēšana) [9]
Dermatoloģija (35-366) [17]	98,69% (validācija) 96,71% (validācija)	98,36% (10 reižu testēšana) [9]
Sēnes (23-8124) [17]	98,94% (validācija) 99,05% (validācija)	98,16% (validācija) [24]
Desas (10-958) [17]	76,87% (validācija) 77% (validācija)	93,01% (validācija) [24]
Lapu bloku klasifikācija (11-5473) [17]	93,19% (validācija) 94,56% (validācija)	97,34% (10 reižu testēšana) [9]

Salīdzinot ar 5.3. apakš nodaļā novēroto, apstiprinās tas, ka autora GA ansambļa klasifikācijas risinājums ir kopumā rezultatīvāks par autora GA klasifikācijas risinājumu. Joprojām tiek iegūts labāks testēšanas rezultāts Īrisa kopā [17], salīdzinot ar pētījuma [18] testēšanas rezultātu [18], kā arī labāks validācijas rezultāts Sēņu kopā [17], salīdzinot ar pētījuma [24] validācijas rezultātu [24]. Šajā ziņā abi autora GA ansambļa klasifikācijas risinājuma varianti pārspēj attiecīgos rezultātus. Papildus tam labāki rezultāti ir iegūti arī Debrecenas diabētiskas retinopātijas datu kopā [4, 7, 17], kur autora GA ansambļa klasifikācijas risinājuma ar klasifikatoru selekciju validācijas rezultāts pārspēj pētījuma [9] 10 reižu testēšanas rezultātu [9], kā arī Dermatoloģijas datu kopā [17], kur autora GA ansambļa klasifikācijas risinājuma bez klasifikatoru selekcijas validācijas rezultāts pārspēj pētījuma [9] 10 reižu testēšanas rezultātu [9].

Novērojamās atšķirības iegūtajās precizitātēs pamatā nav lielas (pārsvarā mazākas par 5%), izņemot dažus gadījumus. Īrisa kopā [17] pētījuma [18] testēšanas rezultāts [18] ir ievērojami mazāks par autora abu GA ansambļa klasifikācijas variantu iegūto testēšanas rezultātu (17,8% atšķirība), Desu kopā [17] autora iegūtie validācijas rezultāti ir ievērojami mazāki par pētījuma [24] validācijas rezultātu [24] (16,14% un 16,01% atšķirības).

Kopumā autora rezultāti ir sliktāki par citu pētījumu rezultātiem, bet autora GA ansambļa klasifikācijas risinājums ir palielinājis to gadījumu skaitu, kad tiek iegūti labāki rezultāti.

REZULTĀTI UN DISKUSIJA

Autors izveidoja jaunu pieeju atribūtu selekcijai ar GA klasifikācijas problēmai, kura izmanto unikālu struktūru un jaunas idejas krustmijai un mutācijai. Apstiprinājās tas, ka GA atribūtu selekcija var izmest lielu daļu atribūtu, kā arī iegūt dažu % precizitātes uzlabojumu. Ievērojams rezultāts ir izmesti 70% atribūtu un vairāk daļā gadījumu, bet uzlabojumi precizitātē ne vienmēr tika iegūti. Vislielākais uzlabojums eksperimentos bija par 10%.

Autors izveidoja GA klasifikācijas risinājumu, kurš pielieto jaunas autora idejas klasifikācijas indivīdu derīguma funkcijai, kā arī izveidoto klasifikācijas noteikumu sakombinēšanai. Tika noteikts, ka svērtā balsošana un noteikumu selekcija ar izveidoto GA selekcijas risinājumu samazina nestandarta klasifikāciju skaitu un palielina precizitāti. Salīdzinot ar citiem risinājumiem, atsevišķos gadījumos autora risinājums spēj iegūt lielāku precizitāti.

Autors izveidoja GA ansambļa klasifikācijas risinājumu, kurš izmanto ansambļa izveides algoritmu (*bagging*) [15]. Šī pieeja pārspēja autora GA klasifikācijas risinājumu, bet, salīdzinot ar citiem risinājumiem, lielāka precizitāte tiek iegūta tikai atsevišķos gadījumos.

Vispirms darbā tika aprakstīts tā konteksts – klasifikācijas problēma un GA.

Tālāk tika apskatīti vairāki pētījumi par klasifikāciju, kuri izmanto GA. Tika secināts, ka ir aktuāli pielietot GA klasifikācijas problēmai, jo ir veikti daudzi jauni pētījumi, kuri ir izveidojuši efektīvus risinājumus. Dažādie GA pielietojumi klasifikācijā iekļauj lēmumu koku optimizēšanu, klasifikatoru ansambļa struktūras atbalstīšanu, neironu tīklu optimizēšanu un atribūtu selekciju.

Tika izstrādāti autora GA atribūtu selekcijas, klasifikācijas un ansambļa klasifikācijas risinājumi. Tie tika pārbaudīti uz UCI mašīnmācīšanās repozitorija datu kopām [17] ar 10 reižu šķērsvalidāciju un testēšanas kopām.

SECINĀJUMI

Maģistra darbs ir sasniedzis ieplānoto: ir izpētīti veiktie pētījumi par klasifikāciju, kuri izmanto GA, ir izstrādāti autora GA atribūtu selekcijas, klasifikācijas un ansambļa klasifikācijas risinājumi un izstrādātais ir novērtēts uz publiski pieejamām datu kopām.

GA atribūtu selekcijas risinājumā autora jaunā ideja krustmijai ļauj no labāko indivīdu grupas izveidot tiem līdzīgu jaunu indivīdu grupu. Izstrādātā autora mutācija ļauj palielināt indivīdu dažādību, likvidējot vienādus indivīdus.

Izveidotais autora GA atribūtu selekcijas risinājums ir noderīgs, jo veiktie eksperimenti parādīja, ka tas samazina nepieciešamo atribūtu skaitu. Pie tam šis atribūtu skaita samazinājums ir diezgan ievērojams. Veiktajos eksperimentos tas apmēram iekļāvās 50%-70% intervālā. Daļā gadījumu ir iespējams iegūt arī dažu % precizitātes uzlabojumu.

GA klasifikācijas risinājumā autora izstrādātā derīguma funkcija liek uzsvaru uz pareizu datu ierakstu klasificēšanu, kurus noteikumi klasificē pareizi mazā apjomā. Tas ļauj neizlaist datu ierakstus no trenēšanas kopas. Tika izveidots klasifikācijas noteikumu kombinēšanas variants ar svērto balsošanu un noteikumu selekciju, kas samazina nestandarta klasifikāciju skaitu un palielina precizitāti.

Autora GA ansambļa klasifikācijas risinājums uzlabo autora GA klasifikācijas risinājumu, ļaujot iegūt lielāku precizitāti. Klasifikatoru dažādība tiek iegūta trijos veidos, izmantojot datu apakš kopas, atribūtu apakš kopas un pamainot klasifikācijas indivīdu derīguma funkcijas vērtības.

Kopumā autora klasifikācijas risinājumi ir sliktāki par citiem apskatītajiem risinājumiem, bet atsevišķos gadījumos var iegūt labāku precizitāti.

Ar izstrādātajiem risinājumiem tika veikti apjomīgi testi, lai noteiktu izmantojamos parametrus un izvērtētu rezultativitāti. Pie tam tika izvērtēti dažādi klasifikācijas risinājumu varianti.

Turpmākos pētījumos varētu izstrādāt citus GA ansambļa variantus un klasifikācijas noteikumu kombinēšanas veidus.

IZMANTOTĀ LITERATŪRA UN AVOTI

1. E. Ozoliņš, "Ģenētisko algoritmu papildināšana ceļojošā tirgoņa problēmas risināšanai," bakalaura darbs, LU Datorikas fakultāte, Latvijas Universitāte, Rīga, 2016.
2. E. Ozoliņš, "Ģenētisko algoritmu izmantošana klasifikācijā," maģistra kursa darbs, LU Datorikas fakultāte, Latvijas Universitāte, Rīga, 2018.
3. S. Akbar, M. Hayat, M. Iqbal, M.A. Jan, "iACP-GAEnsC: Evolutionary genetic algorithm based ensemble classification of anticancer peptides by utilizing hybrid feature space," *Artificial Intelligence in Medicine,* vol. 79, 2017, pp. 62-70.
4. B. Antal, A. Hajdu, "An ensemble-based system for automatic screening of diabetic retinopathy," *Knowledge-Based Systems*, vol. 60, 2014, pp. 20-27.
5. S. Chatterjee u.c., "Structural failure classification for reinforced concrete buildings using trained neural network based multi-objective genetic algorithm," *Structural Engineering and Mechanics,* vol. 63, no. 4, 2017, pp. 429-438.
6. "The Zettabyte Era: Trends and Analysis," white paper, CISCO, Jun. 2017.
7. E. Decencière u.c, "Feedback on a publicly distributed database: the Messidor database," *Image analysis & Stereology*, vol. 33, no. 3, Aug. 2014, pp. 231-234.
8. F. Eibe, H.A. Mark, W.H. Ian, "The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques", Morgan Kaufman, Fourth Edition, 2016," *The University of Waikato,* 2016. Pieejams: https://www.cs.waikato.ac.nz/ml/weka/Witten_et_al_2016_appendix.pdf
9. E. Elyan, M.M. Gaber, "A genetic algorithm approach to optimising random forests applied to class engineered data," *Information Sciences*, vol. 384, 2017, pp. 220-234.
10. E. Gibaja, S. Ventura, "A Tutorial on Multi-Label Learning," *ACM Computing Surveys*, vol. 9, no. 4, 2015, art. 52.
11. D. Iskrich, D. Grigoriev, "Generating long-term trading system rules using a genetic algorithm based on analyzing historical data," in *Proc. 20.th Conf. Open Innovations Association*, 2017, pp. 91-97. Pieejams: http://ieeexplore.ieee.org/document/8071297.
12. B. Johnson, "High resolution urban land cover classification using a competitive multi-scale object-based approach," *Remote Sensing Letters*, vol. 4, no. 2, 2013, pp.131-140.
13. B. Johnson, Z. Xie, "Classifying a high resolution image of an urban area using super-object information," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 83, 2013, pp. 40-49.

14. T.M. Khoshgoftaar, E.B. Allen, “Controlling overfitting in classification-tree models of software quality,” *Empirical Software Engineering*, vol. 6, no. 1, 2001, pp. 59-79.
15. L.I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. Hoboken, New Jersey, United States of America: John Wiley & Sons, 2004.
16. H. Li u.c., “Genetic algorithm for the optimization of features and neural networks in ECG signals classification,” *Scientific Reports*, vol. 7, no. 41011, 2017.
17. M. Lichman, “UCI Machine Learning Repository,” *Irvine, CA: University of California, School of Information and Computer Science*, 2013. Pieejams: http://archive.ics.uci.edu/ml.
18. D.S. Liu, S.J. Fan, “A modified decision tree algorithm based on genetic algorithm for mobile user classification problem,” *The Scientific World Journal*, Feb. 2014.
19. B. Marr, “Big Data: 20 Mind-Boggling Facts Everyone Must Read,” *Forbes*, 30 Sep. 2015. Pieejams: https://www.forbes.com/sites/bernardmarr/2015/09/30/big-data-20-mind-boggling-facts-everyone-must-read/#54ac28f717b1.
20. M. Mitchell, *An Introduction to Genetic Algorithms*. Cambridge, Massachusetts, United States of America: A Bradford Book The MIT Press, 1999.
21. A.P. Pawlovsky, M. Hiroki, “A kNN method for breast cancer prognosis that uses a genetic algorithm for component selection,” *Toin University of Yokohama Repository*, 2017. Pieejams: https://toin.repo.nii.ac.jp/?action=repository_action_common_download&item_id=216&item_no=1&attribute_id=22&file_no=1.
22. A. Rammal u.c., “Selection of discriminant mid-infrared wavenumbers by combining a naïve Bayesian classifier and a genetic algorithm: Application to the evaluation of lignocellulosic biomass biodegradation,” *Mathematical Biosciences*, vol. 289, 2017, pp. 153-161.
23. P. Sangani, “Global data to increase 10x by 2025: Data Age 2025,” *The Economic Times*, 04 Apr. 2017. Pieejams: https://economictimes.indiatimes.com/tech/internet/global-data-to-increase-10x-by-2025-data-age-2025/articleshow/58004862.cms.
24. P. Sharma, “Discovery of Classification Rules Using Distributed Genetic Algorithm,” *Procedia Computer Science*, vol. 46, 2015, pp. 276-284.
25. C.N. Silla Jr., A.A. Freitas, “A survey of hierarchical classification across different application domains,” *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, 2011, pp. 31-72.

26. Y. Zelenkov, E. Fedorova, D. Chekrizov, “Two-step classification method based on genetic algorithm for bankruptcy forecasting,” *Expert Systems with Applications*, vol. 88, 2017, pp. 393-401.

PIELIKUMI

1. pielikums

Autora izstrādātais kods un tā darbināšanai nepieciešamais

Pirmkods, izpildāmā datne un izmantotie dati ir pieejami sekojošajā saitē: https://www.dropbox.com/s/5syc7mk8txpgf5f/EdmundsOzolinsKods.zip?dl=0. Lai minēto atrastu ir nepieciešams atvērt attiecīgo saiti. Iespējams, ka vispirms tiks izteikts piedāvājums reģistrēties, bet tas izstrādātā lejupielādei nav nepieciešams. Tātad attiecīgo piedāvājumu var aizvērt. Tālāk augšējajā labajā stūrī ir jāatrod poga "Download", kas ļauj lejupielādēt arhīva datni. Attiecīgi ir jāveic datnes lejupielāde un atarhivēšana.

Pirmkodu var apskatīt mapē "Pirmkods". Citādi tālāk ir aprakstīts, kā izmēģināt autora izstrādāto. Izpildei ir nepieciešama atjaunināta Java.

Vispirms nepieciešams atvērt datni "GAClassifier.jar" (ja nepieciešams, tad risinājuma palaišanai no komandrindas jālieto komanda "java -jar "GAClassifier.jar""), kas atver saskarnes logu.

Augšējā loga daļā var izvēlēties izmantojamos datus un kur saglabāt rezultātu. Šim nolūkam ir izmantojamas pieejamās ".arff" datnes. Autors norāda, ka ja izmanto pieejamās ".arff" datnes un ja tās atrodas vienā mapē, tad pēc trenēšanas datnes izvēles testēšanas datne tiks izvēlēta automātiski. Rezultāts tiek parādīts loga apakšējajā daļā. Papildus tam var izvēlēties arī vietu, kur saglabāt rezultātu datni.

Loga vidējajā daļā var izvēlēties izmantojamos parametrus. Parametru skaidrojumi ir pieejami caur paskaidrēm, ar kursoru uzejot uz attiecīgā parametra nosaukuma. Informāciju par parametriem var apskatīt arī maģistra darba 4.1., 5.1., 5.3. un 6.1. tabulās.

Loga apakšējajā daļā ir vairākas pogas, kuras piedāvā palaist dažādus izpildes variantus. Šie varianti secīgi, sākot no kreisās puses ir: autora atribūtu selekcijas risinājums (4. nodaļa), autora klasifikācijas risinājums (5. nodaļa), autora ansambļa klasifikācijas risinājums (6. nodaļa) un nejauši izvēlēto parametru testi (5. un 6. nodaļas risinājumiem).

Autora izstrādātās krustmijas jaunu indivīdu ražošanai no labāko indivīdu grupas Java funkcija

```
//creates a new set of members based on an existing best group of members
//calculates value probabilities and weights from the existing best group of members
public void multipleCrossover(HashSet<Individual> nextPopulation,
                              Individual[] curPopulation, int placeInPop, double percentageFrom,
                              double percentageTo) throws CloneNotSupportedException{

   SelectionIndividual tmpIndividual;   //individual to create
   double[] probabilityChromosome;    //value probabilities as seen in the best group
   double[] minWeights;       //weights are randomized between min and max observed values
   double[] maxWeights;
   int sizeOfChromosome;

   sizeOfChromosome = curPopulation[0].chromosome.length;
   probabilityChromosome = new double[sizeOfChromosome];
   minWeights = new double[sizeOfChromosome];
   maxWeights = new double[sizeOfChromosome];

   for(int c = 0; c < sizeOfChromosome; c++){
      probabilityChromosome[c]=0;
      minWeights[c] = -1.0;
      maxWeights[c] = -1.0;
   }
```

```
//get the probabilities of having true from the best group
//first get the count and then divide by all to get the probability
//for weights register the min/max values and then use them as ranges
for(int c = placeInPop; c < placeInPop+(int)Math.round(percentageFrom*
                                                    GlobalVar.populationSize); c++){

   for(int m = 0; m < sizeOfChromosome; m++){
      //probabilities
      if((Boolean)curPopulation[c].chromosome[m] == true){
         probabilityChromosome[m]++;
      }
      //weights
      if(minWeights[m] == -1.0){
         //initialize if none are present (-1)
         minWeights[m] = ((SelectionIndividual)curPopulation[c]).weights[m];
         maxWeights[m] = ((SelectionIndividual)curPopulation[c]).weights[m];
      } else {
         //extend one way or the other if possible
         if(((SelectionIndividual)curPopulation[c]).weights[m] > maxWeights[m]){
            maxWeights[m] = ((SelectionIndividual)curPopulation[c]).weights[m];
         }
         if(((SelectionIndividual)curPopulation[c]).weights[m] < minWeights[m]){
            minWeights[m] = ((SelectionIndividual)curPopulation[c]).weights[m];
         }
      }
   }

}

//get the probabilities for chromosome values
for(int c = 0; c < sizeOfChromosome; c++){
   probabilityChromosome[c] /= Math.round(percentageFrom*GlobalVar.populationSize);
}
```

```
    //create the required amount of members from the probabilities and ranges
    for(int i = 0; i < (int)Math.round(percentageTo*GlobalVar.populationSize); i++){

        //create a new individual
        tmpIndividual = new SelectionIndividual(sizeOfChromosome, false, useWeights);
        for(int c = 0; c < sizeOfChromosome; c++){
            //roll a number and decide on the chromosome value
            tmpIndividual.chromosome[c] = GlobalVar.randomGenerator.nextDouble() <
                                                                probabilityChromosome[c];
            //weights are randomized between min and max observed
             ((SelectionIndividual)tmpIndividual).weights[c] = minWeights[c] + (maxWeights[c] -
                                    minWeights[c]) * GlobalVar.randomGenerator.nextDouble();
        }

        //add it or use mutation if it already exists
        if(nextPopulation.contains(tmpIndividual) == false){
            nextPopulation.add(tmpIndividual.clone());
        } else {
            diversificationMutation(nextPopulation, tmpIndividual);
        }

    }

}
```

Autora izstrādātās mutācijas indivīdu dažādības palielināšanai Java funkcija

```
//mutation changes values till an unique individual is created
public void diversificationMutation(HashSet<Individual> nextPopulation,
                    Individual tmpIndividual) throws CloneNotSupportedException{

   boolean uniqueFound;   //whether an unique value has been found
   int placeToFlip;           //place where to flip the value
   int sizeOfChromosome;

   sizeOfChromosome = tmpIndividual.chromosome.length;
   uniqueFound = false;

   //uniqueness is determined by using the hash set
   while(uniqueFound == false){

      placeToFlip = GlobalVar.randomGenerator.nextInt(sizeOfChromosome);

      //flips only chromosome or changes both (chromosome + weights)
      if(useWeights == false || GlobalVar.randomGenerator.nextInt(2)<1){

         tmpIndividual.chromosome[placeToFlip] =
                                    !(Boolean)tmpIndividual.chromosome[placeToFlip];

      } else {
```

```
            //for the weight - first decide on addition or substraction
            //and then change by some random value
            if(GlobalVar.randomGenerator.nextInt(2)<1){
                ((SelectionIndividual)tmpIndividual).weights[placeToFlip] +=
                                                GlobalVar.randomGenerator.nextDouble();
            } else {
                ((SelectionIndividual)tmpIndividual).weights[placeToFlip] -=
                                                GlobalVar.randomGenerator.nextDouble();
            }

            //but don't allow negative weights
            if(((SelectionIndividual)tmpIndividual).weights[placeToFlip] < 0)
                        ((SelectionIndividual)tmpIndividual).weights[placeToFlip] = 0.01;

        }

        //add if unique else try again
        if(nextPopulation.contains(tmpIndividual) == false){
            nextPopulation.add(tmpIndividual.clone());
            uniqueFound = true;
        }

    }

}
```

Autora izstrādātās indivīdu derīguma noteikšanas lielākam datu ierakstu pārklājumam Java funkcija

```
//if this is used then fitness is set after evaluation
//those individuals who classify instances that other individuals don't
//are rewarded better
//uses previous information (from evaluation) about voting results
@Override
public void setFitness(Individual[] curPopulation){

   double correctVotes;          //TT classifications
   double incorrectVotes;        //TF classifications
   double correctPercentage;     //based on the previous two variables (votes)

   //turn correctlyVotedFor into probabilities
   for(int i = 0; i < correctlyVotedFor.length; i++){
      correctlyVotedFor[i] /= (double)GlobalVar.populationSize;
   }

   //sets fitness for all individuals of the current population
   for(int i = 0; i < curPopulation.length; i++){

      //don't do anything if fitness is already predetermined as "-max"
      if(curPopulation[i].fitness >= 0.0){

         correctVotes = 0.0;
         incorrectVotes = 0.0;
```

```
            for(int k = 0; k < correctlyVotedFor.length; k++){
                //add fitness (uses correctlyVotedFor)
                if(curPopulation[i].correctVotes[k] == true){
                    curPopulation[i].fitness += (double)((double)1.0-(double)correctlyVotedFor[k]);
                    correctVotes++;
                }
                //register incorrect votes
                if(curPopulation[i].incorrectVotes[k] == true){
                    incorrectVotes++;
                }
            }

            //finally modify the result with correctPercentage and save the achieved correctness
            //penalizes high amount of incorrect votes
            //one correct classification is guaranteed due to -MAX assignments
            correctPercentage = (correctVotes)/(correctVotes+incorrectVotes);
             ((ClassificationIndividual)curPopulation[i]).correctness = correctPercentage;
            curPopulation[i].fitness *= correctPercentage;

        }

    }

    for(int i = 0; i < correctlyVotedFor.length; i++){
        correctlyVotedFor[i] = 0.0;
    }

}
```

Maģistra darbs "**Ģenētisko algoritmu izmantošana klasifikācijā**" izstrādāts LU Datorikas fakultātē.

Darba teksta galīgā versija izgatavota **18.05.2018.**

Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: _____

(Autora paraksts un datums)

Ar savu parakstu apliecinu, ka esmu lasījis augstāk minēto maģistra darbu un atzīstu to par **p i e m ē r o t u / n e p i e m ē r o t u** (nevajadzīgo svītrot) aizstāvēšanai Latvijas Universitātes datorzinātņu maģistrantūrā.

Darba vadītājs: _____

(Vadītāja paraksts un datums)

Darbs iesniegts **maģistratūras sekretariātā** _____ .

(Iesniegšanas datums)

Ar šo es apliecinu, ka darba elektroniskā versija ir augšupielādēta LU informatīvajā sistēmā.

Studiju metodiķe: _____ .

(Metodiķes paraksts)

Recenzents: profesors Dr. dat. Kārlis Podnieks

(Akad.amats, zin.grāds, vārds, uzvārds)

Darbs aizstāvēts maģistra gala pārbaudījuma komisijas sēdē

_____prot. Nr. _____

(Darba aizstāvēšanas datums)

Komisijas sekretārs: _____

(Sekretāra paraksts)