УДК 004.8:004.89:519.7
DOI: 10.25140/2411-5363-2020-4(22)-72-90

Ігор Повхан

ОСОБЛИВОСТІ ПРОГРАМНИХ РІШЕНЬ МОДЕЛЕЙ ЛОГІЧНИХ ДЕРЕВ КЛАСИФІКАЦІЇ НА ОСНОВІ СЕЛЕКЦІЇ НАБОРІВ ЕЛЕМЕНТАРНИХ ОЗНАК

Актуальність теми дослідження. *Нині існує декілька незалежних, загальних підходів (концепцій) для вирішення різнотипних завдань класифікації в класичній постановці, розроблено набір різних концепцій, підходів, методів, моделей та схем, інструментарію, які охоплюють загальну проблему теорії штучного інтелекту та інформаційних систем. Причому всі ці підходи в теорії розпізнавання мають свої фіксовані переваги і недоліки та утворюють єдиний інструментарій розв'язку прикладних задач теорії штучного інтелекту. Важливою проблемою залишається питання ефективності програмних схем та алгоритмів синтезу конструкцій дерев класифікації в розрізі ефективності критерію розгалуження їхньої структури. Отже, центральну увагу в цьому дослідженні буде приділено актуальній концепції дерев рішень (дерев класифікації), розглядається загальна задача програмної (алгоритмічної) побудови логічних дерев розпізнавання (класифікації). Об'єктом цього дослідження є логічні дерева класифікації (структури ЛДК) їхні сучасні програмні реалізації. Предметом дослідження є актуальні методи та алгоритмічні схеми побудови логічних дерев класифікації.*

Постановка проблеми. *Основні наявні методи та алгоритми роботи з масивами дискретної інформації при побудові функції розпізнавання (класифікаторів) не дозволяють досягнути наперед заданого рівня точності (ефективності) системи класифікації та регулювати їх складність у процесі побудови. Однак цей недолік відсутній у методах та схемах побудови систем розпізнавання, які базуються на концепції логічних дерев класифікації (дерев рішень). Структура логічного дерева являє собою зв'язану множину гілок та вузлів, причому на гілках дерева розташовуються деякі мітки (атрибути, значення ознак), від яких залежить цільова функція (у випадку ЛДК – функція розпізнавання), а у вузлах (вершинах) знаходяться значення функції розпізнавання або розширені атрибути переходів. Тобто схема покриття навчальної вибірки набором елементарних ознак у випадку ЛДК породжує фіксовану деревоподібну структуру даних (модель ЛДК), яка забезпечує стискання та перетворення початкових даних НВ, а отже, дає змогу суттєво оптимізувати та зекономити апаратні ресурси системи, до того ж в основі лежить єдина методологія – оптимальної апроксимації навчальної вибірки набором елементарних ознак (атрибутів), які входять у деяку схему (оператор), побудовану в процесі навчання.*

Аналіз останніх досліджень і публікацій. *У представленому дослідженні були розглянуті останні публікації у відкритому доступі, які присвячені загальній тематиці підходів, методів, алгоритмів та схем розпізнавання (моделей дерев класифікації – структур ЛДК/АДК) дискретних об'єктів, відповідним програмним реалізаціям дерев рішень у задачах розпізнавання образів теорії штучного інтелекту.*

Виділення недосліджених частин загальної проблеми. *Можливість ефективної та економної програмної (алгоритмічної) схеми побудови логічного дерева класифікації (моделі структури ЛДК) на основі початкових масивів навчальних вибірок (масивів дискретної інформації) великого об'єму.*

Постановка завдання. *Розробка простого та якісного програмного методу (алгоритму та програмної системи) побудови моделей (структур) ЛДК для великих масивів початкових вибірок шляхом синтезу мінімальних форм дерев класифікації та розпізнавання, які забезпечують ефективну апроксимацію навчальної інформації набором ранжованих елементарних ознак (атрибутів) на основі схеми розгалуженого вибору ознак у широкому спектрі прикладних задач.*

Виклад основного матеріалу. *Пропонується загальна програмна схема побудови структур логічних дерев класифікації, яка для заданої початкової навчальної вибірки будує деревоподібну структуру (модель класифікації), яка складається з набору елементарних ознак, оцінених на кожному кроці побудови моделі за даною вибіркою. Запропоновано метод та готова програмна система побудови логічних дерев, основна ідея якого полягає в апроксимації навчальної вибірки довільного об'єму набором елементарних ознак. Цей метод при формуванні поточної вершини логічного дерева (вузла) забезпечує виділення найбільш інформативних (якісних) елементарних ознак із початкового набору. Такий підхід при побудові результуючого дерева класифікації дозволяє значно скоротити розмір та складність дерева (загальну кількість гілок та ярусів структури) підвищити якість його наступного аналізу.*

Висновки відповідно до статті*. Розроблене та запропоноване в роботі програмне забезпечення побудови структур ЛДК (моделей дерев класифікації) дозволяє використовувати його для розв'язання широкого спектра практичних задач розпізнавання та класифікації, а перспективи подальших досліджень можуть полягати у створенні обмеженого методу логічного дерева класифікації (структур ЛДК), який полягає у введенні критерію зупинки процедури побудови логічного дерева за глибиною структури, оптимізації його програмних реалізацій, а також експериментальних дослідженнях цього методу на більш широке коло практичних задач.*

Ключові слова*: задачі розпізнавання; дерева класифікації; логічне дерево; схема розпізнавання; алгоритм; дискретний об'єкт; елементарна ознака; розгалужений вибір ознак.*

Рис.: 4. Табл.: 4. Бібл.: 31.

Актуальність теми дослідження. Аналізуючи загальну проблему методів, алгоритмів та схем деревоподібних моделей класифікації та розпізнавання (структур логічних та алгоритмічних дерев – ЛДК/АДК) можна зафіксувати брак поточних досліджень у цьому напрямі – теорії дерев рішень, зважаючи на зміщення головної уваги в бік концепції нейромережевого розпізнавання [1]. Нині немає універсального підходу до їх розв'язання, запропоновано декілька досить загальних теорій та підходів, що дозволяють вирішувати

багато класів задач, але їх прикладні застосування відрізняються досить великою чутливістю до специфіки самої задачі або предметної області застосування. Зрозуміло, що значною мірою це пояснюється особливостями самих методів, алгоритмів та моделей (структур) дерев класифікації (ЛДК/АДК), значними труднощами реалізаційних моментів концепції алгоритмічного дерева класифікації (найвищого рівня абстракції концепції ЛДК), набором жорстких правил та обмежень щодо практичної роботи з такими структурами даних, проблемами інструментарію розробки та представлення [2-4]. Однак треба зауважити, що в більшості випадків прикладне застосування концепції дерев рішень, представлення навчальних вибірок (масивів дискретної інформації) великого об'єму у вигляді структур (конструкцій) логічних або алгоритмічних дерев (дерев узагальнених ознак) має свої суттєві переваги щодо простого та економічного опису даних, ефективних механізмів представлення та роботи з ними [5]. Важливим сегментом області практичних застосувань концепції логічних дерев залишаються методи дерев рішень (дерева класифікації, регресійні дерева), які активно використовуються як для задач теорії штучного інтелекту, засобом підтримки прийняття рішень, так і в суміжних практичних галузях економіки, управління тощо [6-12]. Нейромережева концепція розпізнавання – незважаючи на значні переваги, має, однак, істотні недоліки, які обмежують галузь її застосування. Нейронні мережі дають змогу знаходити відповідні субоптимальні розв'язки, що є проблемою в задачах із вимогою на високу точність моделі, що будується. Загальна схема функціонування зводиться до принципу чорної (сірої в окремих випадках) скрині, що недопустимо за умови аналізу причини прийняття того чи іншого рішення (в умовах правила класифікації). Значні апаратні та часові витрати інформаційної системи на процес навчання моделі не компенсуються швидкістю фінальної класифікації. Обмеження на формат вхідних даних, яке зводиться до числової шкали для методів нейромереж, накладає додаткове принципове обмеження щодо спектра можливих прикладних задач. Тому слід визнати, що клас реальних задач, які підпадають під ці обмеження, достатньо великий [11-31]. Так, концепція дерев класифікації (дерев рішень) позбавлена значної частини наведених вище недоліків та дозволяє ефективно працювати в задачах із даними довільних шкал (де інформація задається у природній формі). На сьогодні актуальними є різні підходи до побудови класифікаторів у вигляді дерев класифікації (ЛДК), причому інтерес до методів розпізнавання, які використовують ЛДК, викликаний певними корисними властивостями, якими вони володіють. З одного боку, складність класу класифікаторів у вигляді моделей ЛДК, за визначених умов, не перевищують складності класу лінійних функцій розпізнавання (простішого з відомих). Саме загальним питанням програмної (алгоритмічної) побудови дерев класифікації, структур ЛДК (у розрізі процедури їх генерації) і буде присвячена ця робота.

Постановка проблеми. Задана початкова множина M об'єктів (сигналів) w. Додатково на множині M задане розбиття R на кінцеве число підмножин, класів Ω_i, $(i = 1, \dots m)$, $M = \bigcup_{i=1}^{m} \Omega_i$. Припустимо, що розбиття M визначено неповністю. Умовами задачі задана тільки деяка інформація I про класи Ω_i, причому елементи w задаються значеннями деяких ознак x_j, $j = 1, \dots, n$, причому цей набір однаковий для всіх об'єктів, тобто однакова розмірність об'єктів). Деяку скінчено значну функцію $f_R(w)$, яка задає розбиття R, що задана на множині об'єктів M, та дає на виході номер класу i, будемо називати функцією розпізнавання (ФР). Зауважимо, що кожний образ (клас) множини M характеризується певною спільністю деяких властивостей (атрибутів) його елементів (об'єктів), а елементи з різних образів початкового розбиття не мають цієї спільності. У межах цього дослідження загальна задача розпізнавання полягає в тому, щоб для довільного об'єкта w встановити його належність певному класу (образу) на основі схеми дерева розпізнавання. Множини Ω_i також називаються компонентами розбиття множини

M. Сукупність значень ознак x_j, визначає опис (інформацію) $I(w)$ об'єкта w. Кожна з ознак може набувати значення з різних множин допустимих значень. Опис об'єкта $I(w) = (x_1(\mathrm{w}), \dots, x_n(\mathrm{w}))$ будемо називається стандартним, якщо $x_j(w)$ набариє значення лише з множини допустимих значень. Задача класифікації зі стандартною інформацією полягає в тому, щоб для фіксованого об'єкта w та набору класів $\Omega_1, \dots, \Omega_m$ за допомогою навчальної інформації $I(\Omega_1, \dots, \Omega_m)$ та опису $I(w)$ розрахувати значення деяких предикатів $P_i(w), (w \in \Omega_i; i = 1, \dots, m)$. Задача розпізнавання образів буде зводитися до навчання системи Q обчислювати функцію $f_R(x)$. Тобто система має реагувати при подачі на вхід деякого сигналу (об'єкта) x, сигналом $f_R(x)$ (фактичним номером класу належності). Основною інформацією при навчанні системи Q є значення функції $f_R(x)$ в деяких точках n-мірного простору (розмірністю в кількість ознак об'єктів множини M). Останнє означає, що при навчанні системи Q їй подаються пари сигналів $\left(\left(x_i, f_R(x_i)\right)\right)$. На основі цієї інформації (апріорної інформації) система Q будує схему обчислення функції розпізнавання. Тобто ставиться задача дослідження та розробки таких методів та моделей розпізнавання (відповідних програмних реалізацій), які б давали можливість у процесі навчання побудувати, по можливості, просту деревоподібну схему розпізнавання (схему у вигляді ЛДК або АДК), яка забезпечує необхідну ефективність та складність системи розпізнавання Q.

Аналіз останніх досліджень і публікацій. Домінуючими підходами на сьогодні в концепції дерев рішень (дерев класифікації) є системи на основі методів CART (спрямованих на розв'язок задач класифікації та регресивного аналізу), а також системи на основі схеми C4.5 та її сучасних модифікацій (для розв'язку задач розпізнавання та класифікації) та ID3 [7]. Схема ID3 базується на використанні обмеженого ентропійного критерію – структура ЛДК будується доти, поки для кожної результуючої вершини (листа дерева) не залишаться лише об'єкти одного фіксованого класу, або доки сама процедура розгалуження в дереві, що будується, дає зменшення початкового ентропійного критерію. Схема C4.5/C5.0 ґрунтується на відомому критерії *Gain-Ratio* (нормативний ентропійний критерій), причому як критерії зупинки процедури розгалуження (побудови дерева) використовується обмеження на кількість об'єктів для результуючої вершини (листа структури ЛДК) [8]. Треба зауважити, що процедура відсікання у структурі ЛДК проводиться за схемою Error-Based Pruning, яка базується на загальній оцінці здатності узагальнення для прийняття рішення щодо видалення гілок та вершин конструкції дерева класифікації. Схема CART у своїй роботі використовує критерії Джині, причому процедура відсіканні у структурі ЛДК проводиться за схемою *Cost-Complexity Pruning*, а для випадку наявних пропусків атрибутів використовується базова схема сурогатних предикатів. До того ж базову ідею методів розгалуженого вибору ознак (вершин алгоритмів) у структурі АДК можна визначити як оптимальну апроксимацію деякої початкової НВ набором ранжованих алгоритмів класифікації (ознак, атрибутів об'єкта у випадку ЛДК), то на перший план виходить центральне питання – задача вибору ефективного критерію розгалуження (відбору вершин, атрибутів, ознак дискретних об'єктів для ЛДК та алгоритмів для АДК). Ці принципові задачі розглядаються в роботах [4; 6; 15], де порушуються питання якісної оцінки окремих дискретних ознак, їх наборів та фіксованих сполучень, що дозволяє запровадити ефективний механізм реалізації розгалуження. Відомо, що структури моделей дерев класифікації (ЛДК/АДК) характеризується компактністю, з одного боку, та нерівномірністю заповнення (розрядженістю) ярусів, з іншого боку, в порівнянні з конструкціями регулярних дерев [3; 13]. При цьому важливими питаннями залишаються питання збіжності процесу побудови дерев класифікації за методами роз-

галуженого вибору ознак та питання вибору критерію зупинки процесу синтезу логічного дерева [14]. Зауважимо, що концепції дерев класифікації не суперечить можливість як ознаку (вершин структури) дерева класифікації використовувати не тільки окремі атрибути (ознаки) об'єктів їх сполучення (ідея узагальненої ознаки, розглядалась у роботі [4]) та набори, але якщо піти далі та не розглядати як розгалуження атрибутів об'єктів (ознаки), а відбирати окремі незалежні алгоритми розпізнавання, то на виході буде отримане нова структура – АДК (вищий рівень ЛДК). Саме програмним структурам ЛДК і буде присвячена ця робота.

Виділення недосліджених частин загальної проблеми. Можливість ефективної та економної програмної (алгоритмічної) схеми побудови логічного дерева класифікації (моделі структури ЛДК) на основі початкових масивів навчальних вибірок (масивів дискретної інформації) великого об'єму потребує ґрунтовного всебічного дослідження.

Мета роботи. Метою цієї роботи є вивчення особливостей генерації та представлення правил, схем, моделей класифікації в задачах розпізнавання на основі структур дерев рішень. Розробка загальної програмної схеми методу побудови ЛДК для навчальних вибірок великого об'єму. Результатом роботи є простий програмний механізм синтезу моделей ЛДК для задач класифікації дискретних об'єктів у довільних шкалах.

Виклад основного матеріалу. Треба зауважити, що довільне ЛДК можна досить просто представити у програмному форматі у вигляді таких базових елементів-масивів:

$T[1..MN]$, $SLED[1..MN]$, $UPOD[1..MN]$, де MN – кількість вершин ЛДК;

$T[1..MN]$ – масив, що характеризує інформацію, яка знаходиться у вершинах ЛДК (перелік міток, атрибутів, вершин структури дерева);

$SLED[1..MN]$ – масив, що вказує на місце наступної вершини, яка знаходиться справа і є нащадком цієї вершини у структурі ЛДК;

$UPOD[1..MN]$ – масив, що характеризує інформацію про місце лівого нащадка цієї вершини.

Зауважимо, що тут $MN < 2N - 1$, M – загальна кількість усіх об'єктів початкової інформації $I(l)$. Відмітимо також, що, крім вказаних масивів, для побудови ЛДК у цій програмній реалізації використовуються набір таких допоміжних масивів – $PR[1..N]$, $PR1[1..N]$, $SPR[1..N]$.

Отже, для програмної побудови ЛДК необхідно $M * N$ байт оперативної пам'яті (у найпростішому випадку) для зберігання початкової інформації $I(l)$, а також $3 * MN * (6M - 1)$ байт – для зберігання масивів N, $SLED$, $UPOD$ та $3N$ байт для зберігання масивів PR, $PR1$, SRP (знову же таки – байт тільки для найпростішого випадку). На наступному етапі розглянемо наступний приклад та через нього представимо загальну схему алгоритму програмної побудови фіксованого ЛДК.

Приклад. Нехай маємо деяке ЛДК, яке має фіксовану чотириярусну структуру (рис. 1), причому наведене дерево однозначно визначається наступною інформацією базових масивів T, $SLED$ та $UPOD$ – представлених у табличній формі (табл. 1).

Таблиця 1

Інформаційний вміст масивів T, SLED та UPOD

T	7	5	1	0	3	1	2	1	0	0	4	1	0
$SLED$	0	3	0	5	0	7	0	9	0	11	0	13	0
$UPOD$	2	4	6	0	8	0	10	0	0	0	12	0	0

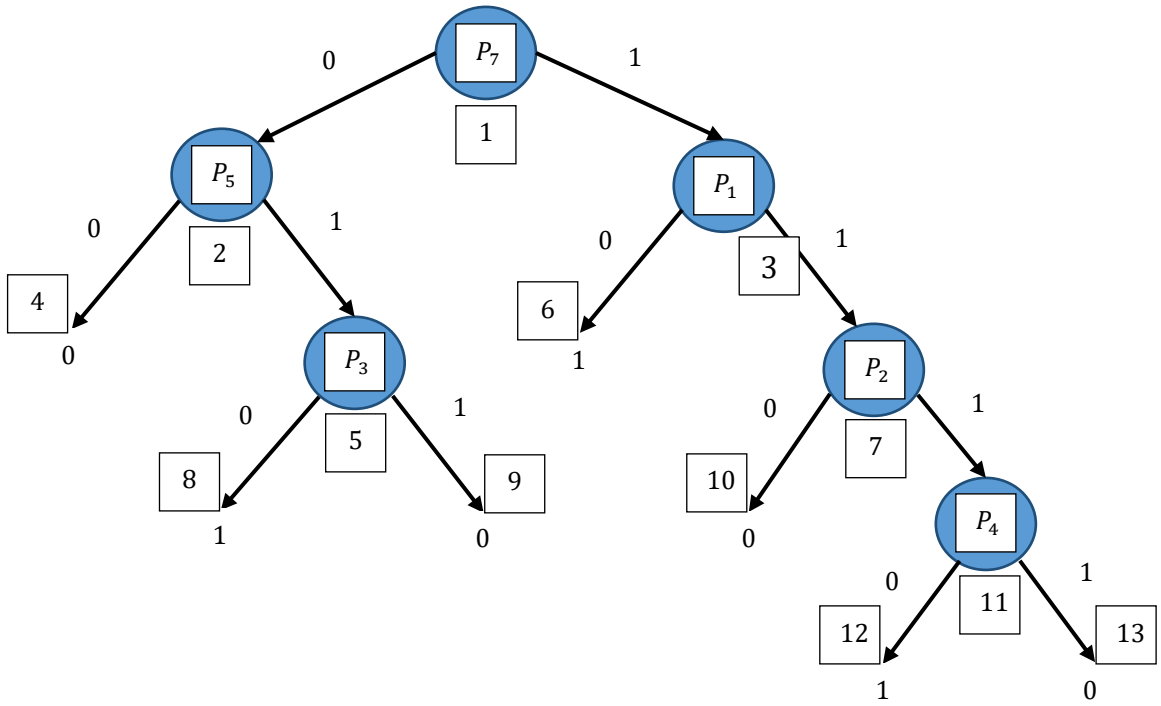


Рис. 1. Загальний вигляд початкового ЛДК

Тут слід зауважити, що типи всіх представлених масиві – це масиви байтів (зрозуміло, що це залежить від інформаційної ємності міток самого ЛДК, типізації атрибутів логічного дерева, початкових умов самої задачі, типу та об'єму початкової інформації $I(l)$).

Зрозуміло, що в цьому прикладі фактична структура отриманого ЛДК представляється за допомогою вмісту трьох основних масивів. Отже, маючи сформовані ці три масиви, можна графічно відтворити структуру побудованого програмно ЛДК.

Зважаючи на все вищенаведене, запропонуємо таку алгоритмічну реалізацію побудови ЛДК за даними початкової інформації $I(l)$.

Загальна схема алгоритму побудови ЛДК за даними початкової НВ.

Крок 1. Вибір шляху в ЛДК, що не приводить у кінцеву вершину. $UPOD = 1, PR = 0, TT = 2, KI1 = 0, SLED = 0, MMM1 = M * 2 - 1$.

Крок 2. $TT = TT + 2$. Якщо $TT = 0$, то перейти на *Крок 9*.

Крок 3. $KI = KI + 1$. Якщо $UPOD[KI] \geq 0$, то перейти на *Крок 7*. $KII = KI$. Якщо $KII = 1$, то перейти на *Крок 9*.

Крок 4. Якщо $SLED[KII] > 0$, то: $(SRP[KI1 + 1] = 0, PAR = KI)$, в іншому випадку: $(SRP[KI1 + 1] = 1, PAR = KII - 1, KII = KII - 1)$.

Крок 5. $IK2 = KII + 1$.

Крок 6. $KI2 = KI2 - 1$. Якщо $UPOD[IK2] = PAR$, то: $(KI1 = KI1 + 1, PR1[T[IK2]] = = 1, PR1[KI1] = T[IK2], KII = IK2)$. Якщо $KII = 1$, то перейти на *Крок 9*.

Якщо $KII = 0$, то перейти на *Крок 5*.

Крок 7. Якщо $KII < TT$, то перейти на *Крок 3*.

Крок 8. $IJKL = 0$.

Крок 9. Якщо $PR[1] = 0$ та $TT = 0$, то перейти на *Крок 13*.

Крок 10. (Етап обчислення значень функціоналу важливості ознак [6; 15]). Обчислити значення функціоналу для всіх P_i, що входять в область його визначення та знайти ознаку, яка має найбільшу інформативність (якість, важливість). Інформація щодо $P_{i_1}, \dots, P_{i_\xi}$ та $\eta_{i_1}, \dots, \eta_{i_\xi}$ знаходиться в масивах PR, SRP відповідно.

Крок 11. На основі *Кроку 10* заповнити відповідні значення масивів T, $SLED$, $UPOD$.
Крок 12. Перейти на *Крок 2*.
Крок 13. (Етап розпізнавання НВ за побудованим ЛДК).
Крок 14. $I = I + 1$, $II = 1$.
Крок 15. $I1 = T[II]$. Якщо $UPOD[II] = 0$, то ($PASP[I] = I1$. Перейти на *Крок 16*).
Якщо $TB[I, I1] = 0$, то ($II = UPOD[II]$. Перейти на *Крок 15*). $II = UPOD[II] + 1$.
Перейти на *Крок 14*.
Крок 16. Якщо $I < M$, то перейти на *Крок 14*.
Крок 17. Очистити всі змінні та масиви, звільнити пам‘ять.
Крок 18. Закінчити роботу алгоритму (END).

Зазначимо, що в запропонованому алгоритмі заслуговують уваги такі часові характеристики:

Загальний час побудови результуючого ЛДК залежить від об’єму початкової інформації $I(l)$, тобто при збільшенні потужності початкової інформації $I(l)$ час на побудову ЛДК є лінійною функцією від $I(l)$;

Загальний час прийняття рішень (на основі програмно побудованої структури цього ЛДК) не перевищує n простих порівнянь, де n – кількість ознак у початковій інформації $I(l)$ (фактичний час проходження за фіксованим шляхом у структурі ЛДК).

Зауважимо, що якщо в наведеному вище алгоритмі *Крок 10* замінити на виклик процедури генератора випадкових чисел (PRG) для вибору деякої ознаки, то цей алгоритм буде будувати множину випадкових дерев розпізнавання, над якими можна буде здійснювати операції замикання (питання ВДК уже розглядалось у попередніх розділах цього дослідження).

Станом на сьогодні відомо близько трьох десятків готових ПС та середовищ для побудови різних типів моделей дерев класифікації (дерев рішень) у вигляді структур ЛДК (RStudio, RulQuest, DTTL v1.5, RLQTree, DCT v7, Precision Tree System, Edraw, SHAIDS, Weka, ЛАСТАН, АУРОН та інші) та лише одна ПС яка базується на концепції АДК (ОРІОН). Усі ці системи відрізняються прикладною спрямованістю задач, що розв’язуються, методами та концептуальними засадами, різноманітним рівнем підтримки, причому багато з них знаходять у вільному (або частково вільному) доступі. Домінуючими підходами є системи на основі методів CART (спрямованих на розв’язок задач класифікації та регресивний аналіз), а також ПС на основі схеми C4.5 та її сучасних модифікацій (для розв’язку задач розпізнавання та класифікації), ID3. Зауважимо що ПС, які базуються на алгоритмах схеми C4.5/C5.0 (за авторством J. Ross Quinlan) використовують як критерій чистоти підмножин початкової НВ параметр ентропії. Використовуючи ентропію як міру чистоти (однорідності) класів (підмножин початкової НВ), які є результатом процедури розбиття, алгоритм може зафіксувати (відібрати) ту ознаку (атрибут), розбиття за якою дає найчистішу (однорідну) підмножини початково НВ (тобто підмножину початкової НВ з найменшою ентропією). Така схема в літературі позначається – *information gain* (схема підсилення інформації), причому якщо для відібраної ознаки x_i величина *information gain* є нульовою, то це фактично означає безперспективність (неможливість) розбиття НВ на підмножини – не приводить до зменшення коефіцієнту ентропії. Підкреслимо, що максимально можливе значення величини *information gain* дорівнює величині ентропії до розбиття, а це, у свою чергу, означає, що ентропія після поточного розбиття частини НВ буде дорівнювати нулю для повністю чистих (однорідних) підмножин початкової НВ [17-21].

У зв’язку з тим, що структури ЛДК після побудови за вибірками реальних даних великого об’єму мають здебільшого складну для аналізу та неоднорідну за рівнями (ярусами) структуру, то принциповою проблемою залишається питання організації процедури оптимізації або обрізки (*pruning*) таких конструкції. Під складністю структури ЛДК

розуміється загальна кількість вершин конструкції дерева (вузлів розгалуження), у такому випадку зазвичай мається на увазі, що модель ЛДК перевизначена (*is overfitted*). Важливою особливістю схеми C5.0 у плані корекції структури побудованого дерева є можливість використання механізму – *post-pruning*, коли відкидаються ті вузли, боки конструкції, піддерева, які мало (відповідно до деякого заданого критерію) впливають на результат загальної класифікації (допустима помилка), причому допускається не лише просте відсікання структур дерева, але і їх перенесення в іншу частину структури ЛДК, або заміну на іншу конструкцію з меншою структурною складністю (меншої розгалуженості). Ці схеми оптимізації (обрізки) структур ЛДК – *subtree raising* (підняття піддерева) та *subtree replacement* (заміна піддерева) у процедурі *pruning* використовуються в C5.0 та в небагатьох інших методах побудови дерев класифікації, причому абсолютна більшість інших методів та схем базується на процедурі попередньої обрізки структури ЛДК що будуються – *pre-pruning*, яка має суттєві недоліки щодо можливості пропуску (відсікання) важливих даних, які важко виявити. Отже, зважаючи на вище сказане, можна зафіксувати наступні особливості схеми C5.0 у плані побудови структур ЛДК.

Високий рівень універсальності та адаптивності дозволяє роботу з широким спектром прикладних задач різноманітних галузей практичної діяльності (обмеження щодо структури та природи початкової НВ не накладаються).

Універсальність щодо типів початкових масивів даних дозволяє працювати не тільки з дискретними вибірками, але також із масивами номінальних даних (дозволяє коректну обробку випадків пропущених даних).

Організація розгалуження у структурі ЛДК за принципом селекції елементарних ознак – причому враховуються тільки найбільш важливі (інформативні) ознаки (атрибути) дискретних об'єктів, тобто такі, які мають найбільший вплив на остаточну класифікацію.

Незалежність відносно об'єму та структури початкової НВ – дає можливість працювати з початковими НВ як щодо невеликого об'єму, так і з надвеликими масивами даних.

Висока простота та наочність інтерпретації роботи побудованої моделі (структури ЛДК), яка не вимагає спеціалізованої математичної підготовки.

Висока ефективність побудованих моделей ЛДК – навіть у порівнянні з аналогічними структурами (моделями) побудованими за класичними схемами C4.5 та CART.

Попри високу ефективність у практичній площині, наявність якісного механізму оптимізації (обрізки, *post-pruning*) побудованих структур ЛДК схема C5.0 не позбавлена й певних системних недоліків, які обов'язково потрібно враховувати як при реалізації, так і при роботі з побудованими моделями ЛДК.

Визначальною особливістю є те, що будуються дерева високої структурної складності з великою кількістю вершин, рівнів (ярусі) та високою неоднорідністю побудованої структури. Така особливість схеми C5.0 накладає високі вимоги на ефективність роботи процедури обрізки побудованої структури ЛДК та негативно впливає на інтерпретабельність – можливість доступного аналізу моделі та простого сприйняття побудованих конструкцій дерев класифікації.

Модель ЛДК, яка побудована на основі схеми C5.0 може бути як недовизначеною (*overfit*), так і перевизначеною (*underfit*).

У роботі моделі ЛДК, яка побудована на основі схеми C5.0 можливі певні неточності (помилки) класифікації у зв'язку з використанням лише прямого розбиття на підмножини (*axis-parallel split*).

Принциповою особливістю схеми C4.5/C5.0 [24-27] є її дуже висока чутливість щодо корекції у структурі та об'єму початкової НВ – причому навіть її відносно невеликі зміни можуть приводити до дуже різких змін структурної складності (радикального збільшення вершин, ярусів конструкції дерева класифікації) та ефективності процедури кінцевої оптимізації (обрізки) побудованих моделей ЛДК.

Моделі ЛДК, які побудовані на основі схеми C5.0, в абсолютній більшості випадків відрізняються великою складністю, а їх аналіз можливий лише за рахунок автоматичного або зовнішнього виділення правил класифікації конструкції ЛДК.

Звичайно, що представленими алгоритмами, ПС та фреймворками не обмежуються програмні реалізації концепція дерев рішень [28-31]. Так, в Ужгородському національному університеті на основі представленої вище в дослідженні схеми побудови ЛДК (селекції наборів елементарних ознак) була написана ПС DeTree, яка базується на концепції розгалуженого вибору ознак та дозволяє працювати з НВ великого та надвеликого об'єму (рис. 2). Причому на початковому етапі проєктування ПС ставилися такі базові вимоги щодо загального функціонала системи.

Значна увага приділялася оптимізації алгоритмів побудови моделі класифікації для досягнення максимальної швидкості – як генерації, так і роботи самої побудованої моделі ЛДК (*Runtime speed/Opertion speed*).

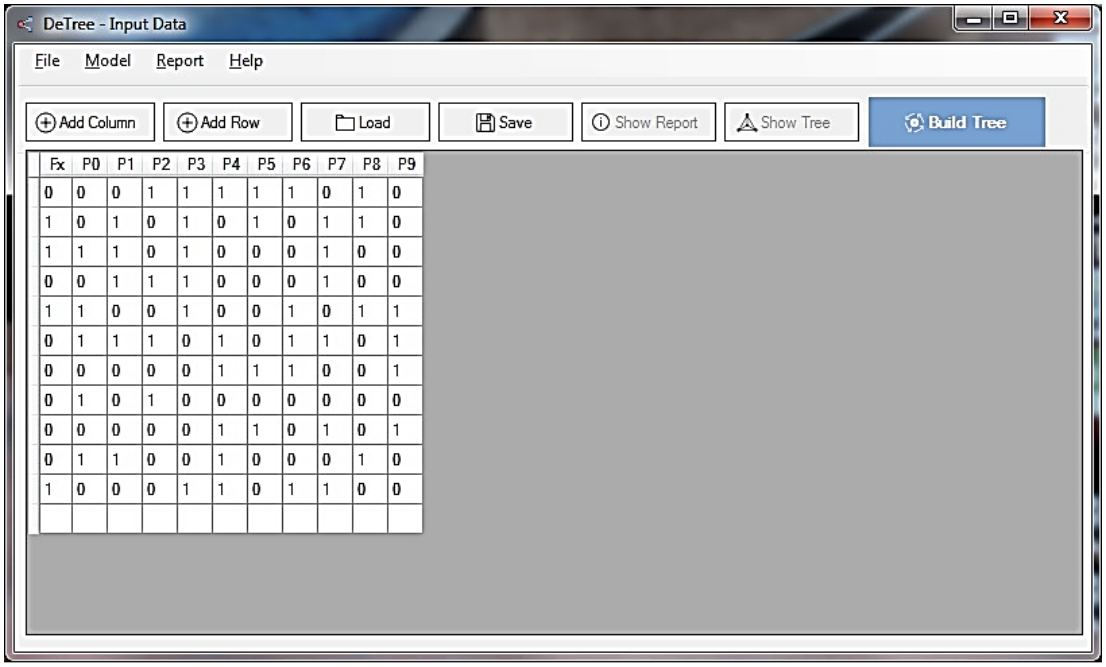
Вимога на якісну та ефективну роботи з оперативною та постійною пам'яттю інформаційної системи у зв'язку зі спрямованістю на масиви початкових даних великого об'єму.

Простий та зручний інтерфейс для оператора, обов'язкова наявність автоматичного (заснованого на різних алгоритмах) та інтерактивного режиму генерації моделей ЛДК.

Вимога простого портування готової ПС на інші апаратно/програмні платформи в перспективі.

Вимога на можливість постпроцедурної корекції та донавчання побудованої структури дерева класифікації.

Вимога на можливість роботи з НВ вибірками великого та надвеликого об'єму.



Fx	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9
0	0	0	1	1	1	1	1	0	1	0
1	0	1	0	1	0	1	0	1	1	0
1	1	1	0	1	0	0	0	1	0	0
0	0	1	1	1	0	0	0	1	0	0
1	1	0	0	1	0	0	1	0	1	1
0	1	1	1	0	1	0	1	1	0	1
0	0	0	0	0	1	1	1	0	0	1
0	1	0	1	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	1	0	1
0	1	1	0	0	1	0	0	0	1	0
1	0	0	0	1	1	0	1	1	0	0

Рис. 2. Загальний інтерфейс ПС DeTree

Відповідно до даних вимог було вирішено розділити програмний продукт на два базові проєкти (компоненти) – *DeTreeBackend* та *DeTree,* причому для зручності компонент *DeTreeBackend* представляється як бекенд, а *DeTree* – як фронтенд. У цій схемі ПС *DeTree* компонент *DeTreeBackend* забезпечує функціонал усіх базових обрахунків (усіх схем обробки даних та менеджер пам'яті) та додатково містить у собі набір класів та алгоритмів низькорівневого функціонала. Компонент *DeTree*, у свою чергу, реалізує повний функціонал роботи з боку користувача (інтерактив) та забезпечує роботу довідкової

та сервісної служби. Якщо розглядати реалізаційну схему програмного продукту з погляду патерну *MVC* (*Model-View Controller*), проект фронтенду (*DeTreeBackend*) виконує функції *View* та відповідає за базовий ввід та вивід даних, причому контролером у цій реалізаційній схемі виступає безпосередньо проєкт бекенду (*DeTree*). За основу базових потоків даних у програмному продукті фіксувалася схема вводу та виводу даних із файлів для не критичних за швидкоістю ділянок коду (для максимальної економії оперативної пам'яті). Така організація обчислень дозволяє достатньо ефективно розділяти логіку програмного забезпечення на окремі логічні підсистеми (компоненти).

Як інструментарій розробки для компонента (*DeTree*) було обрано мову програмування C++ (з додатковою можливістю портування коду), причому такий вибір значною мірою пояснюється можливістю компіляції в нативний код та можливістю роботи з низькорівневими задачами, такими як робота з оперативною пам'яттю, функціонал вводу виводу даних. Також такий вибір інструментарію розробки дозволяє забезпечити ефективну розробку основної логіки програми, а її базовий код відносно просто може бути портований на різні апаратно/програмні системи. Для компонента *DeTreeBackend* було обрано інструментарій C# з метою простоти створення графічних інтерфейсів для платформи Windows (середовище розробки *Microsoft Visual Studio*) (рис. 3).

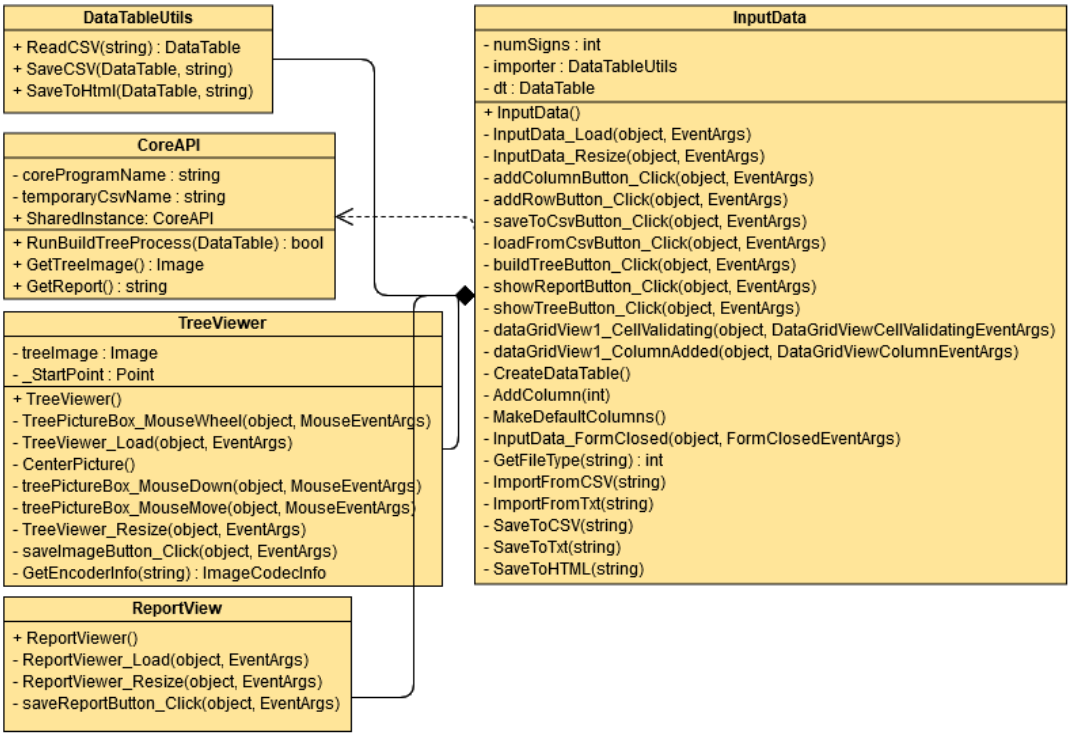


Рис. 3. UML – схема організації компонентів DeTree

Frontend Overview (DeTree). Відмітимо, що проєкт фронтенду складається з чотирьох основних вікон (*Forms*) та двох базових класів, причому форми забезпечують функції користувацького інтерфейсу та відображення різнотипної робочої інформації в процесі побудови структур ЛДК. Серед основних форм можна виділити такі:

- Форма поточного статусу роботи програми.

- Форма базового вводу виводи інформації задачі та даних інтерактивного режиму генерації ЛДК.

- Форма візуалізації згенерованої структури моделі ЛДК для автоматичного та інтерактивного режиму роботи ПС.

- Форма сервісної візуалізації проміжних процедур обрахунку даних у процесі побудови структури ЛДК.

Основним компонентом базового вводу даних обрано графічний елемент – *DataGrid*, що дозволяє досить гнучко та просто працювати з динамічними табличними даними (генерувати та корегувати таблиці різної розмірності та типу, виконувати первинні перевірки коректності вхідних даних та забезпечувати електронний підпис елементів табличних даних унікальними цифровими ідентифікаторами. Додатково у структурі компонента базового вводу даних для цієї форми доступний функціонал завантаження та зберігання табличних даних у найбільш поширених форматах даних (*cvs/cur/bin/dat/txt/html*), причому ця форма містить верхню панель інструментів відповідного завантаження/збереження даних, інструменти зміни типу/розмірності таблиці вхідних даних, інструменти візуалізації результатів обрахунків у текстовому та графічному вигляді, а також безпосередньо інструмент генерації структури ЛДК.

Зауважимо, що форма візуалізації побудованого дерева (структури ЛДК) відображає згенероване зображення з відрендереним деревом класифікації, причому доступний функціонал, який дозволяє основні функції зміни масштабу зображення та інструменти збереження його у графічній формі (основних графічних форматах).

Додатково вікно з візуалізатором процесу обрахунків проміжних даних задачі показує поточний лог розрахунку даних ЛДК на стороні компонента бекенду, причому до функціонала цієї форми можна віднести можливість збереження опису обрахунків у системний лог – файл для наступної перевірки (корекції параметрів моделі) та аналізу. Так, структура компонента фронтенду містить два базові класи – один з яких для роботи з імпортом та експортом файлів у форматі даних (CSV, HTML та ін.), інший для безпосереднього керування процесом обчислень на стороні бекенду. Робота з бекендом проводиться безпосередньо за допомогою базового класу *Process* бібліотеки *System.Diagnostics,* саме за його допомогою створюється системний процес, який запускає компонент бекенду з набором основних аргументів командного рядка.

Так, перед запуском процесу побудови дерева класифікації, компонент фронтенду формує пакет вхідних даних – цей процес проводиться за допомогою конвертування даних графічного елементу *DataGrid* з типу *DataTable* у файл формату CSV, а після конвертування даних процес побудови дерева класифікації може завантажити файл на своїй стороні та провести необхідні обчислення (рис. 4).

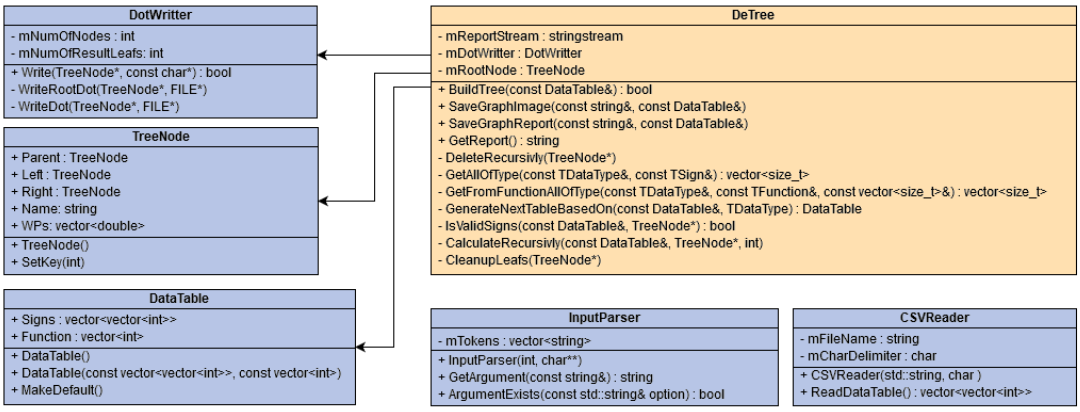


Рис. 4. UML – схема бекенду DeTree

Backend Overview (DeTreeBackend). Як уже наголошувалось вище, проєкт бекенду написаний мовою програмування C++ та являє собою класичний консольний додаток, який забезпечує базовий функціонал усіх математичних обрахунків даних (алгоритмічних схем), збір та роботу з інформацією на їх основі. Основний комунікаційний інтерфейс із нею реалізований, як варіант – через набори аргументів командного рядка, відповідно сформовані пакети даних, так і через графічний інтерфейс форми фронтенду.

На сьогодні бекенд складається з базових 14 класів та 12 структур даних. Наприклад, клас *InputParser* описує загальну логіку роботи для аналізу та розбиття наборів вхідних аргументів командного рядка на токени, цей клас забезпечує можливість швидкої перевірки та обробки вхідних параметрів командного рядка ПС, перевірки коректності їхніх значень. Наприклад, для завантаження даних ознак (атрибутів), у набори аргументів командного рядка використовується синтаксис вигляду *(-f <filename>)*, де відповідний клас парсеру перевіряє наявність аргументу вхідного файлу та дає можливість отримати безпосередньо значення шляху.

Після етапу обробки та аналізу набору вхідних параметрів та сформованих пакетів даних настає етап первинного завантаження масиву даних НВ. Основна схема інтерфейсу комунікації даних між компонентами фронтенду та бекенду полягає через посилання абсолютних шляхів файлів даних у форматі CSV (залежно від налаштувань ПС). Така схема комунікації при передачі даних між компонентами ПС пояснюється значною ефективністю, зручністю в можливості корекції даних та простотою процесу організації. Для завантаження даних у форматі CSV в компоненті бекенду реалізований відповідний клас *CSVReader,* який забезпечує повний функціонал задачі завантаження, базової валідації, корекції та формування внутрішнього представлення CSV таблиць масивів даних. Наприклад, у найпростішому випадку – дані представляються за допомогою внутрішнього псевдоніма *TDataTable,* який, у свою чергу, описується вектором векторів (*std::vector<std::vector<int>>*).

Після етапу завантаження, усіх процедур перевірки даних та їх трансформації у внутрішній формат ПС, система переходить до безпосередньої побудови дерева (структури ЛДК). На етапі проєктування ПС, усю внутрішню логіку пов'язану з розрахунками та аналізом даних було винесено в базовий клас *DeTree,* причому для побудови дерева класифікації у структурі цього класу використовується основний метод *BuildTree*. Послідовність кроків побудови структури дерева класифікації (моделі ЛДК) можна визначити таким порядком дій.

Підготовчий етап початкового вибору та ініціалізації базових параметрів та налаштувань ПС під реалії конкретної прикладної задачі відповідно до умов генерації структури (моделі) ЛДК – критеріїв розгалуження та критеріїв зупинки побудови дерева класифікації.

Початковий етап валідації наборів вхідних даних поточної задачі (перевірка на коректність різних типів даних НВ та ТВ), вибір та перевірка на коректність основних режимів (параметрів) роботи процедури генерації дерева класифікації.

Етап забезпечення первинних процедур запиту, виділення та розподілу оперативної пам'яті системи під центральну вершину (*Root Node*), вузли та переходи структури дерева класифікації (робота менеджера пам'яті ПС).

Етап роботи рекурсивної процедури обрахунку набору величин інформативності (важливості) атрибутів (ознак) $W(P_i)$, відповідно до обраного автоматично або в інтерактивному режимі критерію розгалуження структури ЛДК.

Етап генерації структури ЛДК (вузлів дерева класифікації) відповідно до зафіксованих критеріїв розгалуження та зупинки побудови дерева класифікації.

Етап роботи процедури відсікання (оптимізації структури ЛДК) для мінімізації структурних компонентів (вузлів, блоків) побудованого дерева класифікації.

Етап фінальної перевірки основних параметрів побудованої структури ЛДК (моделі класифікації) в автоматичному або інтерактивному режимі залежно від початкових налаштувань ПС.

Етап постпроцедурного аналізу побудованої моделі (структури ЛДК), етап декомпозиції синтезованого дерева класифікації з виділенням та збереженням окремих правил класифікації.

Зазначимо, що процедура рекурсивного обрахунку величин інформативності атрибутів базується на основному методі *CalculateRecursivly*, а весь функціонал вузлів конструкції ЛДК працює на основі базової структури вузла дерева (*Node*) *TreeNode*. У спрощеному випадку структура *TreeNode* складається зі списку інформаційної оцінки атрибутів $W(P_i)$, літерального імені для компонента візуалізації фрагменту ЛДК та унікального цифрового ключа, причому залежно від розташування дана структура містить набір посилань на батьківський, лівий та правий вузли (блоки) конструкції ЛДК (відповідні структури).

Функцію візуалізації побудованих структур ЛДК (генерації зображень як повних дерев класифікації, так і їхніх окремих компонентів) було покладено на формат даних *Dot*, який входить у відкритий програмний продукт *GraphViz* (бібліотека з відкритим програмним кодом), саме на її основі будується графічне представлення дерева класифікації (візуалізації складних граф – схемних представлень) і в багатьох інших подібних програмних продуктах. Для побудови файлу у форматі *Dot* було розроблено базовий клас *DotWritter,* цей клас формує текстове представлення файлу формату *Dot* та на онові рекурсивної процедури обрахунку даних – заповнює його інформацією на основі побудованого дерева класифікації.

На наступному етапі роботи ПС – після завершення всіх операції розрахунків даних, генерації всіх структур файлу формату *Dot* та формування пакетів даних, базовий клас *DeTree* зберігає всі побудовані набори даних у поточну робочу папку ПС у спеціальному найменуванні (форматі), та викликає зовнішню утиліту (компонент) програмного пакета *GraphViz* – яка на основі файлу формату *Dot* будує графічне (граф – схемне) представлення дерева класифікації (побудованої моделі класифікації). На наступному кроці – після успішного виконання всіх попередніх етапів функціонування ПС, компонент фронтенду відстежує подію коректного у плані формування пакетів даних та безпомилкового завершення процесу бекенду та переходить до процедури фінального завантаження та візуалізації всіх результатів розрахунків поточної задачі (моделі класифікації).

Звернемо увагу, що ПС DeTree має можливість збереження як графічного та параметричного представлення побудованої моделі дерева класифікації (структури ЛДК), так і допоміжних даних (проміжних етапів) синтезу конструкції дерева, що дозволяє провести ефективний аналіз самої процедури синтезу дерева класифікації (моделі) та знайти можливі варіанти його фінальної оптимізації (обрізки структури). Одним із варіантів аналізу побудованої структури ЛДК (моделі класифікації) може бути режим роботи ПС DeTree – (*step-by-step model decomposition*), який дозволяє провести поетапне виділення правил класифікації з конструкції ЛДК (шляхів у структурі дерева класифікації або фіксованих Т – опорних множин) для їх подальшого аналізу та збереження. Зрозуміло, що такий підхід буде актуальний для прикладних задач класифікації з НВ великого та надвеликого об’єму та у випадку великої складності фінальної моделі класифікації (структури ЛДК) після етапу остаточної оптимізації, мінімізації (обрізки конструкції дерева).

Для перевірки побудованого програмного забезпечення використовувалась відома задача про тип лісового покриву (ліс – 581012 елементів масиву вибірки). Структура вибірки (НВ та ТВ) містить сім класів розбиття (типи можливого лісового покрову), причому об‘єкт класифікації представляється як послідовність 12 числових ознак та додатково двох багатозначних дискретних атрибутів. Зауважимо що половина початкового масиву вибірки (а саме 290 506 об’єктів відомої класифікації) відводилась для навчання системи, а інша частина для тестування побудованих моделей ЛДК. Загальні дані про задачу та безпосередньо сам масив вибірки для перевірки можна отримати з ресурсу *UCI KDD Archive* (http://kdd.ics.uci.edu).

Крім того, алгоритм C4.5 та його ідеологічний нащадок C5.0 були скомпільовані за допомогою відкритого компілятора GCC (для системи *Linux*) з набором однакових параметрів оптимізації та компіляції бінарного коду. Як базова операційна система використовувалася *Fedora Linux 23 (RedHat)*, а для емуляції та запуску *Win32 API* (розробленої ПС DeTree) прошарок *Wine* (бібліотека *libwine*). Усі заміри часу проводились у секундах, а системи побудови структур ЛДК було запущено на двох різних апаратних конфігураціях:

Configuration № 1 – Intel Core I7 7700K / Ram 16 GB;

Configuration № 2 – AMD FX8370 / Ram 16 GB.

Основні результати тестування наведені в порівняльних таблицях 2-4.

Таблиця 2

Порівняння схем побудови ЛДК за кількістю помилок класифікації, кількістю правил класифікації, та часом побудови ЛДК

Алгоритмічна схема	**Загальна кількість помилок – Er_{All}, %**	**Загальна кількість правил класифікації – Rt_{All}**	**Загальний час генерації ЛДК – T_{All}**
C4.5	7,2	5420	Config. № 1 – 34 с. Config. № 2 – 42 с.
C5.0	6,3	4845	Config. № 1 – 186 с. Config. № 2 – 230 с.
DeTree	6,7	5028	Config. № 1 – 102 с. Config. № 2 – 129 с.

Зазначимо, що наведені алгоритмічні схеми C4.5, C5.0 та ПС DeTree можуть генерувати як класифікатори (набори правил класифікації), так і працездатні моделі (структури) ЛДК. В багатьох випадках у задачах аналізу даних набори правил класифікації є кращим варіантом з погляду експерта за рахунок простоти та наочності, ніж структури ЛДК, але, з іншого боку, таке представлення даних є повільним і вимагає значно більшої оперативної пам'яті. Зазначимо, що побудовані схемою C5.0 набори правил (модель ЛДК) мають помітно нижчу частоту помилок у порівнянні з C4.5 та DeTree, причому мають однакову точність готової моделі, але набір правил у моделі C5.0 незначно, але менше. Відносно часу генерації структур ЛДК (та правил класифікації) зафіксуємо, що алгоритм C4.5 набагато швидший за рахунок простоти схеми, достатньої оптимізації та обмежень на процедуру фінальної обрізки (*pruning*). Так, витрати оперативної пам'яті схеми C5.0 переважно значно менше, ніж у C4.5 при побудові набору правил (моделі ЛДК) – було використано 230 МБ, а для C4.5 витрачено 4.3 GB, але однаково більше, ніж у ПС DeTree – 180 Mb.

Таблиця 3

Порівняння схеми бустингу для алгоритму C5.0

Первинне ЛДК алгоритму C5.0	**ЛДК алгоритму C5.0 на основі бустингу**	**Набір первинних класифікаторів алгоритму C5.0**	**Набір класифікаторів алгоритму C5.0 на основі бустингу**
6,7 %	3,8 %	6,2 %	3,6 %

Зауважимо, що загальна схема процедури бустингу в алгоритмі C5.0 не є принципово складною, що дозволяє реалізацію подібних методик (алгоритмів та схем) навіть більшої складності (ефективності) для ПС DeTree та C4.5 у вигляді окремого програмного компонента. Проте механізм бустингу є якісним та ефективним для роботи в порівнянні з іншими подібними ПС побудови дерев рішень.

Таблиця 4

Порівняння схем побудови ЛДК за фіксованою точністю, кількістю вузлів та часом побудови структури ЛДК

Алгоритмічна схема	**Загальна кількість помилок – Er_{All}, %**	**Загальна кількість вузлів ЛДК – Vt_{All}**	**Загальний час генерації ЛДК – T_{All}**
C4.5	6,8	10167	Config. № 1 – 57 с. Config. № 2 – 43 с.
C5.0	6,8	9201	Config. № 1 – 62 с. Config. № 2 – 51 с.
DeTree	6,7	1012	Config. № 1 – 50 с. Config. № 2 – 47 с.

Принциповою особливістю схеми C5.0 у порівнянні з іншими такими алгоритмами побудови дерев класифікації є наявність механізму бустингу (*booosting*). Під бустингом будемо розуміти загальний метод генерації та заключного об'єднання декількох побудованих класифікаторів для підвищення точності класифікації. Зауважимо, що схема C5.0 підтримує прямий бустинг із будь-якою кількістю ітерацій, причому більша кількість ітерацій зазвичай призводить до подальших покращень якості синтезованих класифікаторів. Зрозуміло, що на створення гібридних класифікаторів (на основі процедури бустингу) витрачається значно більше часу, але виграш у додатковій точності моделі може виправдати додаткові витрати процесорного часу. Отже, процедуру бустингу завжди слід проводити, коли потрібна максимальна точність (якість) класифікації, особливо коли первинні класифікатори вже дають непогану точність.

Звернемо увагу, що схеми C4.5, C5.0 та DeTree синтезують моделі ЛДК з аналогічною прогнозною точністю для запропонованої задачі, але C5.0 показує трохи більший час роботи при побудові дерев класифікації при майже однаковій кількості помилок на початковій виборці. Причому основні відмінності полягають у розмірах та структурах побудованих дерев класифікації і часу обчислень при побудові готової моделі – структура дерева C5.0 помітно менше, але витрати часу C5.0 є порівняно більші.

Звернемо увагу, що схема C5.0 включає в себе кілька нових функції, таких як змінні втрат на неправильну класифікацію, що фактично дозволяє вводити додаткові типи помилок класифікації з різною вартістю (якістю та ціною). У схемі C4.5 та ПС DeTree всі помилки розглядаються як рівні за ціною (якістю, кінцевим впливом), але у прикладних задачах деякі особливі помилки класифікації (різних типів) можуть бути більш серйозними (важкими), ніж інші (звичайно залежно від специфіки задачі). Так, алгоритм C5.0 дозволяє визначити (врахувати) окрему вартість (ціну впливу) для кожної помилки (довільного типу) моделі, що будується, – якщо використовується ця опція (у схемі алгоритму), то в цьому випадку будується класифікатори для мінімізації очікуваних витрат на неправильну класифікацію, а не загальної частоти помилок структур ЛДК, причому самі помилки класифікації (випадки) також можуть мати неоднакове значення (ціну). У схемі C5.0 передбачено відповідний параметр (атрибут) ваги випадку (помилки класифікації), який кількісно та якісно визначає важливість кожного такого випадку (фактично схема C5.0 намагається мінімізувати зважену частоту помилок прогнозування).

Більшість сучасних ПС і комплексів інтелектуального аналізу даних характеризуються дуже високою універсальність у плані розмірності та структури НВ – із сотнями й тисячами атрибутів (ознак). Принциповою особливістю схеми C5.0, яка відсутня в C4.5 та DeTree (але може бути додатково реалізована окремим модулем) є автоматичне відсіювання атрибутів (ознак) перед побудовою класифікатора, які характеризуються лише незначною кореляцією (релевантністю). Для НВ великого та надвеликого об'єму така

початкова корекція масивів даних може призвести до зменшення складності класифікаторів та підвищення точності розпізнавання, а також часто може скоротити час необхідний для побудови наборів правил класифікації. Звернемо також увагу на простоту у використанні системи DeTree та C5.0. Програмний інструмент RuleQuest (з відкритим кодом) дозволяє забезпечити процедури читання та інтерпретації класифікаторів схем See5/C5.0 (See5 – відкрита програмна реалізація алгоритму C5.0 на Java). Після того як класифікатори (моделі дерев класифікації) були побудовані алгоритмами See5/C5.0 – ця система дозволяє отримати до них доступ з інших незалежних ПС.

Висновки відповідно до статті. Отже, зважаючи на все вищенаведене, можна зафіксувати такі пункти:

В одному з варіантів, загальну структуру довільного ЛДК можна досить просто представити у програмному форматі у вигляді трьох базових елементів (набору трьох масивів). Треба зауважити, що для програмної побудови ЛДК необхідно $M * N$ байт оперативної пам'яті (для простого випадку) для зберігання початкової інформації $I(l)$, а також $3 * MN * (6M - 1)$ байт – для зберігання трьох базових масивів та $3N$ байт для зберігання трьох допоміжних. Зауважимо, що час побудови результуючого ЛДК залежить від об'єму початкової інформації $I(l)$ (є лінійною функцією від $I(l)$), а час прийняття рішень за побудованим ЛДК не перевищує n простих порівнянь, де $n -$ кількість ознак у початковій інформації $I(l)$. Запропонований вище алгоритм дозволяє забезпечити ефективний механізм програмної побудови фіксованого ЛДК за набором деяких початкових даних (НВ).

На сьогодні відомо десятки ПС побудови різних типів моделей дерев класифікації (дерев рішень) у вигляді структур ЛДК та лише одна ПС, яка базується на концепції АДК, причому всі ці системи відрізняються прикладною спрямованістю задач, що розв'язуються, методами та концептуальними засадами, різноманітним рівнем підтримки, причому багато з них є у вільному (або частково вільному) доступі.

Домінуючими підходами методів та схем дерев рішень є системи на основі методів CART (спрямованих для розв'язку задач класифікації та регресивного аналізу), а ПС на основі схеми C4.5/C5.0 та їхніх сучасних модифікацій/реалізацій (для розв'язку задач розпізнавання та класифікації) та платформи (набори алгоритмів) прямого та градієнтного бустингу (бібліотеки LightGBM та XGBoost).

У зв'язку з тим, що структури дерев класифікації після побудови за вибірками реальних даних великого об'єму – мають здебільшого складну для аналізу та неоднорідну за рівнями (ярусами) структуру, то принциповою проблемою залишається питання організації процедури оптимізації або обрізки (pruning) таких конструкції. Значна ефективність того чи іншого алгоритму або схеми дерев класифікації значною мірою визначається ефективною реалізацією (ефективністю реалізованих алгоритмів) саме цього компонента.

Попри високу ефективність у практичній площині, наявність якісного механізму оптимізації, мінімізації побудованих структур дерев класифікації схема C5.0 не позбавлена і певних системних недоліків, які обов'язково потрібно враховувати як при реалізації, так і при роботі з побудованими моделями ЛДК.

ПС DeTree базується на концепції розгалуженого вибору ознак (поетапної селекції ознак) та дозволяє працювати з НВ різнотипної інформації широкого спектра прикладних задач. Причому на початковому етапі проєктування ПС ставилися базові вимоги щодо загального функціонала системи – наявність ефективного менеджера пам'яті системи, спрямованість на роботу з масивами даних великого та надвеликого об'єму, вимога на ефективність та оптимізацію коду системи, простота інтерфейсу оператора, наявність автоматичного та інтерактивного режимів роботи системи, вимога кросплатформності готової системи, вимога на корекцію та донавчання готової моделі класифікації (структури ЛДК).

Послідовність кроків побудови структури дерева класифікації (моделі ЛДК) у ПС DeTree можна визначити таким порядком дій: початковий етап визначення та ініціалізації базових параметрів, етап валідації вхідних даних та визначення режимів роботи ПС, етап роботи базових процедур менеджера пам'яті, етап формування та інформаційної оцінки наборів ознак (атрибутів), етап формування структури ЛДК (вузлів та переходів), етап оптимізації та мінімізації конструкції ЛДК (фінальної обрізки дерева класифікації), етап кінцевої перевірки параметрів побудованої моделі класифікації (ЛДК), етап аналізу та виділення правил класифікації.

Одним із варіантів аналізу побудованої структури ЛДК (моделі класифікації) може бути режим роботи ПС DeTree (step-by-step model decomposition), який дозволяє провести поетапне виділення правил класифікації з конструкції ЛДК для їх подальшого дослідження та збереження, причому зрозуміло, що такий підхід буде актуальний для прикладних задач класифікації з НВ великого та надвеликого об'єму та у випадку великої складності фінальної моделі класифікації (структури ЛДК) після етапу остаточної оптимізації, мінімізації (обрізки конструкції дерева).

Список використаних джерел

1. Повхан І. Ф. Особливості випадкових логічних дерев класифікації в задачах розпізнавання образів. *Вчені записки Таврійського національного університету. Серія: технічні науки.* 2019. Т. 30(69), № 5, С. 152–161.

2. Povhan I. Generation of elementary signs in the general scheme of the recognition system based on the logical tree. *Збірник наукових праць «Електроніка та інформаційні технології»*. 2019. Vol. 12. С. 20-29.

3. Povhan I. Question of the optimality criterion of a regular logical tree based on the concept of similarity. *Збірник наукових праць «Електроніка та інформаційні технології»*. 2020. Vol. 13. С. 19-27.

4. Повхан І. Ф. Особливості синтезу узагальнених ознак при побудові систем розпізнавання за методом логічного дерева. *Інформаційні технології та комп'ютерне моделювання ІТКМ-2019 : матеріали міжнародної науково-практичної конференції.* Івано-Франківськ, 2019. С. 169–174.

5. Kotsiantis S. B. Supervised Machine Learning: A Review of Classification Techniques. Informatica. 2007. № 31. Pp. 249–268.

6. Суботин С. А. Построение деревьев решений для случая малоинформативных признаков. *Radio Electronics, Computer Science, Control*. 2019. № 1. Pp. 121–130.

7. Deng H., Runger G., Tuv E. Bias of importance measures for multi-valued attributes and solutions. *Proceedings of the 21st International Conference on Artificial Neural Networks (ICANN).* 2011. Pp. 293–300.

8. Srikant R., Agrawal R. Mining generalized association rules. *Future Generation Computer Systems.* 1997. Vol. 13. № 2. Pp. 161–180.

9. Quinlan J.R. Induction of Decision Trees. *Machine Learning*. 1986. № 1. Pp. 81–106.

10. Miyakawa M. Criteria for selecting a variable in the construction of efficient decision trees. *IEEE Transactions on Computers.* 1989. Vol. 38, № 1. Pp. 130–141.

11. Whitley D. An overview of evolutionary algorithms: practical issues and common pitfalls. *Information and Software Technology.* 2001. Vol. 43, № 14. Pp. 817–831.

12. Vtogoff P. E. Incremental Induction of Decision Trees. *Machine Learning.* 2009. № 4. Pp. 161−186.

13. Лавер В. О., Повхан І. Ф. Алгоритми побудови логічних дерев класифікації в задачах розпізнавання образів. *Вчені записки Таврійського національного університету. Серія: технічні науки.* 2019. Т. 30(69), № 4. С. 100–106.

14. Povhan I. Designing of recognition system of discrete objects. *2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP)*, Lviv, 2016, Ukraine. Lviv, 2016. Pp. 226–231.

15. Повхан І. Ф. Проблема функціональної оцінки навчальної вибірки в задачах розпізнавання дискретних об'єктів. *Вчені записки Таврійського національного університету. Серія: технічні науки*. 2018. Т. 29(68), № 6. С. 217–222.

16. Mingers J. An empirical comparison of pruning methods for decision tree induction. *Machine learning*. 1989. Vol. 4, № 2. Pp. 227–243.

17. Subbotin S. A. Methods and characteristics of locality-preserving transformations in the problems of computational intelligence. *Radio Electronics, Computer Science, Control*. 2014. № 1. Pp. 120–128.

18. Subbotin S. A. Methods of sampling based on exhaustive and evolutionary search. *Automatic Control and Computer Sciences*. 2013. Vol. 47, № 3. Pp. 113–121.

19. De Mántaras R. L. A distance-based attribute selection measure for decision tree induction. *Machine learning.* 1991. Vol. 6, № 1. Pp. 81–92.

20. Alpaydin E. Introduction to Machine Learning. London: The MIT Press, 2010. 400 p.

21. Painsky A., Rosset S. Cross-validated variable selection in tree-based methods improves predictive performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2017. Vol. 39, № 11. Pp. 2142–2153.

22. Василенко Ю. А., Повхан І. Ф., Ващук Ф. Г. Загальна оцінка мінімізації деревоподібних логічних структур. *Eastern-European Journal of Enterprise Technologies.* 2012. Т. 2, № 4(56). С. 29-33.

23. Povhan I. General scheme for constructing the most complex logical tree of classification in pattern recognition discrete objects. *Збірник наукових праць «Електроніка та інформаційні технології».* 2019. Вип. 11. С. 112–117.

24. What is the C4.5 algorithm and how does it work (2019). URL: https://towardsdatascience.com/what-is-the-c4-5-algorithm-and-how-does-it-work-2b971a9e7db0.

25. C5.0 Classification Models (2020). URL: https://cran.r-project.org/web/packages/C50/vignettes/C5.0.html.

26. C5.0 Decision Trees and Rule-Based Models (2020). URL: https://topepo.github.io/C5.0/reference/C5.0.html.

27. C5.0 An Informal Tutorial (2020). URL: https://www.rulequest.com/see5-unix.html.

28. Subbotin S., Oliinyk A. The dimensionality reduction methods based on computational intelligence in problems of object classification and diagnosis. *Recent Advances in Systems, Control and Information Technology* / R. Szewczyk, M. Kaliczyńska (Eds.) Cham: Springer (Advances in Intelligent Systems and Computing), 2017. Vol. 543. Pp. 11–19.

29. Subbotin S. A. Random forest model building using a priori information for diagnosis. *CEUR Workshop Proceedings*. 2019. № 23. Pp. 962–973.

30. Dietterich T. G. An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Machine learning*. 2019. Vol. 40(2). Pp. 139–157.

31. Kamiński B., Jakubczyk M., Szufel P. A framework for sensitivity analysis of decision trees. *Central European Journal of Operations Research*. 2017. Vol. 26(1). Pp. 135–159.

References

1. Povkhan, I. F. (2019). Osoblyvosti vypadkovykh lohichnykh derev klasyfikatsii v zadachakh rozpiznavannia obraziv [Features random logic of the classification trees in the pattern recognition problems]. *Vcheni zapysky Tavriiskoho natsionalnoho universytetu. Seriia: tekhnichni nauky – Scientific notes of the Tauride national University. Series: technical Sciences, 30(69)*(5), pp. 152–161.

2. Povhan, I. (2019). Generation of elementary signs in the general scheme of the recognition system based on the logical tree. *Electronics and information technologies*, *12*, pp. 20–29.

3. Povhan, I. (2020). Question of the optimality criterion of a regular logical tree based on the concept of similarity. *Electronics and information technologies*, *13*, pp. 19–27.

4. Povkhan, I. F. (2019). Povkhan I. F. Osoblyvosti syntezu uzahalnenykh oznak pry pobudovi system rozpiznavannia za metodom lohichnoho dereva [Features of synthesis of generalized features in the construction of recognition systems using the logical tree method*]. Informatsiini tekhnolohii ta kompiuterne modeliuvannia ITKM-2019: materialy mizhnarodnoi naukovo-praktychnoi konferentsii – Materials of the international scientific and practical conference "Information technologies and computer modeling ITKM-2019*" (pp. 169–174).

5. Kotsiantis, S. B. (2007). Supervised Machine Learning: A Review of Classification Techniques. *Informatica*, (31), pp. 249–268.

6. Subbotin, S. A. (2019). Postroenye derevev reshenyi dlia sluchaia maloynformatyvnыkh pryznakov [Construction of decision trees for the case of low-information features]. *Radio Electronics, Computer Science, Control*, (1), pp. 121–130.

7. Deng, H., Runger, G., Tuv, E. (2011). Bias of importance measures for multi-valued attributes and solutions. *Proceedings of the 21st International Conference on Artificial Neural Networks (ICANN)* (pp. 293–300).

8. Srikant, R., Agrawal, R. (1997). Mining generalized association rules. *Future Generation Computer Systems, 13*(2), pp. 161–180.

9. Quinlan, J. R. (1986). Induction of Decision Trees. *Machine Learning,* (1), pp. 81–106.

10. Miyakawa, M. (1989). Criteria for selecting a variable in the construction of efficient decision trees. *IEEE Transactions on Computers, 38*(1), pp. 130–141.

11. Whitley, D. (2001). An overview of evolutionary algorithms: practical issues and common pitfalls. *Information and Software Technology*, *43*(14), pp. 817–831.

12. Vtogoff, P. E. (2009). Incremental Induction of Decision Trees. *Machine Learning,* (4), pp. 61−186.

13. Laver, V.O., Povkhan, I.F. (2019). Alhorytmy pobudovy lohichnykh derev klasyfikatsii v zadachakh rozpiznavannia obraziv [Algorithms for constructing logical classification trees in pattern recognition problems]. *Vcheni zapysky Tavriiskoho natsionalnoho universytetu. Seriia: tekhnichni nauky – Scientific notes of Tauride national University. Series: technical Sciences, 30(69)*(4), pp. 100–106.

14. Povhan, I. (2016). Designing of recognition system of discrete objects, *IEEE First International Conference on Data Stream Mining & Processing (DSMP)* (pp. 226–231).

15. Povkhan, I. F. (2018). Problema funktsionalnoi otsinky navchalnoi vybirky v zadachakh rozpiznavannia dyskretnykh obiektiv [The problem of functional evaluation of the training sample in the problems of recognition of discrete objects]. *Vcheni zapysky Tavriiskoho natsionalnoho universytetu. Seriia: tekhnichni nauky – Scientific notes of Taurida national University. Series: technical Sciences, 29(68)*(6), pp. 217–222.

16. Mingers, J. (1989). An empirical comparison of pruning methods for decision tree induction. *Machine learning*, *4*(2), pp. 227–243.

17. Subbotin, S. A. (2014). Methods and characteristics of locality-preserving transformations in the problems of computational intelligence. *Radio Electronics, Computer Science, Control*, (1), pp. 120–128.

18. Subbotin, S. A. (2013). Methods of sampling based on exhaustive and evolutionary search. *Automatic Control and Computer Sciences*, *47*(3), pp. 113–121.

19. De Mántaras, R. L. (1991). A distance-based attribute selection measure for decision tree induction. *Machine learning, 6*(1), pp. 81–92.

20. Alpaydin, E. (2010). *Introduction to Machine Learning*. The MIT Press.

21. Painsky, A., Rosset, S. (2017). Cross-validated variable selection in tree-based methods improves predictive performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(11), pp. 2142–2153.

22. Vasilenko, Y. A., Povkhan, I.F., Vashchuk, F.G. (2012). Zahalna otsinka minimizatsii derevopodibnykh lohichnykh struktur [General estimation of tree logical structures minimization]. *European Journal of Enterprise Technologies, 1*(4(56)), pp. 29–33.

23. Povkhan, I. (2019). General scheme for constructing the most complex logical tree of classification in pattern recognition of discrete objects. *Collection of scientific papers "electronics and information technology", 11*, pp. 112–117.

24. What is the C4.5 algorithm and how does it work (2019). https://towardsdatascience.com/what-is-the-c4-5-algorithm-and-how-does-it-work-2b971a9e7db0.

25. C5.0 Classification Models (2020). https://cran.r-project.org/web/packages/C50/vignettes/C5.0.html.

26. C5.0 Decision Trees and Rule-Based Models (2020). https://topepo.github.io/C5.0/reference/C5.0.html.

27. C5.0 An Informal Tutorial (2020). https://www.rulequest.com/see5-unix.html.

28. Subbotin, S., Oliinyk, A. (2017). The dimensionality reduction methods based on computational intelligence in problems of object classification and diagnosis. In R. Szewczyk, M. Kaliczyńska, Eds., *Recent Advances in Systems, Control and Information Technology* (vol. 543, pp. 11–19). Springer (Advances in Intelligent Systems and Computing).

29. Subbotin, S. A. (2019). Random forest model building using a priori information for diagnosis. *CEUR Workshop Proceedings*, (23), pp. 962–973.

30. Dietterich, T. G. (2019). An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. *Machine learning*, *40(2)*, pp. 139–157.

31. Kamiński, B., Jakubczyk, M., Szufel, P. (2017). A framework for sensitivity analysis of decision trees. *Central European Journal of Operations Research*, *26 (1)*, pp. 135–159.

UDC 004.8:004.89:519.7

Igor Povkhan

FEATURES OF SOFTWARE SOLUTIONS OF MODELS OF LOGICAL CLASSIFICATION TREES BASED ON SELECTION OF SETS OF ELEMENTARY FEATURES

Urgency of the research. *Currently there are several independent approaches (concepts) to solve the classification problem in the general setting, and the development of various concepts, approaches, methods, and models that cover the general issues of the theory of artificial intelligence and information systems, all of these approaches in a recognition theory have their advantages and disadvantages and form a single tool to solve applied problems of the theory of artificial intelligence. This study will focus on the current concept of decision trees (classification trees). The general problem of software (algorithmic) construction of logical recognition trees (classification) is considered. The object of this research is logical classification trees (LCT structures). The subject of the research is actual methods and algorithmic schemes for constructing logical classification trees.*

Target setting. *The main existing methods and algorithms for working with arrays of discrete information in the construction of recognition functions (classifiers) do not allow you to achieve a predetermined level of accuracy (efficiency) of the classification system and regulate their complexity in the construction process. However, this disadvantage is absent in methods and schemes for building recognition systems based on the concept of logical classification trees (decision trees). That is, the coverage of the training sample the set of elementary signs in the case of LCT generates a fixed tree data structure (model LCT), which provides compression and conversion initial data TS, and therefore allows significant optimization and savings of hardware resources of the system, and is based on a single methodology – the optimal approximation test sample set of elementary features (attributes) that are included in some schema (operator) constructed in the learning process.*

Actual scientific researches and issues analysis. *The possibility of an effective and economical software (algorithmic) scheme for constructing a logical classification tree (LCT structure model) based on the source arrays of training samples (arrays of discrete information) of a large sample.*

The research objective. *Development of a simple and high-quality software method (algorithm and software system) for building models (structures) LCT for large arrays of initial samples by synthesizing minimal forms of classification and recognition trees that provide an effective approximation of educational information with a set of ranked elementary features (attributes) is created on the basis of a scheme for branched feature selection in a wide range of applied problems.*

The statement of basic materials. *We propose a general program scheme for constructing structures of logical classification trees, which for a given initial training sample builds a tree structure (classification model), which consists of a set of elementary features evaluated at each step of building the model for this sample. A method and ready-made software system build logic trees the main idea is to approximate the initial random sampling of the volume set of elementary features. This method provides the selection of the most informative (qualitative) elementary features from the source set when forming the current vertex of the logical tree (node). This approach allows to significantly reduce the size and complexity of the tree (the total number of branches and tiers of the structure) and improve the quality of its subsequent analysis.*

Conclusions. *The developed and proposed mathematical support for constructing LCT structures (classification tree models) allows it to be used for solving a wide range of practical problems of recognition and classification, and the prospects for further research may consist in creating a limited method of logical classification tree (LCT structures), which consists in maintaining the criterion for stopping the procedure for constructing a logical tree by the depth of the structure, optimizing its software implementations, as well as experimental studies of this method for a wider range of practical problems.*

Keywords: *tasks of recognition, classification tree, logical tree recognition algorithm, discrete object, elementary basis, extensive characteristic selection.*

Fig.: 4. Table: 4. References: 31.

Повхан Ігор Федорович – кандидат технічних наук, доцент, доцент кафедри програмного забезпечення систем, ДВНЗ «Ужгородський національний університет» (вул. Заньковецької, 89Б, м. Ужгород, 88000, Україна).
Povkhan Igor – PhD in Technical Sciences, Associate Professor, Associate Professor of Department of Software, Uzhgorod National University (89Б Zankovetska Str., 88000 Uzhgorod, Ukraine).
E-mail: igor.povkhan@uzhnu.edu.ua
ORCID: http://orcid.org/0000-0002-1681-3466

Повхан І. Особливості програмних рішень моделей логічних дерев класифікації на основі селекції наборів елементарних ознак. *Технічні науки та технології.* 2020. № 4(22). С. 72-90.