УДК 004.5

О. Г. Додонов[1], О. В. Коваль[2], В. Р. Сенченко[1], В. В. Шпурик[2]
[1]Інститут проблем реєстрації інформації НАН України
вул. М. Шпака, 2, 03113 Київ, Україна
[2]Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Проспект Перемоги, 37, 03056 Київ, Україна

Автоматизована система формування сценарію аналітичної діяльності

Проведено дослідження щодо тенденцій інтелектуалізації програмних компонентів у сучасних аналітичних системах. Показано що однією з головних вимог до сучасних аналітичних систем є комфортність самого процесу спілкування з системою за рахунок їхньої інтелектуалізації, тобто здатність системи пропонувати користувачеві найбільш імовірній крок сценарію, виходячи з аналізу попередніх дій і накопичених знань. Запропоновано підхід до вирішення задачі інтелектуалізації процесу формування сценарію аналітичної діяльності, що заснований на розвитку методів машинного навчання, а саме розвитку навчання деревами класифікації і регресії (Classification and Regression Trees) з використанням комбінації метрик оцінок ефективності запропонованого варіанта — коефіцієнта Gini або розрахунку ентропії корисності інформації. На підставі запропонованого підходу виконано реалізацію алгоритму у вигляді програми мовою Python, яка дозволяє пропонувати ймовірний крок сценарію аналітичної діяльності, навчаючись на діях користувача.

***Ключові слова**: аналітична діяльність, машинне навчання, дерева рішень, коефіцієнт Gini, ентропія, орграф, мова Python.*

Вступ

Останнім часом для певного кола аналітичних процесів спостерігається стійка тенденція інтелектуалізації програмних компонентів, які реалізують ці процеси [2]. Це можна пояснити наявністю як об'єктивних, так і суб'єктивних факторів. До об'єктивних факторів відноситься зростаюча складність самих процесів аналізу даних, і, як наслідок, сценаріїв їхньої обробки. Залежно від цілей і мети аналізу, а також природи даних (структуровані, неструктуровані, слабкоструктуровані, або навіть веб-дані), для аналізу можуть залучатися різноманітні методи обробки, на-

приклад, очищення, нормалізація, перетворення форматів, кластерний аналіз, навчання, моделювання, прогнозування, оцінка якості тощо, які самі по собі потребують від аналітика як навичок їхнього застосування, так і спеціалізованих знань. Крім того, для аналізу різних типів даних мають використовуватися саме ті сценарії обробки, які відповідають як типу даних, так і цілям, і меті аналізу. Суб'єктивні фактори асоціюються як з діями людини — кроками виконання процесів аналізу складної предметної області, так і можливістю сприйняття та інтерпретації результатів аналітичного дослідження. Безумовно, як об'єктивні, так і суб'єктивні фактори впливають на ефективність методів аналізу. Тому для певного кола аналітичних систем дуже важливо полегшити процеси виконання складних сценаріїв за рахунок їхньої інтелектуалізації.

Одним із проявив інтелектуалізації є накопичення знань щодо особливостей функціонування аналітичній системи, включаючи й знання о діях аналітика. У цьому контексті сенс накопичення знань полягає в тому, щоби програмний засіб був у змозі самостійно класифікувати нові дані, що одержуються при аналізі, і пропонувати користувачеві найбільш доцільні кроки сценарію, виходячи з накопичених знань і ситуації, що склалася на попередніх кроках [2, 3]. Тобто бажано, щоб сучасний аналітичний засіб мав можливість прогнозувати найбільш імовірний наступний крок складного сценарію та пропонувати його користувачеві, виходячи з наявних у системі знань.

Постановка задачі

Сценарій аналітичної діяльності, виходячи з парадигми інтелектуалізації програмних засобів, можна розглядати як певну структуру представлення знань, яка використовується для опису послідовності пов'язаних подій — у вигляді орграфа. У цьому підході орграф фактично визначає сукупність шляхів досягнення мети в конкретній стереотипній ситуації (pattern) для заданої предметної області, представленої у вигляді семантичної мережі, наприклад, онтології [4]. Тобто, для генерування сценарію аналітичної діяльності з елементами інтелектуалізації має формуватися орієнтований ациклічний граф можливих операцій сценарію (Directed Acyclic Graph — DAG). Слід нагадати, що DAG відображає припущення про взаємозв'язок між змінними — вузлами в контексті побудованого графа, в якому відсутні орієнтовані цикли [5]. При такому підході задача побудови сценарію аналітичної діяльності фактично зосереджується на вирішенні відомої задачі пошуку найкоротшого шляху (Single Source Shortest Path) [6]. Отже, пошук найкоротшого шляху в орграфі від витоку до його стоку з урахуванням накопичених знань про стереотипні дії користувача можна розглядати як основу для інтелектуалізації процесу формування сценарію аналітичної діяльності. Враховуючи тенденції щодо інтелектуалізації програмних засобів, така постановка задачі і досі є дуже актуальною не тільки для інтелектуалізації процесу формування сценарію аналітичної діяльності, але й для вирішення певного кола завдань.

У статті пропонується підхід до вирішення задачі інтелектуалізації процесу формування сценарію аналітичної діяльності, що заснований на розвитку методів машинного навчання, а саме розвитку навчання деревами класифікації і регресії (Classification and Regression Trees — CART) із використанням комбінації метрик оцінок ефективності запропонованого варіанта.

Слід зазначити, що у веб-просторі зараз існують певні реалізацій вирішення відомої задачі пошуку найкоротшого шляху (Single Source Shortest Path). Найбільш відомим є алгоритм Graph Algorithms in Neo4j, який входить до складу Javascript library [7] та позиціонується як free and open-source javascript. Алгоритм Graph Algorithms in Neo4j обчислює найкоротший (зважений) шлях від вузла до всіх інших вузлів на графі. Цей алгоритм застосовується одночасно з алгоритмом Shortest Path і алгоритмом Dijkstra та спрямований на моделювання та прогнозування складних динамік, таких як потік ресурсів або інформації, шляхів у багатонодовій мережевій системі.

Вирішення задачі

У сучасній літературі інформаційно-аналітичні системи розглядаються як окремий клас систем, які призначені для аналітичної обробки даних, що об'єднує, аналізує та зберігає інформацію, видобуту як з баз даних організації, так і із зовнішніх джерел, забезпечує перетворення великих обсягів деталізованих даних в узагальнену інформацію, що придатна для прийняття рішень [1]. У цьому контексті аналітична діяльність направлена передусім на переробку інформації з метою забезпечити повну та достовірну оцінку стану досліджуваного об'єкта, визначення показників його основних характеристик, прогнозування ймовірних сценаріїв його зміни [3]. Типовими задачами аналітичної діяльності є наступні.

1. Аналіз цілей і формулювання завдання на проведення інформаційно-аналітичної роботи.

2. Адаптивне управління збором інформації в інтересах вирішення управлінських завдань в умовах мінливої ситуації.

3. Аналіз і оцінювання отриманої інформації у контексті цілей аналізу, виявлення сутності процесів і явищ, що досліджуються.

4. Побудова моделі предметної області досліджень, об'єкта дослідження та середовища їхнього функціонування, перевірка адекватності моделі та її корекція.

5. Планування та проведення модельних експериментів — дослідження об'єкта з використанням сценаріїв, які описують динаміку його поведінки.

6. Синтез нового знання (інтерпретація результатів, прогнозування тощо), яке є необхідним для вирішення задач управління.

7. Процес доведення результатів аналітичної роботи (нового знання) до суб'єкта управління (структури або особи, що приймає рішення).

На рис. 1 представлено структурну схему аналітичної діяльності, яка базується на вищенаведених типових процесах.

Однією із головних вимог до сучасних аналітичних систем є комфортність самого процесу спілкування з системою за рахунок їхньої інтелектуалізації. Тобто аналітик сподівається на дружній інтерфейс системи, який може пропонувати йому найбільш ймовірній крок сценарію, виходячи з аналізу попередніх дії та накопичених знань. При цьому для зменшення часу та підвищення рівня її ефективності з урахуванням зростаючої складності самих процесів аналізу даних постає задача автоматизації процесу формування сценаріїв і його інтелектуалізації, виходячи з парадигми інтелектуалізації програмних засобів. Одним із підходів вирішення цієї задачі може бути метод машинного навчання.

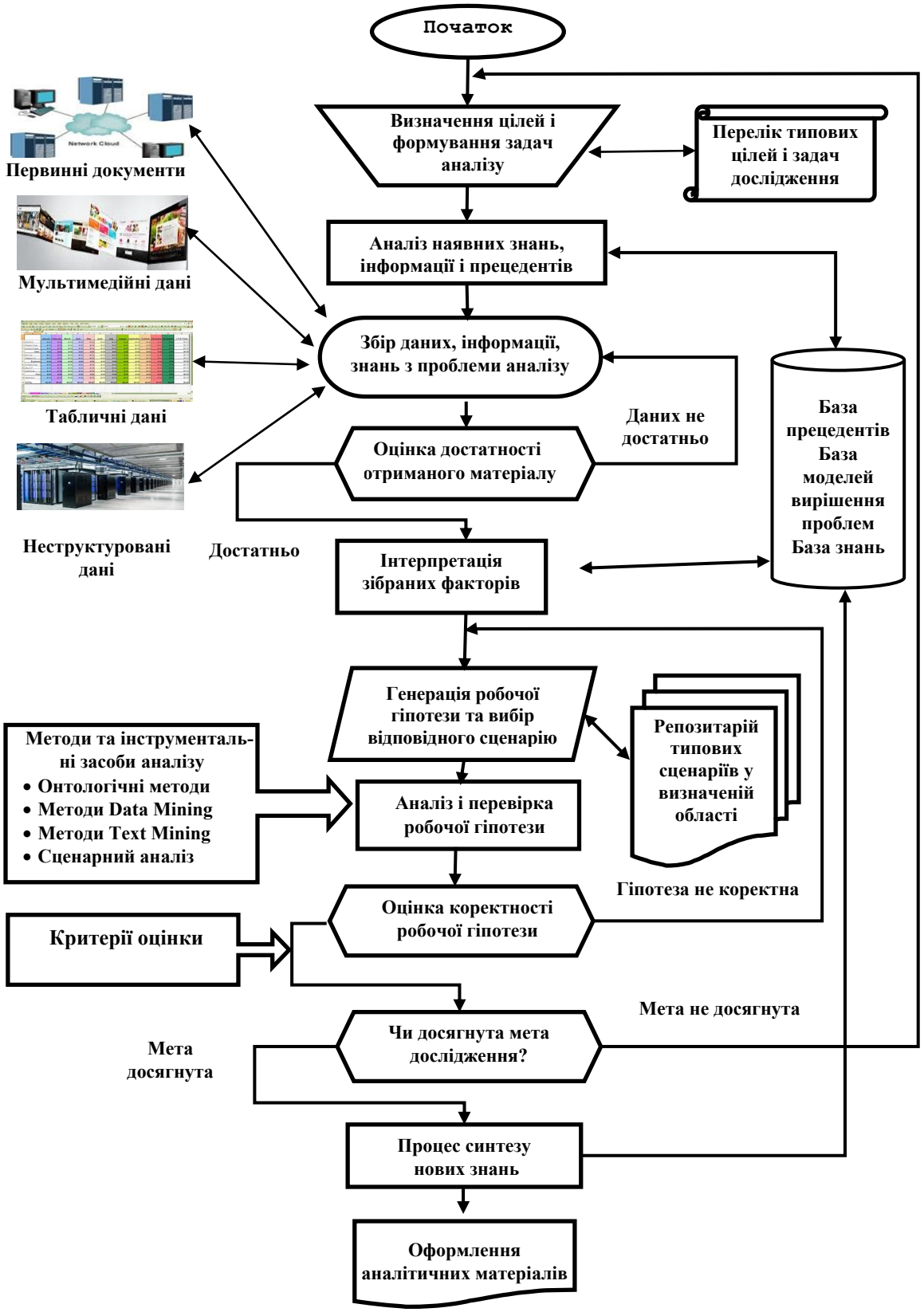

Рис. 1. Структурна схема аналітичної діяльності

Враховуючи особливості функціонування розвинених сценаріїв аналітичної діяльності [1, 3], авторами запропоновано власну версію програмної реалізації методу Classification and Regression Trees, яка відрізняється від відомого методу [7, 8] можливістю застосування різних метрик при аналізі якості розбиття та через нього вибору наступного кроку ймовірних дій аналітика, більш пристосованого для побудови розвинених сценаріїв. На відміну від існуючих підходів при виконанні машинного навчання, пропонується можливість вибору найбільш оптимальної метрики оцінки якості наближення до бажаного результату навчання — коефіцієнта Gini [11] або методу розрахунку ентропії корисності інформації [12].

Роль і місце інтелектуальних програмних засобів при виконанні сценаріїв аналітичної діяльності показано на рис. 2. Отже, це спеціалізовані програмні рішення, які містять у собі необхідні інструменти для здійснення моніторингу за діями користувачів, засоби консолідації інформації у сховище даних, витягу, перетворення, трансформації даних з метою їхньої адаптації до алгоритмів data mining (зокрема, методів машинного навчання деревами класифікації і регресії — Classification and Regression Trees), а також засоби візуалізації результатів і можливості корегування сценаріїв на підставі аналізу попередніх кроків.

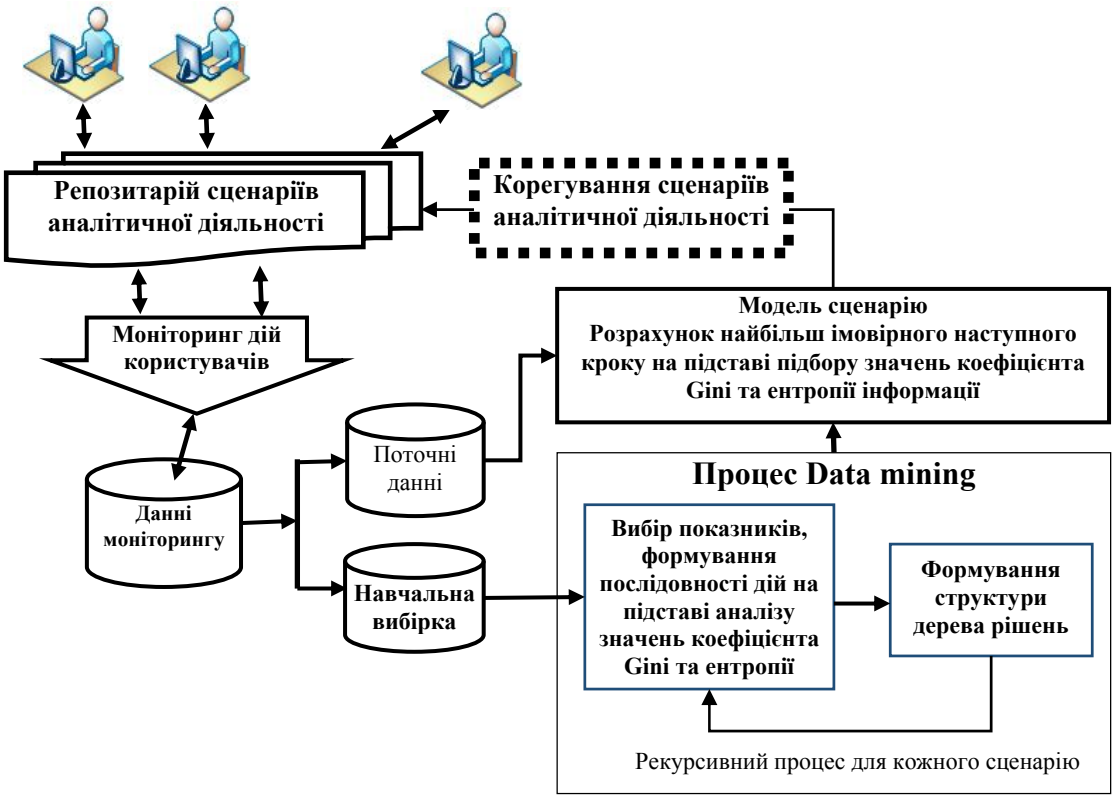

Рис. 2. Структурна схема інтелектуальних програмних засобів при виконанні сценаріїв аналітичної діяльності

Застосування комбінованого методу пояснимо наступним шляхом. На рис. 3 наведено відображення орієнтованого графа *Gr*(*Sc*), який описує гіпотетичний сценарій аналітичної діяльності з можливостями вибору різних шляхів досягнення

мети дослідження. Пунктирними колами відзначені кроки (етапи) деякого аналітичного процесу, які асоціюються з **функціями** сценарію, а кола всередині визначають деяку множину станів — можливих дій користувача або **процедур** дослідження даних, що задають конкретні параметри дослідження («A», або «B», або «C»). Як правило, ці послідовності визначають модель процесу обробки даних і можуть бути задані у вигляді типового сценарію, або отримані інтелектуальним засобом у результаті спостереження за діями користувача.

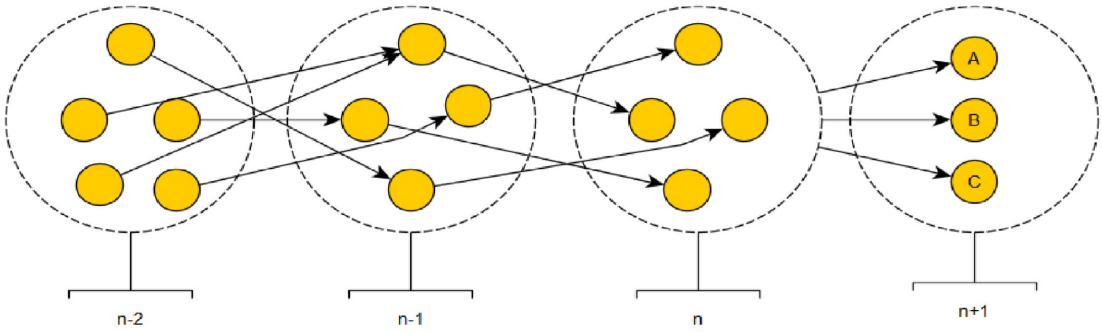

Рис. 3. Гіпотетична модель сценарію аналітичної діяльності

Завдання полягає в тому, щоб з накопичених даних (послідовності кроків при вирішенні певного кола завдань) отримати знання, на основі яких інтелектуальний засіб (алгоритм CART) у змозі запропонувати найбільш вірогідну дію користувача (процедуру обробки даних), виходячи з аналізу попередніх кроків і спираючись на знання про семантику предметної області.

Для пояснення функціонування запропонованого методу будемо розглядати тільки два попередні кроки сценарію: $n - 2$ та $n - 1$ (рис. 3). Отже, метою алгоритму CART є можливість запропонувати користувачеві (крок n), найбільш імовірний варіант наступної дії для кроку $n + 1$ (вибір процедури). Ця пропозиція формується, виходячи з аналізу дій, які користувач зробив на попередніх кроках, наприклад, послідовність дій, позначену як

$$X2 \rightarrow Y2 \rightarrow Z1 .$$

Після виконання цієї послідовності користувач має виконати дію «B» (рис. 4).

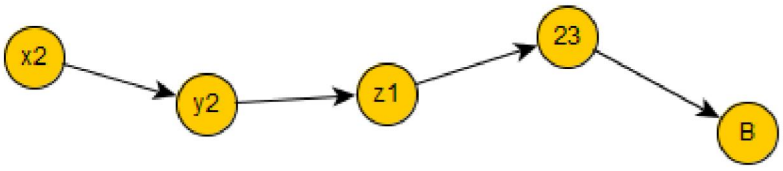

Рис. 4. Відображення варіанта певної послідовності дій

Формально аналітичний процес можна представити у вигляді вершин орієнтованого графа $Gr(Sc)$, що з'єднані ребрами (стрілками), який відображає послідовності дій користувача щодо досягнення мети дослідження. Для більшої реаліс-

тичності будемо вважати, що деякі дані можуть бути або втрачені, або спотворені. Спотворені дані зображені у вигляді порожнього кола. Крім того, одна з дій на графі позначена не символами, а цілим числом, яке розглядається як граничне значення (рис. 5).

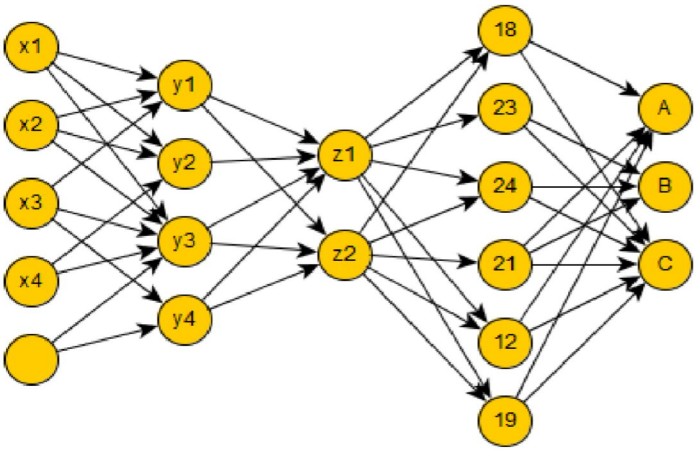

Рис. 5. Граф усіх варіантів можливих дій користувача

Першим кроком є опис матриці всіх можливих станів орієнтованого графа *Gr*(*Sc*) (включаючи і втрачені або спотворені дані (рис. 5) — загальної моделі можливих дій користувача. Для опису матриці можливих станів скористаємося об'єктно-орієнтованою мовою високого рівня Python v 3.6.4 [9]. Вибір цієї мови обумовлено необхідністю застосування різних парадигм програмування, зокрема: об'єктно-орієнтованої, процедурної, функціональної тощо, які ця мова підтримує [10].

```
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:54:40) [MSC v.1900 64
bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import cart
>>> from pprint import pprint
>>> pprint (cart.divideset(in_data, 2, 'z1'))
([['x1', 'y1', 'z1', 18, 'A'],
  ['x2', 'y2', 'z1', 23, 'B'],
  ['x3', 'y1', 'z1', 24, 'C'],
  ['x4', 'y2', 'z1', 23, 'C'],
  ['x1', 'y2', 'z1', 19, 'A'],
  ['x3', 'y4', 'z1', 12, 'C'],
  ['x2', 'y3', 'z1', 18, 'C'],
  ['x4', 'y2', 'z1', 19, 'C']],
 [['x2', 'y3', 'z2', 21, 'B'],
  ['(empty)', 'y4', 'z2', 12, 'A'],
  ['(empty)', 'y3', 'z2', 21, 'C'],
  ['x2', 'y1', 'z2', 24, 'B'],
  ['x3', 'y1', 'z2', 18, 'A'],
  ['x2', 'y3', 'z2', 18, 'A'],
  ['x4', 'y3', 'z2', 19, 'A'],
  ['x1', 'y3', 'z2', 21, 'A']])
>>>
```

Відповідно до гіпотетичного графа *Gr*(*Sc*) в останньому елементі кожного рядка матриці вказується кінцева дія користувача («A» або «B», або «C»), виконання якої потрібно спрогнозувати — передбачити.

Передбачення ймовірних дій користувача потребує, по-перше, навчання інтелектуальної системи на підставі контрольної виборці, а, по-друге, формування обґрунтованої гіпотези щодо можливості досягнення останнього елемента кожного рядку матриці станів.

Тому робота запропонованого алгоритму починається зі створення кореневого вузла графа з метою вибору найкращій змінної, виходячи з посилу, що досягнення кінцевих дій можливе лише за стовпцем (кроком *n*), який містить значення *Z*1 або *Z*2 (рис. 6).

```
26  def divideset (rows, column, value):
27      split_func=None
28      if isinstance(value,int) or isinstance(value,float):
29          split_func=lambda row:row[column]>=value
30      else:
31          split_func=lambda row:row[column]==value
32
33      set1=[row for row in rows if split_func(row)]
34      set2=[row for row in rows if not split_func(row)]
35      return (set1,set2)
```

Рис. 6. Завдання умов розбиття множини з метою вибору найкращій змінної

Для цього спочатку необхідно виконати обчислення неоднорідності вхідної множини значень, які містить матриця можливих дій. Мірою неоднорідності множини є ентропія. Для обчислення неоднорідності множини потрібно використовувати різні метрики.

Для підвищення якості розбиття скористаємося коефіцієнтом Gini (Gini coefficient) [11]. Коефіцієнт Gini — $K_{Gini}(D)$ є мірою нерівності розподілу деякої величини (D), що приймає значення між 0 і 1, де 0 означає абсолютну рівність (величина приймає лише одне значення), а 1 означає повну нерівність. $K_{Gini}(D)$ використовується у деревах рішень при виборі розподілу деякої величини (D). За допомогою цього коефіцієнта вимірюється двійковий розділ для кожного атрибута — вірогідність появи різних подій P_1 у листі дерева. Коефіцієнт Gini розраховується за формулою [11]:

$$K_{Gini}(D) = 1 - \sum_{i=1}^{n} P_i^2 .$$

На рис. 7 наведено фрагмент програмного коду розрахунку $K_{Gini}(D)$ при обчисленні приналежності до розбиття.

Методологія обчислення розподілу значень матриці можливих дій для побудови дереву рішень складається з наступних кроків.

На першому кроці обчислюється коефіцієнт Gini $K_{Gini}(D)$ для кожного значення рядка — тобто вірогідність досягнення кожного з можливих результатів приналежності до певного розбиття (підмножини). Вірогідність обчислюється шляхом ділення лічильника появ цього результату на загальне число рядків у множині.

Далі розраховується вірогідність для кожного рядка методом підсумування. Результатом цього кроку є сумарна вірогідність $\sum_{i=1}^{n} P_i^2$, яка показує, що для випад-

ково обраного рядка може бути прогнозовано не той результат, який насправді має місце. Слід нагадати, що чим вище вірогідність, тим гірше розбиття. Оскільки вірогідність 0 асоціюється з найкращим результатом, це означає,1-2 що всі рядки вже розподілені правильно.

```
45  def giniimpurity(rows):
46    total = len(rows)
47    counts = uniquecounts(rows)
48    imp = 0
49    for t1 in counts:
50      p1 = float(counts[t1]) / total
51      for t2 in counts:
52        if t1 == t2:
53          continue
54        p2 = float(counts[t2]) / total
55        imp += p1*p2
56    return imp
57
58  def entropy(rows):
59    from math import log
60    log2 = lambda x:log(x) / log(2)
61    results = uniquecounts(rows)
62    entr=0.0
63    for t in results.keys():
64        p = float(results[t]) / len(rows)
65        entr = ent - p * log2(p)
66    return entr
```

Рис. 7. Програмний код розрахунку ймовірності приналежності до розбиття різними методами

Оскільки розділення для визначеного атрибуту (наприклад, $Z1$) розпадається на дві частини $D1$ і $D2$, то $K_{Gini}(D)$ для цього розподілу обчислюється за формулою:

$$K_{Gini_{Z1}}(D) = \frac{|D1|}{|D|} K_{Gini}(D1) + \frac{|D2|}{|D|} K_{Gini}(D2).$$

Аналогічно виконується розрахунок $K_{Gini_{Z1}}(D)$ для атрибуту $Z2$.

Наступним кроком є розрахунок ентропії інформації за Шенноном — $H_{gr}(D)$ (ентропія Шеннона), яка обчислюється за формулою [12]:

$$H_{gr}(D) = \sum_{i-1}^{I} P_i \log_2 \frac{1}{P_i} = -\sum_{i-1}^{I} P_i \log_2 (P_i).$$

Для отримання оцінки того наскільки хороша обрана змінна, алгоритм спочатку обчислює ентропію всієї групи $H_{gr}(D)$, а потім намагається розбити групу за можливими значеннями кожної змінної та обчислює ентропію двох нових груп. Найкраща розбивка визначається на підставі обчислення інформаційного виграшу — різницею між поточною ентропією та середньозваженою ентропією двох нових груп, після чого обирається та, для якої інформаційний виграш максимальний. Далі алгоритм створює дві гілки: якщо умова істинна, та якщо умова помилкова. Обчислюючи для кожного вузла найкращий атрибут і розщеплюючи гілки, алгоритм будує дерево рішень.

Побудова дерева рішень полягає в обчисленні найкращого критерію розщеплення «True» або «False», в якості якого використовується середньозважена ент-

ропія, яка розраховується для кожної пари. Зростання гілки припиняється, якщо інформаційний виграш, що отриманий від розщеплення в даному вузлу, виявляється менше або дорівнює нулю (рис. 8)

Маючи на увазі той факт, що в процесі побудови дерева рішень може виникнути ситуація коли деякі дані можуть бути втрачені або спотворені, то з метою усунення такої ситуації запропоновано внесення змін до функції прогнозування наступного кроку, які враховують відсутність даних попереднього етапу, застосовуючи метод оцінки ентропії $H_{gr}(D)$.

```
def growtree(rows,scoref=entropy):
  if len(rows) == 0:
     return dnode()
  current_score=scoref(rows)
  best_gain = 0.0
  best_criteria = None
  best_sets = None
  column_count=len(rows[0])-1
  for col in range(0,column_count):
    column_values = {}
    for row in rows:
       column_values[row[col]]=1
    for value in column_values.keys():
       (set1,set2)=divideset(rows,col,value)
       p = float(len(set1)) / len(rows)
       gain = current_score - p * scoref(set1) - (1-p) * scoref(set2)
       if gain > best_gain and len(set1) > 0 and len(set2) > 0:
          best_gain = gain
          best_criteria = (col,value)
          best_sets = (set1,set2)
  if best_gain > 0:
     trueBranch = growtree(best_sets[0])
     falseBranch = growtree(best_sets[1])
     return dnode(col=best_criteria[0],
                  value=best_criteria[1],
                  tb=trueBranch,
                  fb=falseBranch)
  else:
    return dnode(results=uniquecounts(rows))
```

Рис. 8. Функція побудови дерева рішень

За результатом роботи запропонованого алгоритму формується дерево рішень, в якому можливі переходи до різних станів, позначених як «True» та «False». Для спрощення безпосередньо процесу вибору наступного кроку дій при формуванні сценарію аналітичної діяльності («True» або «False») алгоритм доповнено більш зручним механізмом формування семантичних умов переходу у вигляді послідовності операторів «if – then». Семантичною умовою переходу по дереву рішень є вибір коефіцієнта Gini $K_{Gini}(D)$ або ентропії за Шенонном — вірогідності досягнення кожного з можливих результатів приналежності до певного розбиття (рис. 9).

Для відображення дерева рішень у графічному вигляді бажано інсталювати відповідний plugin, наприклад, **graphviz** [15], який дає можливість семантичного трактування ймовірних дій користувача. Дерево рішень відображує всі ймовірні кроки дій користувача при виконанні сценарію аналітичної діяльності.

На рис. 10 наведено спрощений приклад застосування наведеного підходу для запровадження методів машинного навчання в моделюючому комплексі при формуванні типових сценаріїв обробки інформації, які пов’язані з дослідженнями поведінки гідроакустичних пристроїв у різних морських середах. Такий підхід дозволяє суттєво зменшити кількість помилкових дій користувачів (особливо нових користувачів) при формуванні складних сценаріїв із множиною різноманітних умов використання операторів аналізу даних.

```
def classify2(q, tree):
 if tree.results is not None:
   return tree.results
 else:
   v = q[tree.col]
   if v is None:
     tr, fr = classify2(q, tree.tb), classify2(q, tree.fb)
     tcount = sum(tr.values())
     fcount = sum(fr.values())
     tw = float(tcount) / (tcount+fcount)
     fw = float(fcount) / (tcount+fcount)
     result = {}
     for k,v in tr.items(): result[k] = v * tw
     for k,v in fr.items(): result[k] = v * fw
     return result
   else:
     if isinstance(v, int) or isinstance(v, float):
       branch = tree.tb if v >= tree.value else tree.fb
     else:
       branch = tree.tb if v == tree.value else tree.fb
     return classify2(q,branch)
```

Рис. 9. Формування семантичних умов переходу по дереву рішень операторами «if – then»

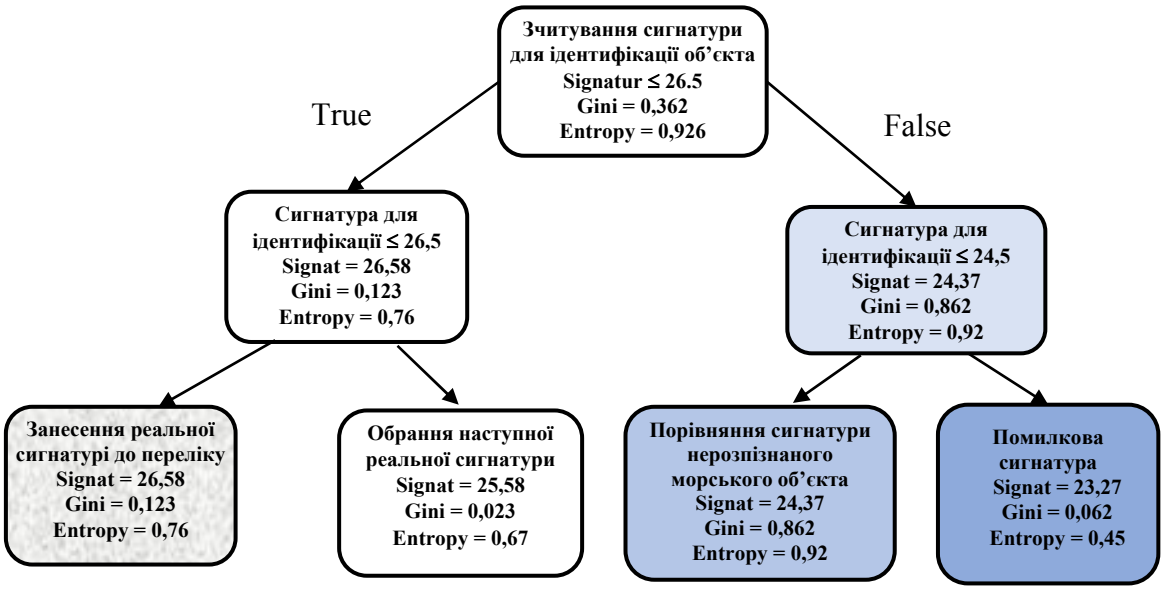

Рис. 10. Приклад формування дерева рішень для дослідження гідроакустичної області

Отже, використання алгоритму побудови дерева рішень — це прозорий і дуже наочний спосіб класифікації імовірних дій користувача при виконанні сценарію аналітичної діяльності, які після навчання видаються як послідовність пропозицій, представлених оператором «if – then» та організованих у вигляді дерева.

Висновки

Запропонований підхід до вирішення задачі інтелектуалізації процесу формування сценарію аналітичної діяльності, що заснований на розвитку методів машинного навчання, дозволяє виконувати формування дерева рішень для класифікації можливих наступних кроків, використовуючи спеціалізовані функції розрахунку коефіцієнта Gini, або розрахунку метрики ентропії інформації. Це дає мож-

ливість значно підвищити точність розбиття при обчисленні ймовірності наступного кроку в складному сценарії аналітичної діяльності.

На підставі запропонованого підходу виконано реалізацію у вигляді інтелектуального програмного засобу (написаного мовою Python). Цей програмний засіб може навчатися на діях користувача та на основі отриманих у процесі навчання знань представляє дії користувача у вигляді дерева рішень імовірного сценарію аналітичного процесу, пропонуючи користувачеві найбільш доцільні наступні кроки.

Запропоновані рішення використовуються як інтелектуальні програмні засоби в моделюючому комплексі при формуванні складних сценаріїв обробки інформації, яка надходить від гідроакустичних пристроїв, що дозволяє значно зменшити кількість помилкових дій користувачів.

1. Додонов О.Г., Коваль О.В, Глоба Л.С, Бойко Ю.Д. Комп'ютерне моделювання інформаційно-аналітичних систем: монографія/Київ: ІПРІ НАН України, 2017. 239 с.

2. Novogrudska R.L., Globa L.S., Koval O.V., Senchenko V.R. Ontology for Applications Development. Chapter 2 Ontology in Information Science. Book edited by Ciza Thomas. Print ISBN 978-953-51-3887-7. Published: March 8, 2018. P. 29–53. http://dx.doi.org/10.5772/intechopen.74042

3. Novogrudska R., Globa L., Koval O. The Method of User's Tasks Scenario Formation. The 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications 21–23 September, 2017. Bucharest, Romania

4. Сенченко В.Р., Бойченко О.А., Бойченко А.В. Дослідження методів та технологій інтеграції онтологічної моделі з реляційними даними. *Реєстрація, зберігання і оброб. даних*. 2018. Т. 20. № 3. С. 91–101.

5. Malcolm Barrett, An Introduction to Directed Acyclic Graphs. URL: https://cran.r-project.org/web/packages/ggdag/vignettes/intro-to-dags.html

6. The Single Source Shortest Path algorithm. URL: https://neo4j.com/docs/graph-algorithms/current/algorithms/single-source-shortest-path/

7. Graph Algorithms in Neo4j: Single Source Shortest Path. URL: https://dzone.com/articles/graph-algorithms-in-neo4j-single-source-shortest-p

8. Decision Tree Classification in Python. URL: https://www.datacamp.com/community/tutorials/decision-tree-classification-python

9. Python 3.6.4. URL: https://www.python.org/downloads/release/python-364//

10. Python 3.7.3 documentation. URL: https://docs.python.org/3/

11. Анализ малых данных. URL: https://dyakonov.org/2015/12/15/

12. Системный аналіз. URL: http://victor-safronov.ru/systems-analysis/books/simankov-lucenko/17.html

13. Коваль О.В., Зайцева К.А. Верифікація комп'ютерної моделі системи інформаційного управління. *Вісник НТУУ «КПІ»*. Серія: Інформатика, управління та обчислювальна техніка. 2014. № 61. С. 43–48.

14. Коваль А.В., Бойко Ю.Д., Волкова Е.А. Особенности сценарно-целевого подхода к анализу объектов действенной аналитики. *Системні дослідження та інформаційні технології*. 2015. № 1. С. 57–61.

15. Graphviz — Graph Visualization Software. URL: https://graphviz.gitlab.io/download/

Надійшла до редакції 11.03.2019