МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра комп'ютерних наук

МЕТОДИЧНІ ВКАЗІВКИ
ДО ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

з дисципліни

«ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ»

для магістрів спеціальності 121
Інженерія програмного забезпечення

Тернопіль
ЗУНУ
2022

Л.І.Гончар, Є.О.Марценюк // Методичні вказівки до виконання лабораторних робіт із дисципліни ”Інтелектуальний аналіз даних” для магістрів спеціальності 121 Інженерія програмного забезпечення

Укладачі: **Гончар Людмила Іванівна**, доцент кафедри КН ЗУНУ

Марценюк Євгенія Олексіївна, доцент кафедри КН ЗУНУ

Затверджено на засіданні кафедри комп’ютерних наук
(протокол № 2 від 14 вересня 2022 року).

Розглянуто та схвалено групою забезпечення спеціальності
121 Інженерія програмного забезпечення
(протокол № 2 від 14 вересня 2022 р.)

ЗМІСТ

ВСТУП

Засоби сучасної інформаційної технології в останній час уможливили накопичення і зберігання великих обсягів даних про бізнесові процеси. Ці дані можуть знаходитися в корпоративних базах або сховищах даних. Вони містять важливі закономірності і зв’язки між системними характеристиками, які можуть бути використані для прийняття обгрунтованих управлінських рішень. Наразі виникла проблема розробки методів відкриття таких закономірностей, про існування яких користувачі можуть і не знати. Проте традиційний аналіз даних передбачує введення даних в стандартні або настроєні користувачем моделі, тобто в будь-якому випадку допускається, що зв'язки між різними показниками добре відомі і можуть бути виражені математично. Однак, в багатьох випадках зв'язки не можуть бути апріорі відомі.

У таких ситуаціях моделювання стає неможливим і тут можна застосовувати дейтамайнінг (*Data Mining) – інтелектуальний аналіз даних (ІАД).*

Тому, особливо важливим аспектом підготовки магістрів спеціальності 121 Інженерія програмного забезпечення є успішне засвоєння ними дисципліни "Інтелектуальний аналіз даних”.

Метою викладання навчальної дисципліни «Інтелектуальний аналіз даних» є формування у студентів знань про основи інтелектуальної технології Data Mining (Data Science), моделей Data Mining, статистичних і кібернетичних методів Data Mining, інтелектуальної технології Text Mining , а також, умінь та використання доступного програмного забезпечення інтелектуального аналізу даних , методів реалізації генетичних алгоритмів та формування динамічних бізнес-процесів у мережі Інтернет тощо.

Методвказівки до виконання лабораторних робіт з дисципліни “Інтелектуальний аналіз даних” включають 5 лабораторних робіт, кожна із яких містить необхідний методичний матеріал для вивчення даного предмету.

ЛАБОРАТОРНА РОБОТА № 1

Тема: Система PolyAnalyst- універсальна система для проведення інтелектуального аналізу даних (Data mining)

Мета: Вивчити основні принципи роботи системи PolyAnalys та ознайомитись із базовими алгоритмами аналізу даних : моделювання, прогнозування, асоціації, текстовий аналіз.

1. Теоретичні відомості

Доступне програмне забезпечення дейтамайнінгу

Компанія «Мегап'ютер» виробляє і пропонує на ринку сімейство продуктів для дейтамайнінгу — *PolyAnalyst.* Система PolyAnalyst призначена для автоматичного і напівавтоматичного аналізу числових баз даних і витягання з сирих даних практично корисних знань. PolyAnalyst знаходить багатофакторні залежності між змінними в базі даних, автоматично будує і тестує багатовимірні нелінійні моделі, що виражають знайдені залежності, виводить класифікаційні правила по повчальних прикладах, знаходить в даних багатовимірні кластери, будує алгоритми рішень.PolyAnalyst використовується в більш ніж 20 країнах світу для вирішення завдань з різних областей людської діяльності: бізнесу, фінансів, науки, медицини. В даний час - це одна з наймогутніших і в той же час доступних в ціновому відношенні комерційних систем для Data mining в світі. Основу PolyAnalyst складають так звані Exploration engines або Машини досліджень - математичні модулі, засновані на різних DM алгоритмах, і призначені для автоматичного аналізу даних. Компанія Megaputer Intelligence веде інтенсивні дослідження, направлені на розширення аналітичних функцій системи PolyAnalyst, розробку нових DM алгоритмів і нових математичних модулів системи.

Архітектура системи

За своєю природою, PolyAnalyst є клієнт/серверним додатком. Користувач працює із клієнтською програмою PolyAnalyst Workplace. Математичні модулі виділені в серверну частину - PolyAnalyst Knowledge Server. Така архітектура надає природну можливість для масштабування системи: від однокористувацького варіанту до корпоративного рішення з декількома серверами. PolyAnalyst написаний на мові C++ з використанням специфікації Microsoft's COM (ACTIVEX). Ця специфікація встановлює стандарт комунікації між програмними компонентами. Математичні модулі (Exploration Engines) і багато інших компонентів PolyAnalyst виділені в окремі динамічні бібліотеки і доступні з інших додатків. Це дає можливість інтегрувати математику PolyAnalyst в ті, що існують ІС, наприклад, в CRM або ERP системи.

1.1. Система PolyAnalyst

Система PolyAnalyst призначена для автоматичного і напівавтоматичного аналізу числових баз даних і витягання з сирих даних практично корисних знань. PolyAnalyst знаходить багатофакторні залежності між змінними в базі даних, автоматично будує і тестує багатовимірні нелінійні моделі, що виражають знайдені залежності, виводить класифікаційні правила по повчальних прикладах, знаходить в даних багатовимірні кластери, будує алгоритми рішень.

PolyAnalyst використовується в більш ніж 20 країнах світу для вирішення завдань з різних областей людської діяльності: бізнесу, фінансів, науки, медицини. В даний час - це одна з наймогутніших і в той же час доступних в ціновому відношенні комерційних систем для Data mining в світі.

Основу PolyAnalyst складають так звані Exploration engines або Машини досліджень - математичні модулі, засновані на різних DM алгоритмах, і призначені для автоматичного аналізу даних. Компанія Megaputer Intelligence веде інтенсивні дослідження, направлені на розширення аналітичних функцій системи PolyAnalyst, розробку нових DM алгоритмів і нових математичних модулів системи.

Остання версія PolyAnalyst включає перелік наступних машин досліджень:

Назва модуля	Технологія/методи
Find Laws Algorithm (FL)	Symbolic Knowledge Acquisition Technology, Еволюційне програмування
Find Dependencies Algorithm (FD)	N-dimensional distribution analysis, N-мірний аналіз ррозподілів
Cluster Algorithm (FC)	Localization of Anomalies, N-мірний кластерізатор
PAY Algorithm (MB)	Memory Based Reasoning and Genetic Algorithms hybrid, гібрид методу "найближчих сусідів" та генетичних алгоритмів
Market Basket Analysis (BA)	Transactional clustering and directed association rules, транзакційний кластерізатор з генерацією направлених асоціативних правил
Linear Regression (LR)	Stepwise Linear Regression, багатопараметрична лінійна регресія із автоматичним вибором незалежних змінних
Classify Algorithm (CL)	Fuzzy logic classification, класифікация по булевій цільовій змінній, необхідно наявність модуля FL, або PN, або MB, або LR
Disciminate (DS)	Модифікація модуля CL, знаходить відмінності між двома таблицями
Decision Trees (DT)	Модуль "дерева рішень", класифікація на категорії
Decision Forest (DF)	Багатомірне «дерево рішень», класифікація на велику кількість категорій
Text Analysis (TA)	Модуль текстового аналізу, перетворює неструктурований текст в простір формальних ознак для наступного аналізу алгоритмами Data mining
Link Analysis (LA)	Модуль знаходження та графічної візуалізації зв'язків між об'єктами
Summary Statistics (SS)	Модуль загальної статистики

Компанія Мегапьютер проводить і пропонує на ринку сімейство продуктів для Data mining - PolyAnalystOn-line Tutor <http://www.megaputer.ru/content/production/pa/tutor_eng/index.html>. Система PolyAnalyst призначена для автоматичного аналізу числових і текстових даних з метою виявлення в них раніше невідомих, нетривіальних, практично корисних і доступних розумінню закономірностей, необхідних для ухвалення оптимальних рішень в бізнесі і в інших областях людської діяльності.

У даний час PolyAnalyst є однією з наймогутніших систем Data Mining в світі, реалізованих для Intel платформ і операційних систем Microsoft Windows. Аналогічні системи Data Mining таких провідних виробників, як IBM (Intelligent Miner, Data Miner), Silicon Graphics (SGI Miner), Integral Solutions (Clementine), SAS Institute (SAS) працюють на середніх і великих машинах і коштують десятки і навіть сотні тисяч доларів. Завдяки унікальній технології "Еволюційного програмування", і іншим інноваційним математичним алгоритмам, PolyAnalyst поєднує в собі високу продуктивність "великих систем" з низькою вартістю, властивою програмам для Windows.

PolyAnalyst - один з небагатьох комерційних продуктів, в якому реалізовані не тільки методи аналізу числових даних, але і алгоритми Text Mining, - аналізу текстової інформації. Протягом своєї більш, ніж 10-річній історії, пакет безперервно розвивається, компанія-виробник додає нову функціональність, нові математичні модулі, планується портація системи на Unix платформи.

PolyAnalyst набув широкого поширення в світі. Більше 500 інсталяцій в 20 країнах світу, серед користувачів системи значний список складають найбільші світові корпорації: Boeing, 3M, Chase Manhattan Bank, Dupont, Siemens та інші. PolyAnalyst - універсальна система Data Mining, вона з успіхом застосовується в різних областях: у вирішенні бізнес-задач (direct marketing, cross-selling, customer retention), в соціологічних дослідженнях, в прикладних наукових і інженерних завданнях, в банківській справі, в страхуванні та медицині.

1.2. Архітектура системи

За своєю природою, PolyAnalyst є клієнт/серверним додатком. Користувач працює із клієнтською програмою PolyAnalyst Workplace. Математичні модулі виділені в серверну частину - PolyAnalyst Knowledge Server. Така архітектура надає природну можливість для масштабування системи: від однокористувацького варіанту до корпоративного рішення з декількома серверами. PolyAnalyst написаний на мові C++ з використанням специфікації Microsoft's COM (ACTIVEX). Ця специфікація встановлює стандарт комунікації між програмними компонентами. Математичні модулі (Exploration Engines) і багато інших компонентів PolyAnalyst виділені в окремі динамічні бібліотеки і доступні з інших додатків. Це дає можливість інтегрувати математику PolyAnalyst в ті, що існують ІС, наприклад, в CRM або ERP системи.

1.3. PolyAnalyst Workplace - лабораторія аналітика

Workplace - це клієнтська частина програми із призначеним для користувача інтерфейсом. Workplace є повнофункціональним середовищем для аналізу даних. Розвинені можливості маніпулювання з даними, багата графіка для представлення даних і візуалізації результатів, майстри створення об'єктів, наскрізний логічний зв'язок між об'єктами, мова символьних правил, інтуїтивне управління через drop-down і pop-up меню, докладна контекстна довідка - ось тільки декілька основних рис призначеного для користувача інтерфейсу програми.

Одиницею Data Mining дослідження в PolyAnalyst є "проєкт". Проєкт об'єднує в собі всі об'єкти дослідження, дерево проекту, графіки, правила, звіти ітд. Проєкт зберігається у файлі внутрішнього формату системи. Звіти досліджень представляються у форматі HTML і доступні через інтернет.

1.4. Загальносистемні характеристики PolyAnalyst

Типи даних
PolyAnalyst працює з різними типами даних. Це - числа, булеві змінні (yes/no), категоріальні змінні, текстові рядки, дати, а також вільний англійський текст.

Доступ до даних
PolyAnalyst може отримувати початкові дані з різних джерел. Це: текстові файли з роздільником кома (.csv), файли Microsoft Excel 97/2000, будь-яка ODBC- сумісна СУБД, SAS data files, Oracle Express, IBM Visual Warehouse.

Підтримка OLE DB for Data Mining

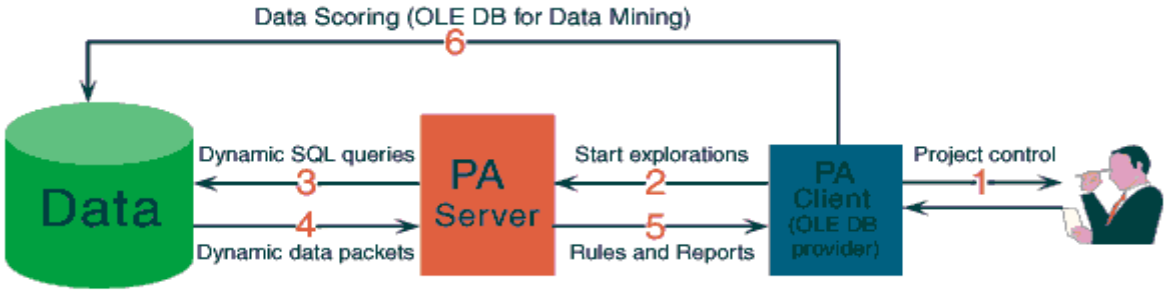
Версія 4.6 PolyAnalyst підтримує специфікацію Microsoft OLE DB for Data Mining (Version 1.0). При виконанні досліджень для більшості математичних модулів (LR, FD, CL, FC, DT, DF, FL,PN, BA, ТБ) можна створювати так звані "Mining Models" (MM). Після завершення аналізу ці моделі можна застосовувати до зовнішніх даних через стандартні інтерфейси OLE DB або ADO з інших програм або скриптів тих, що підтримують створення ADO або COM-об'єктів. Застосування моделі здійснюється за допомогою виконання SQL-команд (Розширення SQL for DM). Mining Models можна також експортувати в PMML.

У подальших планах розвитку програми намічається забезпечити інтеграцію "PolyAnalyst DataMining Provider" з Microsoft Analysis Services(у складі SQL Server 2000)

In-place Data Mining

PolyAnalyst підтримує запуск досліджень на зовнішніх даних через OLE DB інтерфейси при без завантаження цих даних в проект PA. При виконанні дослідження PolyAnalyst отримує дані порціями через виконання SQL запитів до зовнішніх джерел даних. Це дозволяє подолати обмеження пам'яті при дослідженні великих масивів даних.

SQL-Mode Data Mining Workflow Processes



PolyAnalyst Scheduler - режим пакетної обробки

У PolyAnalyst передбачена можливість пакетного режиму аналізу даних. Для цього є спеціальна скрипт- мова, на якій програмується всі аналітичні дії і тимчасова послідовність їх виконання, а також визначаються набори даних. Скрипт зберігається у файлі і автоматично ініціалізує дослідження у вказаний момент часу на певних даних. Для реалізації функції Scheduler в електронній ліцензії повинна бути включена відповідна опція.

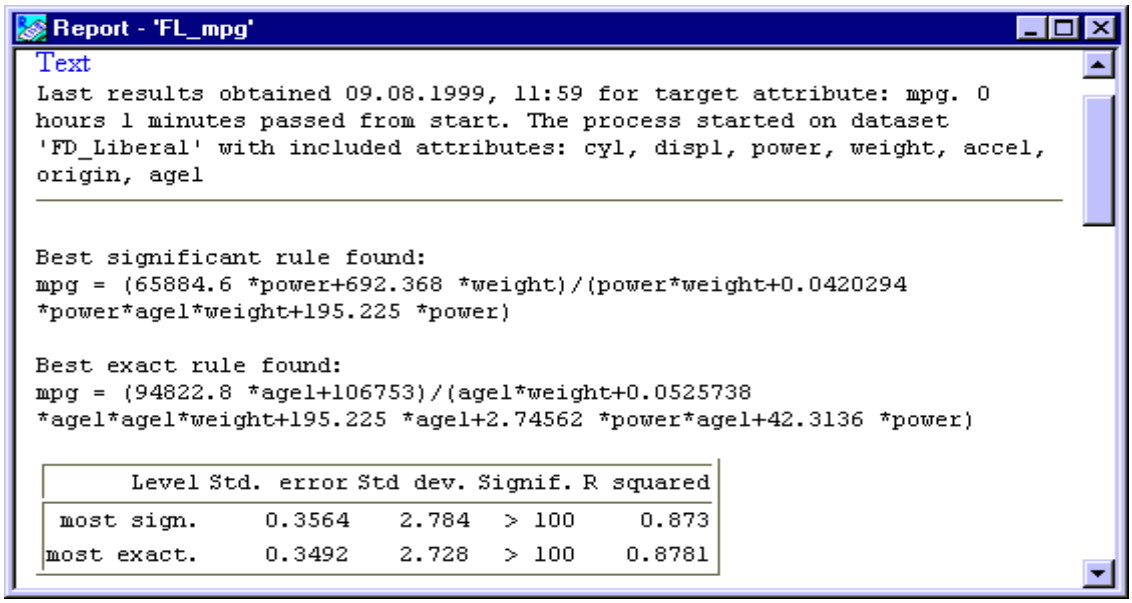
1.5. Аналітичний інструментарій PolyAnalyst

Версія PolyAnalyst 4.6 включає 18 математичних модулів, заснованих на різних алгоритмах Data і Text Mining. Більшість з цих алгоритмів є Know-How компанії Мегапьютер і не мають аналогів в інших системах. Алгоритми аналізу даних можна об'єднати в групи по їх функціональному призначенню: моделювання, прогнозування, кластеризація, класифікація, текстовий аналіз.

Модулі для побудови числових моделей і прогнозу числових змінних

Модуль Find Laws (FL) - будівник моделей

Модуль FL - це серцевина всієї системи. Алгоритм призначений для автоматичного знаходження в даних нелінійних залежностей (вид яких не задається користувачем) і представлення результатів у вигляді математичних формул, що включають і блоки умов. Здатність модуля FL автоматично будувати велике різноманіття математичних конструкцій робить його унікальним інструментом пошуку знання в символьному вигляді. Алгоритм заснований на технології еволюційного або як її ще називають генетичного програмування, вперше реалізованою в комерційних програмах компанією Мегапьютер.



PolyNet Predictor (PN) - поліноміальна нейронна мережа

Робота цього алгоритму заснована на побудові ієрархічної структури, подібній нейронній мережі. При цьому складність цієї мережевої структури та інші її параметри підбираються динамічно на основі властивостей аналізованих даних. Якщо створювана мережева структура не є дуже складною, то може бути побудований еквівалентний нею вираз на мові символьних правил системи. Якщо ж мережа дуже велика, то правило не може бути показане, проте його можна обчислити, або застосувати до початкових чи нових даних для побудови прогнозу.

Даний алгоритм надзвичайно ефективний в інженерних і наукових завданнях, коли потрібно побудувати надійний прогноз для числової змінної.

Stepwise Linear Regression (LR) - покрокова багатопараметрична лінійна регресія

Лінійна регресія як широко поширений метод статистичного дослідження, включена в багато статистичних пакетів і електронні таблиці. Проте, реалізація цього модуля в системі PolyAnalyst має свої особливості, а саме, автоматичний вибір найбільш значущих незалежних змінних і ретельна оцінка статистичної значущості результатів. Потрібно відмітити, що в даному випадку значущість відрізняється від значущості одиничної регресійної моделі, оскільки протягом одного запуску даного обчислювального процесу може бути перевірене велике число регресійних моделей.

Алгоритм працює дуже швидко і застосовний для побудови лінійних моделей на змішаних типах даних.

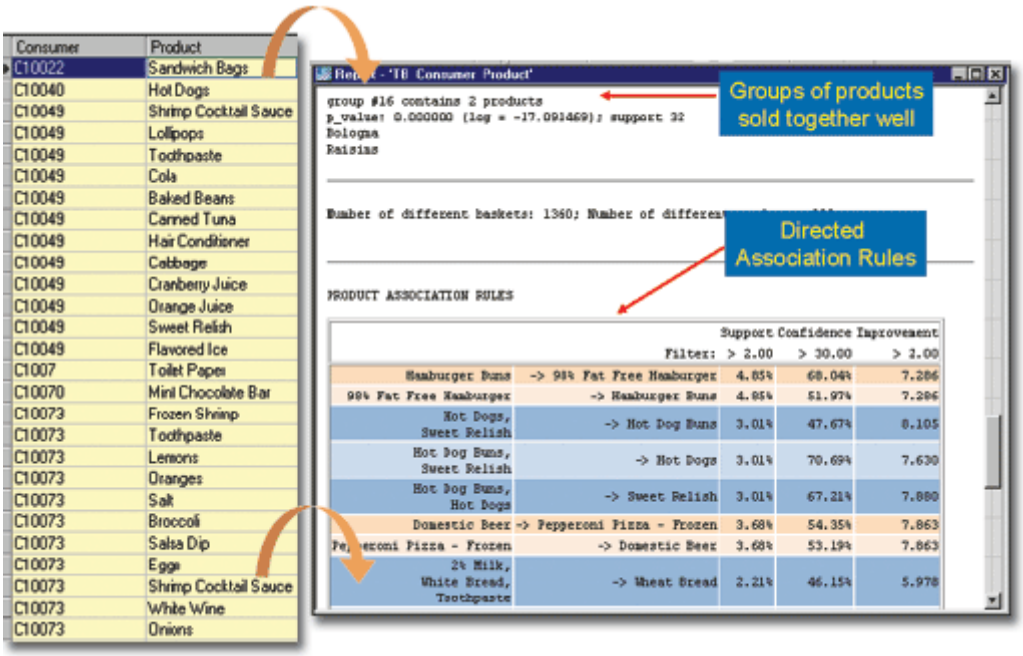
Discriminate (DS) - Дискрімінація

Даний алгоритм є модифікацією алгоритму CL - класифікатор на основі нечіткої логіки. Він призначений для того, щоб з'ясувати, чим дані з вибраної таблиці відрізняються від решти даних, включених в проект, іншими словами для виділення специфічних рис, що характеризують деяку підмножину записів проекту. На відміну від алгоритму CL, він не вимагає завдання цільовій змінній, досить вказати лише таблицю, для якої потрібно знайти відмінності.

Алгоритми асоціації

Market Basket Analysis (BA) - метод аналізу "корзини покупця"

Назва цього методу походить від завдання визначення які товари ймовірно купуються спільно. Проте реальна область його застосування значно ширша. Наприклад, продуктами можна вважати сторінки в Інтернеті, або ті або інші характеристики клієнта або відповіді респондентів в соціологічних і маркетингових дослідженнях ітд. Алгоритм BA отримує на вхід бінарну матрицю, в якій рядок - це одна корзина (касовий чек, наприклад), а стовпці заповнені логічними 0 і 1, такими, що позначають наявність або відсутність даної ознаки (товару). На виході формуються кластери ознак, що спільно зустрічаються, з оцінкою їх вірогідності і достовірності. Окрім цього формуються асоціативні направлені правила типу: якщо ознака "A", то з такою - то вірогідністю ще і ознака "B" і ще ознака "C". Алгоритм BA в PolyAnalyst працює виключно швидко і здатний обробляти величезні масиви даних.



Transactional Basket Analysis (ТБ) - транзакційний аналіз "корзини"

Transactional Basket Analysis - це модифікація алгоритму BA, вживаний для аналізу дуже великих даних, що не рідкість для цього типу завдань. Він припускає, що кожен запис в базі даних відповідає одній транзакції, а не одній корзині (набору куплених за одну

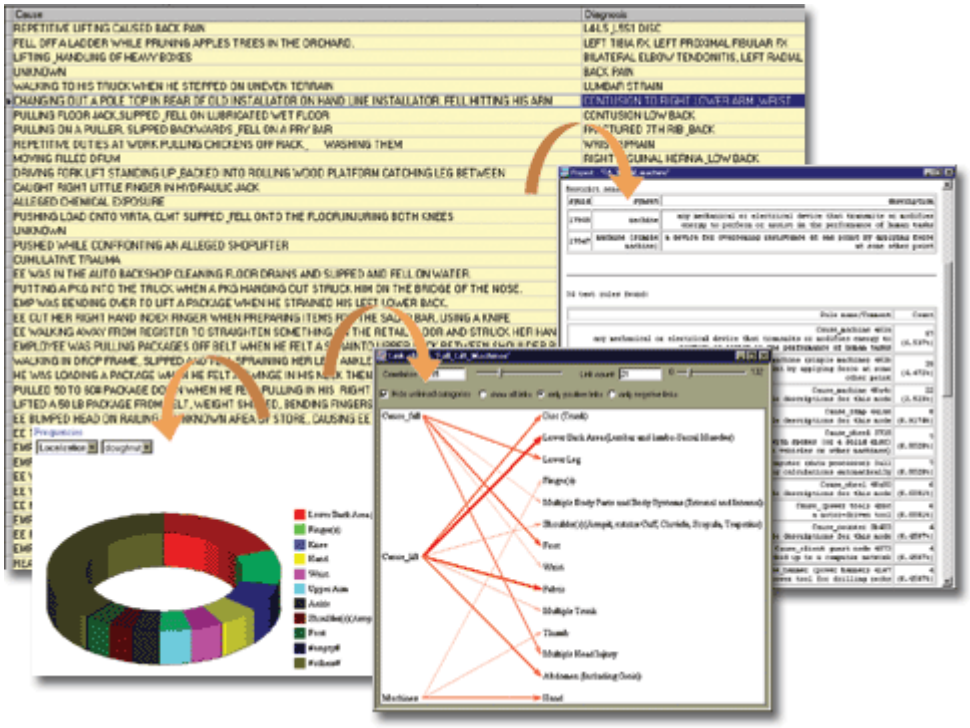
операцію товарів). На основі цього алгоритму компанія "Мегапьютер" окремий продукт - X-SellAnalyst, призначений для on-line рекомендації продуктів в Інтернет магазинах.

Модулі текстового аналізу.

Однією з унікальних особливостей PolyAnalyst є інтеграція інструментів Data Mining - засобів аналізу числової інформації з методами аналізу текстів на природній мові - алгоритмів Text Mining. На жаль в поточній версії програми алгоритми Text Mining реалізовані тільки для англійської мови, проте в найближчих планах виробника забезпечити і підтримку російського і ряду інших європейських мов.

Text Analysis (TA) - текстовий аналіз

Text Analysis є засобом формалізації неструктурованих текстових полів в базах даних. При цьому текстове поле представляється як набір булевих ознак, заснованих на наявності і/або частоті даного слова, стійкого словосполучення або поняття (з урахуванням відносин синонімії і "общєє- приватне") в даному тексті. При цьому з'являється можливість розповсюдити на текстові поля всю потужність алгоритмів Data Mining, реалізованих в системі PolyAnalyst. Крім того, цей метод може бути використаний для кращого розуміння текстовою компоненти даних за рахунок автоматичного виділення найбільш ключових понять.



Text Categorizer (TC) - каталогізатор текстів

Цей модуль дозволяє автоматично створити ієрархічний деревовидний каталог наявних текстів і помітити кожен вузол цієї деревовидної структури найбільш індикативним для текстів, що відносяться до нього. Це потрібно для розуміння тематичної структури аналізованої сукупності текстових полів і ефективної навігації по ній.

Link Terms (LT) - зв'язок понять

Цей модуль дозволяє виявляти зв'язки між поняттями, що зустрічаються в текстових полях бази даних, що вивчається, і представляти їх у вигляді графа. Цей граф також може бути використаний для виділення записів, що реалізовують вибраний зв'язок.

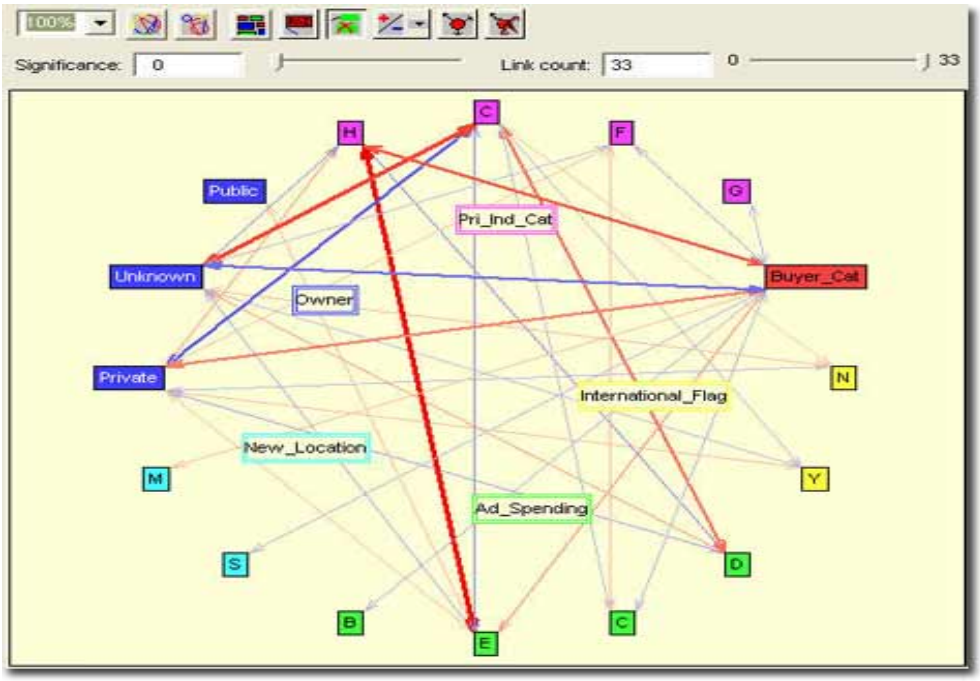
Візуалізація

У PolyAnalyst є багатий набір інструментів для графічного уявлення і аналізу даних і результатів досліджень. Дані можуть представлятися в різних зорових форматах: гістограмах, двовимірних, псевдо- і реальних тривимірних графіках.

Знайдені в процесі Data Mining залежності можуть бути представлені як інтерактивні графіки із слайдерами для зміни значень представлених на них змінних. Ця особливість дозволяє користувачеві графічно моделювати результати. Є набір спеціальних графіків, широко вживаних в бізнесі, це так звані Lift, Gain charts, які використовуються для графічної оцінки якості класифікаційних моделей і вибору оптимального числа контактів (prospects). Окрім цього в останню версію програми включений новий візуальний метод Data Mining - аналіз зв'язків.

Link Analysis (LA) - аналіз зв'язків

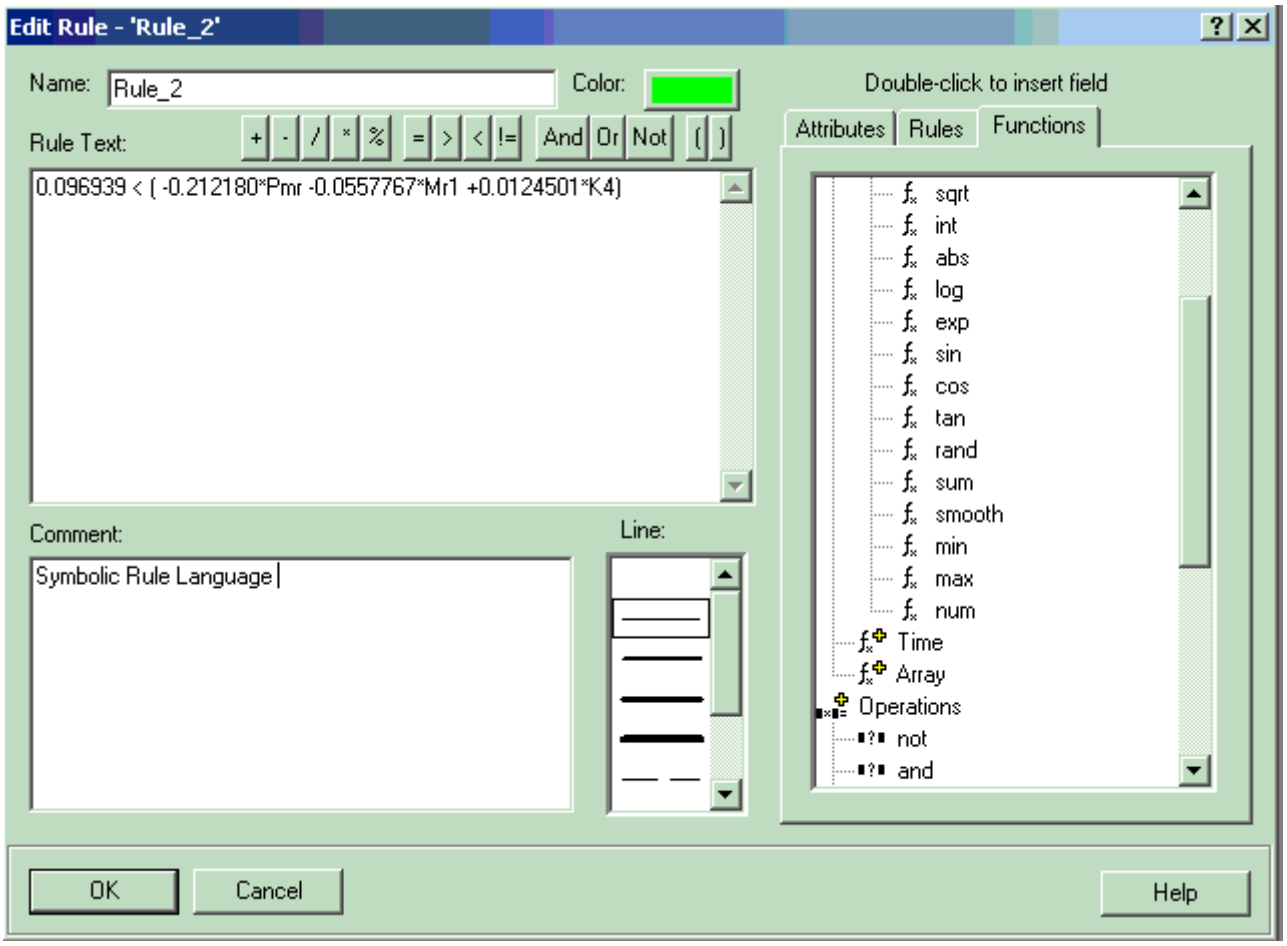
Модуль Link Analysis дозволяє виявляти кореляційні і антикореляційні зв'язки між значеннями категоріальних і булевих полів і представляти їх у вигляді графа Цей граф також може бути використаний для виділення записів, що реалізовують вибраний зв'язок.



Symbolic Rule Language (SRL) - мова символьних правил

SRL - це універсальна алгоритмічна мова PolyAnalyst, яка використовується для символьного уявлення автоматично знайдених системою в процесі Data Mining правив, а також для створення користувачем своїх власних правил. На мові SRL можна виразити

широкий спектр математичних конструкцій, використовуючи операції алгебри, великий набір вбудованих функцій, операції з датами і часом, логічні і умовні конструкції. Для



зручності написання виразів на SRL в програмі передбачений майстер створення правил.

1. Завдання для виконання:

1.4 Перед виконанням роботи ознайомитись із теоретичними відомостями.

1.5 У відповідності до отриманого номеру варіанта описати основні алгоритми аналізу даних і вказати, де задані модулі доцільно використовувати.

1.6 Навести приклад практичного застосування заданого алгоритму , оформити та представити результати.

1.7 Створити текстові коментарі до роботи.

3. Зміст звіту

3.1. Тема та мета роботи.

3.2. Коротко основні теоретичні відомості.

3.3. Відобразити отримані результати (п. 2.2.-2.4).

3.4.Висновки за результатами виконаної роботи.

4. Контрольні запитання

4.1. Поняття ІАД.

4.2. Основні класи процесів ІАД.

4.3. Архітектура системи багатовимірного інтелектуального аналізу даних.

4.4. Характеристика системи PolyAnalyst.

4.5. Архітектура системи PolyAnalyst.

4.6. Аналітичний інструментарій PolyAnalyst.

4.7. Модулі текстового аналізу.

4.8. Візуалізація в PolyAnalyst.

4.9. Текстовий аналіз в PolyAnalyst.

ЛАБОРАТОРНА РОБОТА №2

Тема : Text Mining – аналіз текстової інформації.

Мета: Навчитись здійснювати аналіз текстової інформації в Text Mining

Теоретичні відомості

Аналіз структурованої інформації, яка зберігається в БД, потребує попередньої обробки : проектування БД, ввід інформації за визначеними правилами, розміщення її в спеціальних структурах тощо. Таким чином, безпосередньо для аналізу цієї інформації і отримання з неї нових знань необхідно задіяти додаткові зусилля. При цьому вони не завжди зв'язані з аналізом і не обов'язково приводить до бажаного результату. Через це КПД структурованої інформації знижується, крім цього не всі види даних можна структурувати без втрати корисної інформації.

Наприклад, текстові документи практично не можливо перетворити в табличне представлення без втрати семантики тексту і відношень між сутностями. По цій причині такі документи зберігаються в БД без змін, як текстові поля (BLOB-поля). В цей же час в тексті приховано велику кількість інформації, але її неструктурованість не дозволяє використовувати до неї алгоритми Data Mining.

Рішенням даної проблеми займаються методи аналізу неструктурованого тексту, в західній літературі такий аналіз має назву Text Mining.

Методи аналізу у неструктурованих текстах лежать на стику декількох областей: Data Mining, обробки природніх мов, пошуку інформації, вилучення інформації та управління знаннями.

Знаходження знань в тексті (Text Mining) – це нетравіальний процес знаходження по-справжньому нових, потенційно корисних і зрозумілих шаблонів в неструктурованих текстових даних.

Під визначенням «неструктурованих текстових даних» ховається набір документів, які являють собою логічно об'єднаний текст без будь яких обмежень накладених на його структуру. Прикладами таких документів є : web-сторінки, електронна пошта, нормативні документи і так далі. В загальному випадку такі документи можуть складними і великими, а також включати в себе і графічну інформацію. Документи, що використовують мову розширеної розмітки (XML), стандартна мова загальної (SGML) та інші подібні по структурі формування тексту мови, прийнято називати напівструктурованими документами.

Процес аналізу текстових документів, можна представити як послідовність декількох кроків:

1. *Пошук інформації.* На першому кроці потрібно ідентифікувати документи, які мають бути піддані аналізу, і забезпечити їх доступність. Як правило, користувачі можуть визначити набір документів для аналізу самостійно – вручну, але при великій кількості документів варто використовувати варіанти автоматизованого відбору по заданим критеріям.
2. *Попередня обробка документів.* На ньому виконується найпростіші, проте необхідні перетворення з документами, для представлення їх у вигляді необхідному для роботи з методами Text Mining. Ціллю таких перетворень є видалення лишніх слів і надання тексту більш формалізованої форми.
3. *«Добування» інформації.* Витягнення інформації із обраних документів припускає виділення в них ключових понять, над якими в подальшому буде виконуватись аналіз. Даний етап є дуже важливим.
4. *Застосування методів Text Mining.* На даному кроці витягаються шаблони і відношення, які присутні в текстах. Даний крок є основним в процесі аналізу текстів.
5. *Інтерпретація результатів.* Як правило, інтерпретація заключається або в представленні результатів природною мовою, або в їх візуалізації в графічному вигляді.

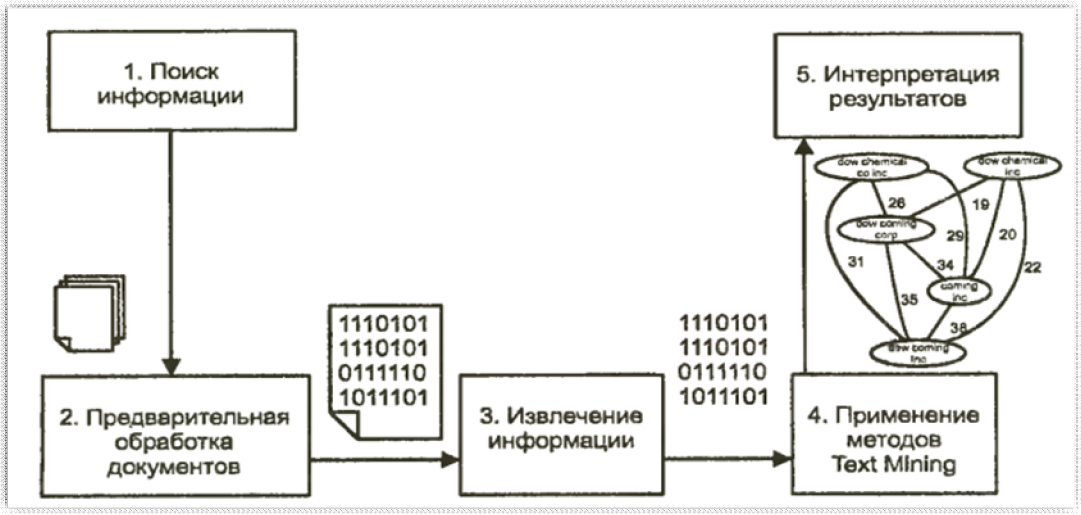


Рис. 1 – Етапи Text Mining

Попередня обробка

Однією із головних проблем аналізу текстів являється велика кількість слів в документі. Якщо кожне слово піддати аналізу, то час пошуку нових знань різко зросте і навряд чи задовольнить запити користувача. В той же час очевидно, що не всі слова в тексті несуть корисну інформацію. Крім цього, в силу гнучкості природних мов формально різні слова (синоніми і т.д.) на справді означають однакові поняття. Таким чином, видалення неінформативних слів, а також приведення близьких по змісту слів до єдиної форми значно зменшують час аналізу текстів. Усунення описаних проблем виконується на етапі попередньої обробки тексту.

Зазвичай використовують наступні прийоми видалення неінформативних слів і підвищення строгості текстів :

- ✓ *Видалення стоп-слів.* Стоп-словами називають слова, які являються допоміжними і несуть мало інформації про вміст документа. Зазвичай попередньо складаються

списки таких слів, і в процесі попередньої обробки вони видаляються із тексту. Прикладами таких слів є допоміжні слова і артиклі.

- ✓ *Стеммінг – морфологічний пошук*. Він полягає в приведенні кожного слова до його нормальної форми. Нормальна форма виключає множинні форми, відмінювання слова, особливості усного мовлення. Наприклад, слово «стиснення» і «стиснений» повинні бути приведені до нормальної форми слова «стискати». Алгоритми морфологічного розбору враховують мовні особливості і внаслідок цього являються мовно-залежними алгоритмами.
- ✓ *N-грами* – альтернатива морфологічному розбору і стоп-слів. N-грама – це частина стрічки, що складається з N символів. Наприклад слово «дата» може бути представлене 3-грамою «_да», «дат», «ата», «та_» або 4-граммою «_дат», «дата», «ата_», де символ підкреслення заміняє попередній чи замикаючий слово пробіл. У порівнянні з стеміннгом чи видаленням стоп-слів, N-грами меньш чутливі до граматичних і типографічних помилок. Крім цього, N-грами не потребують лінгвістичного представлення слів, що робить даний прийом більш незалежним від мови. Проте N-грами дозволяючи робити текст більш строгим, не вирішують проблему зменшення кількості неінформативних слів.
- ✓ *Приведення регістра*. Цей прийом полягає в перетворенні символів до верхнього чи нижнього регістру. Наприклад, всі слова «текст», «Текст», «ТЕКСТ» приводяться до нижнього регістру «текст».

Хід роботи

Система PolyAnalyst призначена для автоматичного і напівавтоматичного аналізу числових баз даних і витягання з сирих даних практично корисних знань. PolyAnalyst знаходить багатофакторні залежності між змінними в базі даних, автоматично будує і тестує багатовимірні нелінійні моделі, що виражають знайдені залежності, виводить класифікаційні правила по повчальних прикладах, знаходить в даних багатовимірні кластери, будує алгоритми рішень.

Основу PolyAnalyst складають так звані Exploration engines або Машини досліджень - математичні модулі, засновані на різних DM алгоритмах, і призначені для автоматичного аналізу даних. Даний додаток працює з різними типами даних. Це - числа, булеві змінні (yes/no), категоріальні змінні, текстові рядки, дати, а також вільний англійський текст.

PolyAnalyst - один з небагатьох комерційних продуктів, в якому реалізовані не тільки методи аналізу числових даних, але і алгоритми Text Mining, - аналізу текстової інформації.

Text Analysis (TA) - текстовий аналіз

Text Analysis є засобом формалізації неструктурованих текстових полів в базах даних. При цьому текстове поле представляється як набір булевих ознак, заснованих на наявності і/або частоті даного слова, стійкого словосполучення або поняття (з урахуванням відносин синонімії і "загальне- приватне") в даному тексті. При цьому з'являється можливість розповсюдити на текстові поля всю потужність алгоритмів Data Mining, реалізованих в системі PolyAnalyst. Крім того, цей метод може бути використаний

для кращого розуміння текстовою компоненти даних за рахунок автоматичного виділення найбільш ключових понять.

1. Створюємо новий проект і називаємо його «Text_mining example»

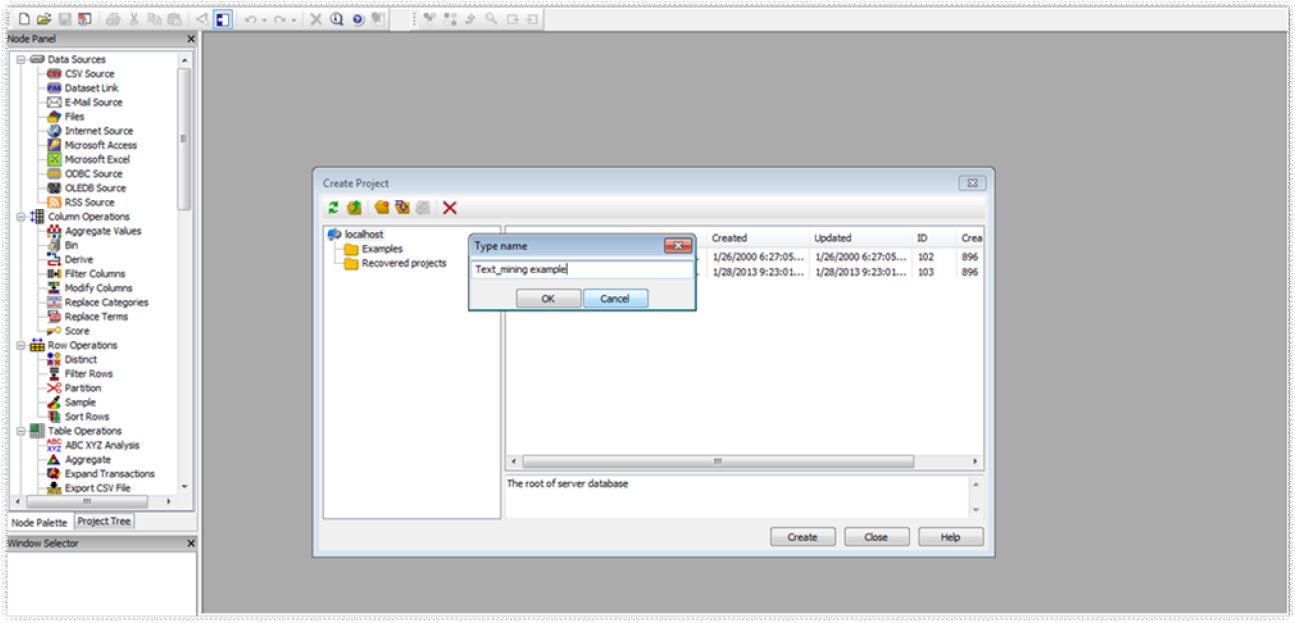


Рис. 3 – Створення нового проекту

2. Додаємо із rss стрічку із сайту BBC на англійській мові «http://feeds.bbci.co.uk/news/rss.xml».

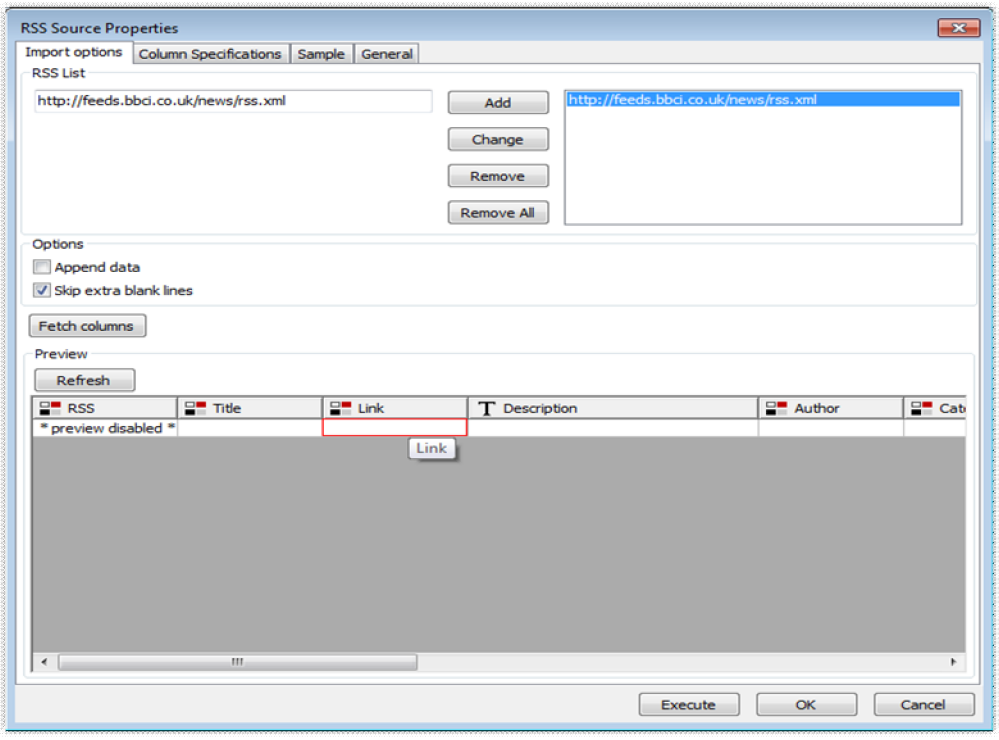


Рис. 4 – Задання rss-джрела

3. Обираємо один із алгоритмів Text Analysis – Entity Extraction.

Даний алгоритм здійснює пошук по всьому документі сутностей, за заданими нами атрибутами та опціями. Ось як ми здійснюємо задання шаблону пошуку :

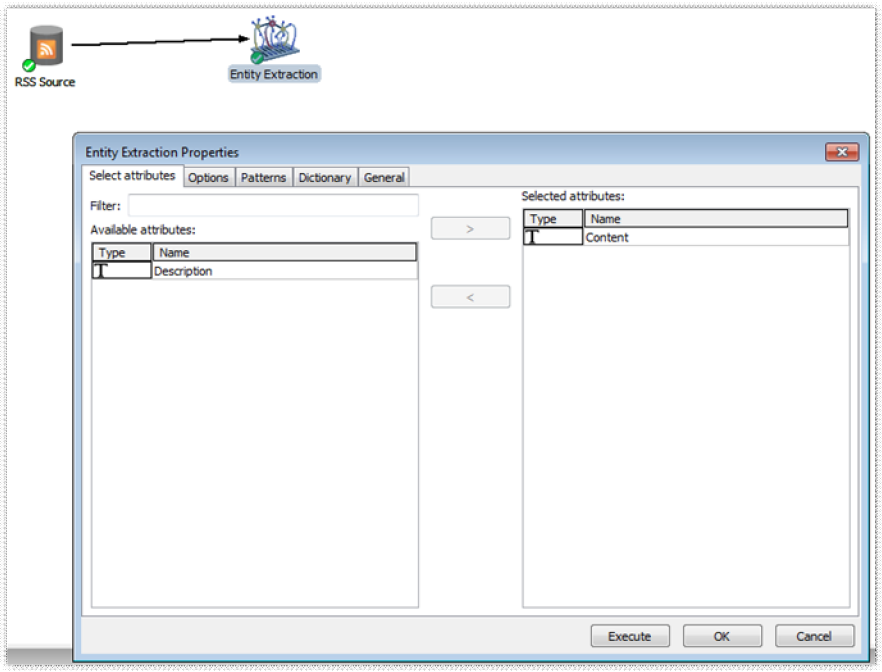


Рис. 5 – Ми обрали із двох доступних нам атрибутів один «Content»

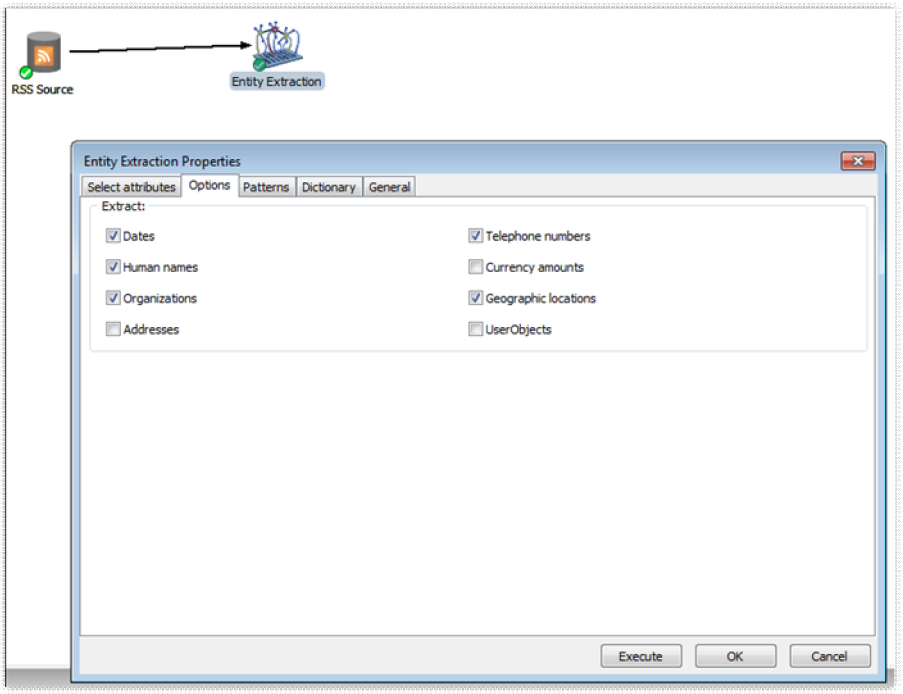


Рис. 6 – Ми відзначили галочками за якими запитами здійснюється аналіз.

4. Результати Entity Extraction.

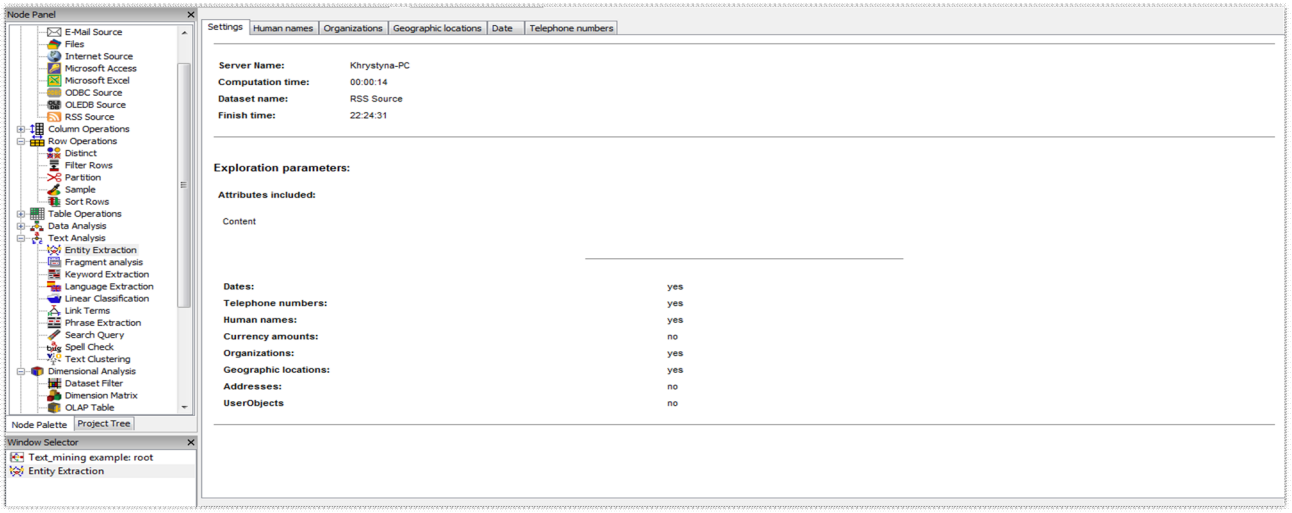


Рис. 7 – Загальна інформація про результати аналізу.

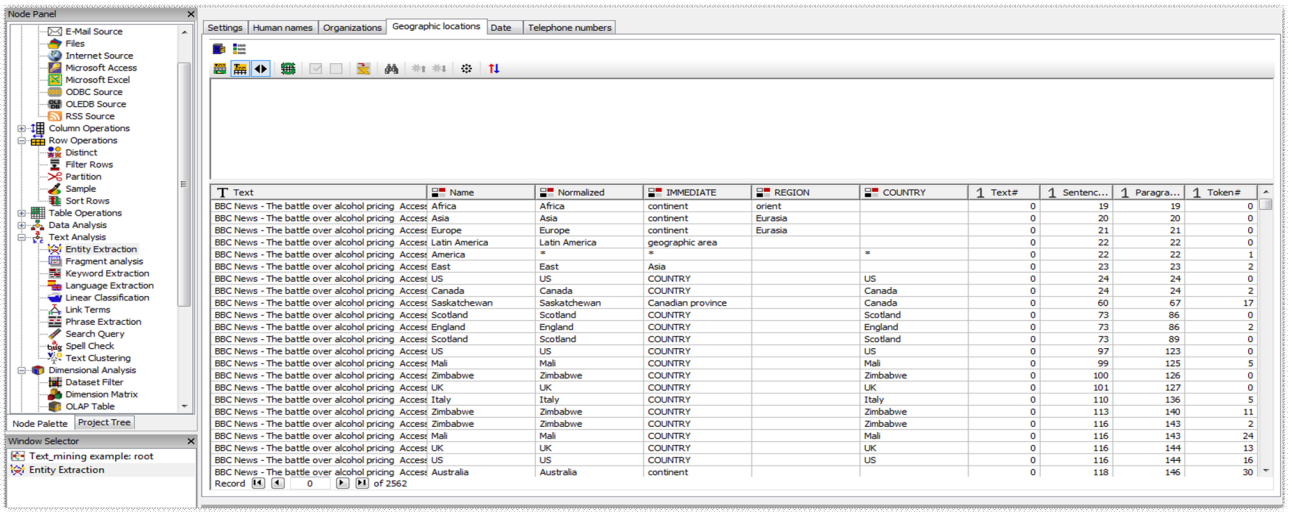


Рис 8. Результат аналізу за заданим запитом «Geographic location»

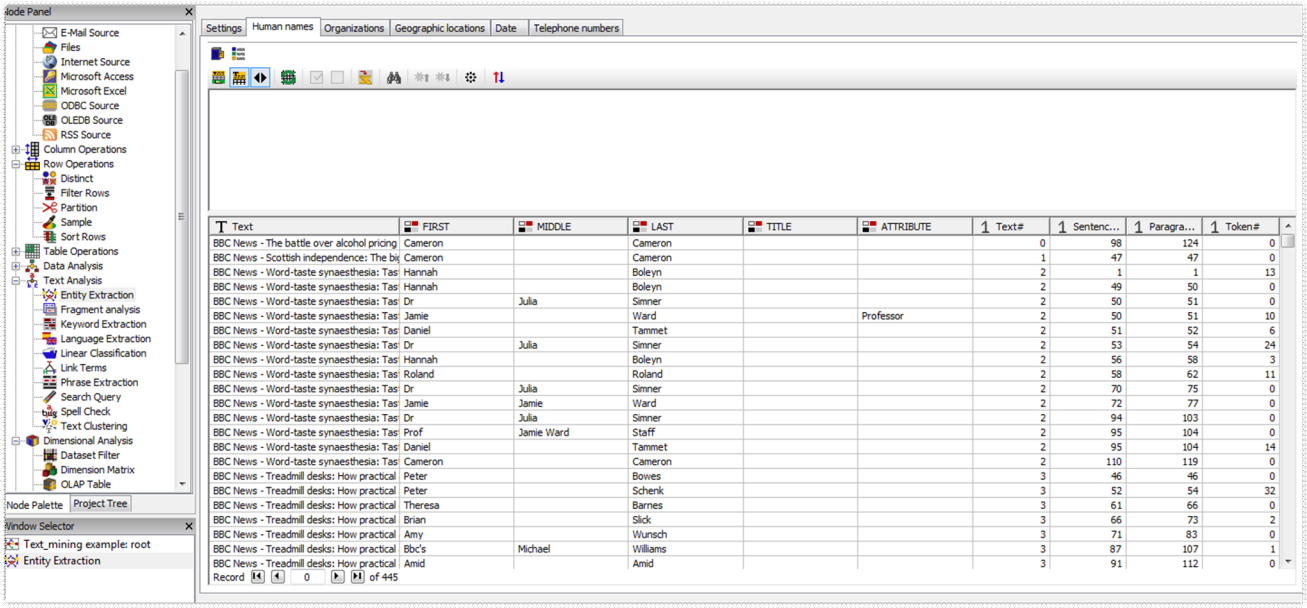


Рис. 9 - Результат аналізу за заданим запитом «Human names»

2. Завдання для виконання:

1. Перед виконанням роботи ознайомитись із теоретичними відомостями.
2. У відповідності до отриманого номеру варіанта описати основні алгоритми текстового аналізу даних і вказати, де задані модулі доцільно використовувати.
3. Навести приклад практичного застосування заданого алгоритму , оформити та представити результати.
4. Створити текстові коментарі до роботи.

3. Зміст звіту

3.1. Тема та мета роботи.

3.2. Коротко основні теоретичні відомості.

3.3. Відобразити отримані результати (п. 2.2.-2.4).

3.4.Висновки за результатами виконаної роботи.

ЛАБОРАТОРНА РОБОТА № 3

Тема: Кластерний аналіз
Мета: Навчитись практично здійснювати кластер ний аналіз

Теоретичні відомості

Завдання та умови

Кластерний аналіз виконує такі *основні завдання:*

- Розробка типології або класифікації.
- Дослідження корисних концептуальних схем групування об'єктів.
- Породження гіпотез на основі дослідження даних.
- Перевірка гіпотез або дослідження для визначення, чи дійсно типи (групи), виділені тим чи іншим способом, присутні у наявних даних (примітка 1).

Незалежно від предмета вивчення застосування кластерного аналізу припускає *наступні етапи:*

- Відбір вибірки для кластеризації.
- Визначення безлічі змінних, за якими будуть оцінюватися об'єкти у вибірці.
- Обчислення значень тієї чи іншої міри подібності між об'єктами.
- Застосування методу кластерного аналізу для створення груп схожих об'єктів.
- Перевірка достовірності результатів кластерного рішення (примітка 1).

Кластерний аналіз пред'являє наступні *вимоги до даних:*

1 показники не повинні корелювати між собою
2 показники повинні бути безрозмірними
3 розподіл показників має бути близько до нормального
4 показники повинні відповідати вимогу "стійкості", під якою розуміється відсутність впливу на їх значення випадкових факторів
5 вибірка повинна бути однорідна, не містити "викидів" (примітка 2).

Якщо кластерному аналізу передує факторний аналіз, то вибірка не потребує "ремонту" - викладені вимоги виконуються автоматично самою процедурою факторного

моделювання (є ще одна перевага - z-стандартизація без негативних наслідків для вибірки; якщо її проводити безпосередньо для кластерного аналізу, вона може спричинити за собою зменшення чіткості поділу груп). В іншому випадку вибірку потрібно коригувати.

Приклад виконання:

Результат виконання

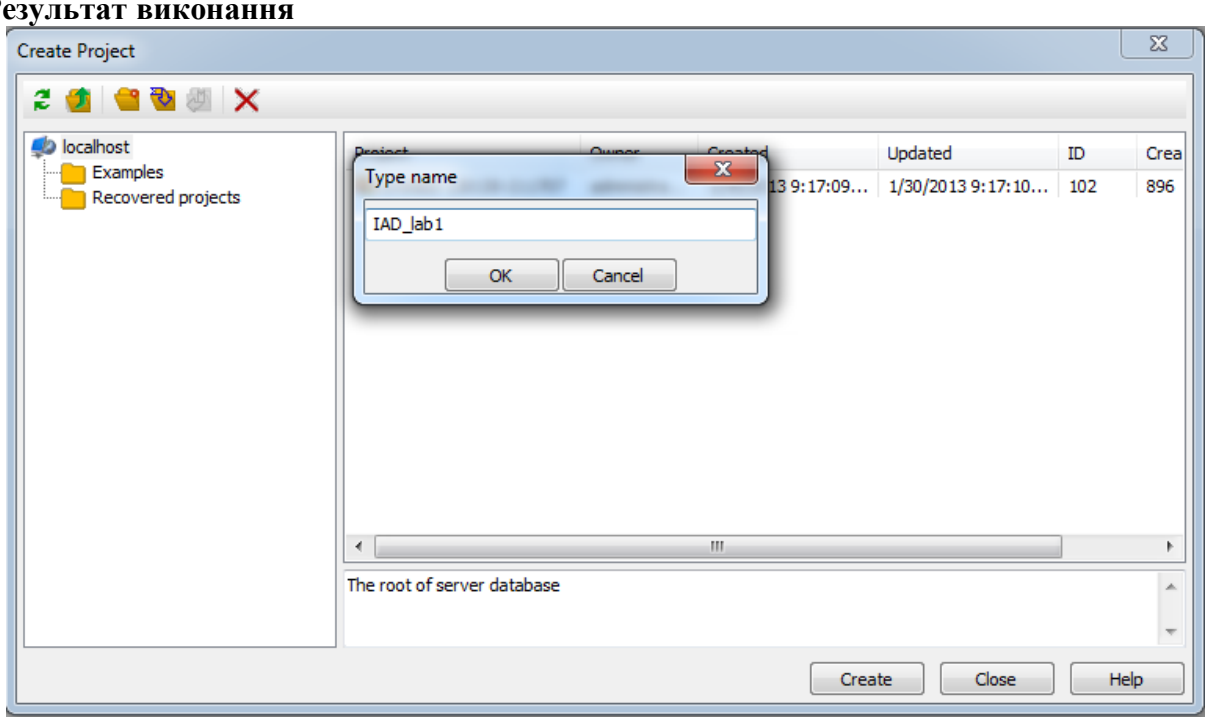


Рис. 1 Створення проекту в PolyAnalist

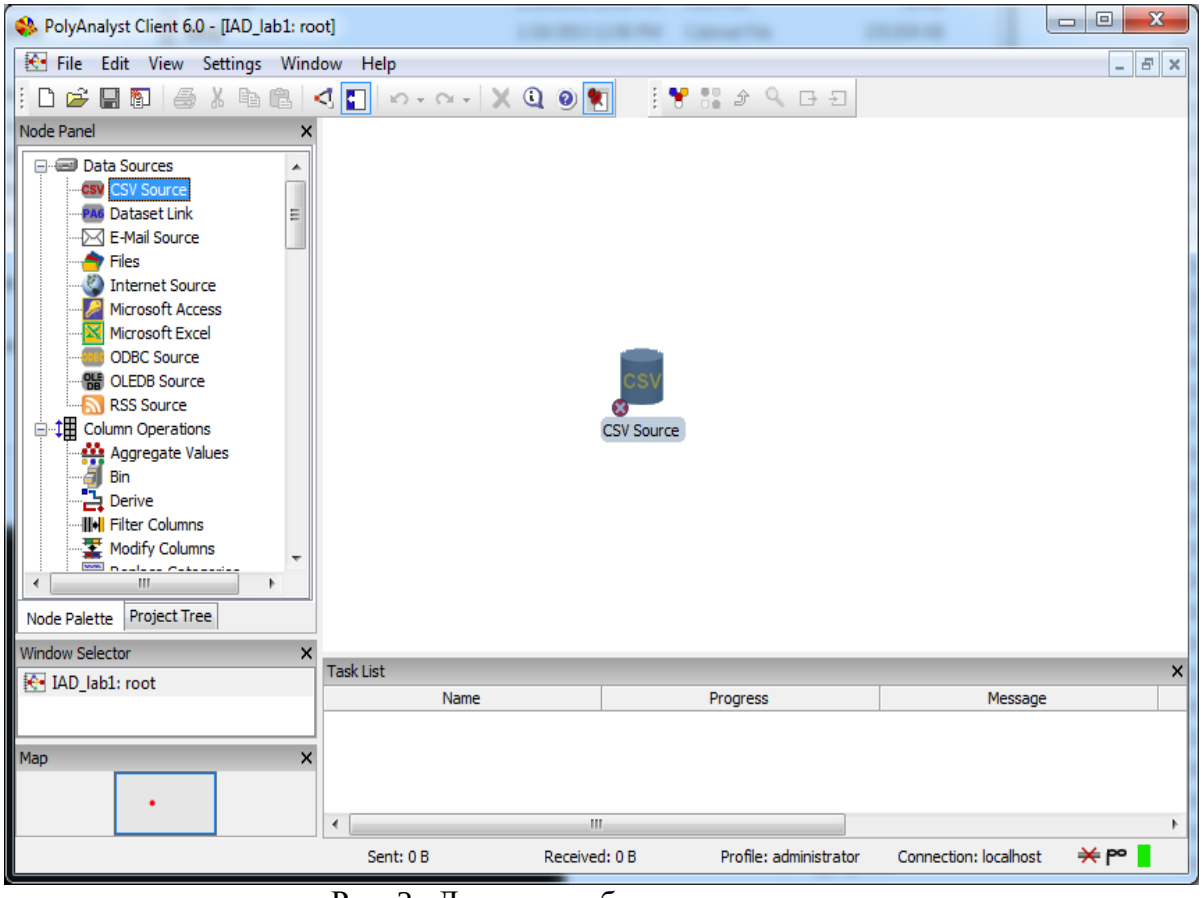


Рис. 2. Додавання бази даних в проєкт

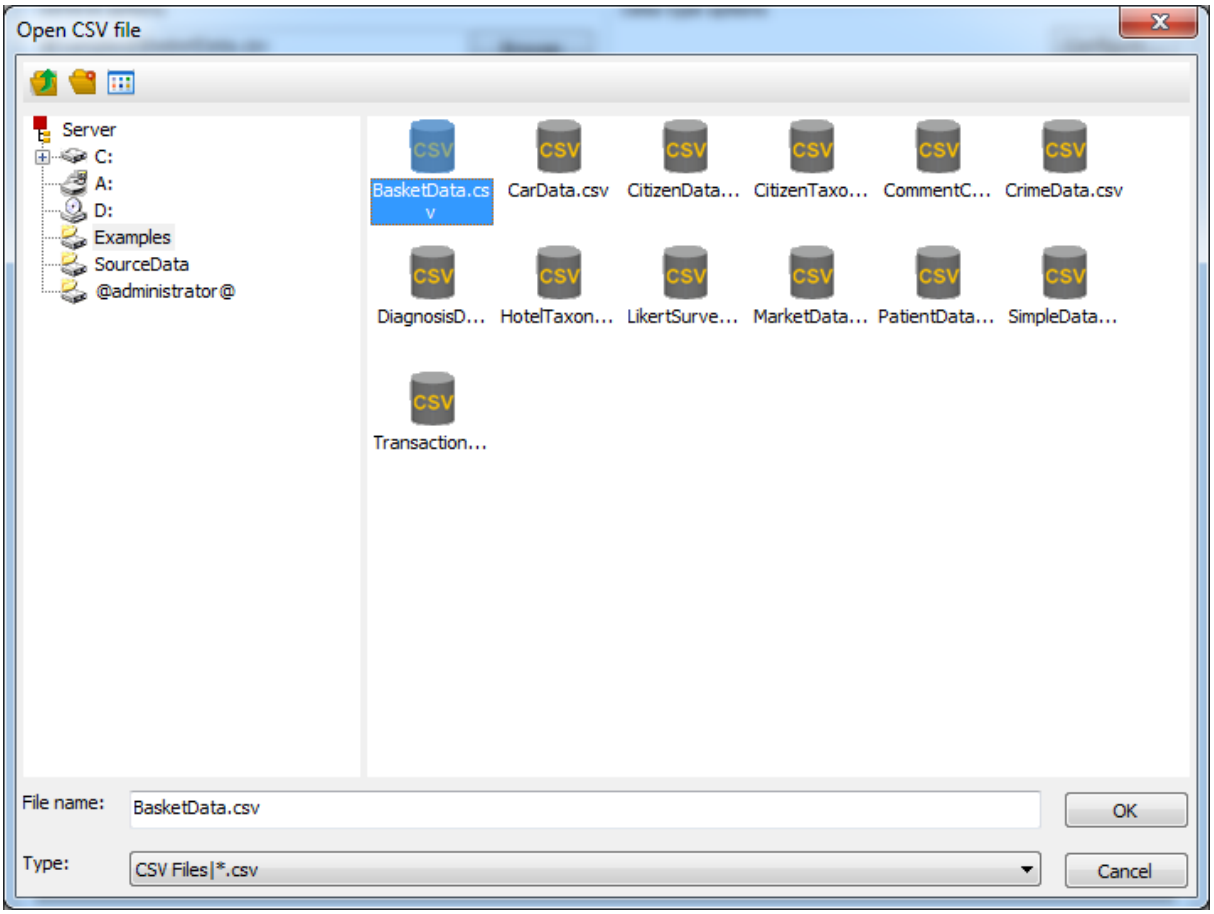


Рис. 3. Додавання даних до бази даних

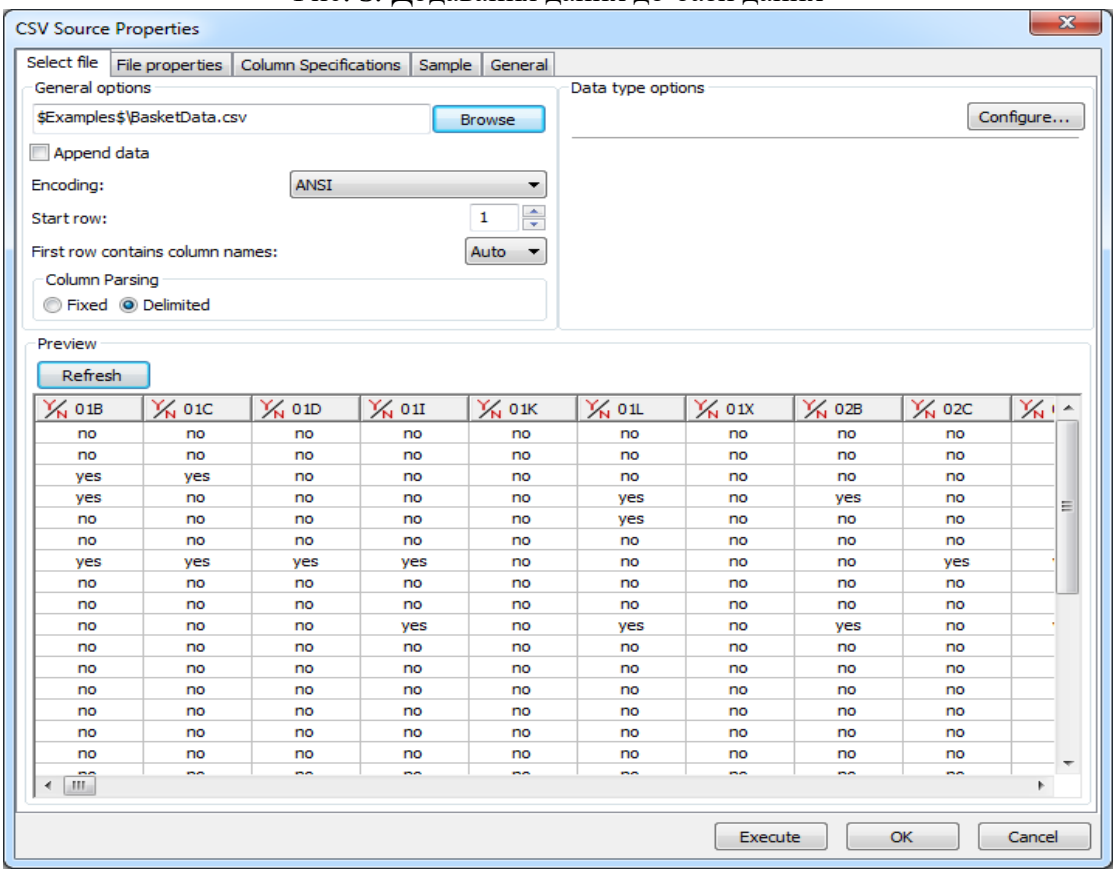


Рис. 4 Натискаємо на кнопку Execute

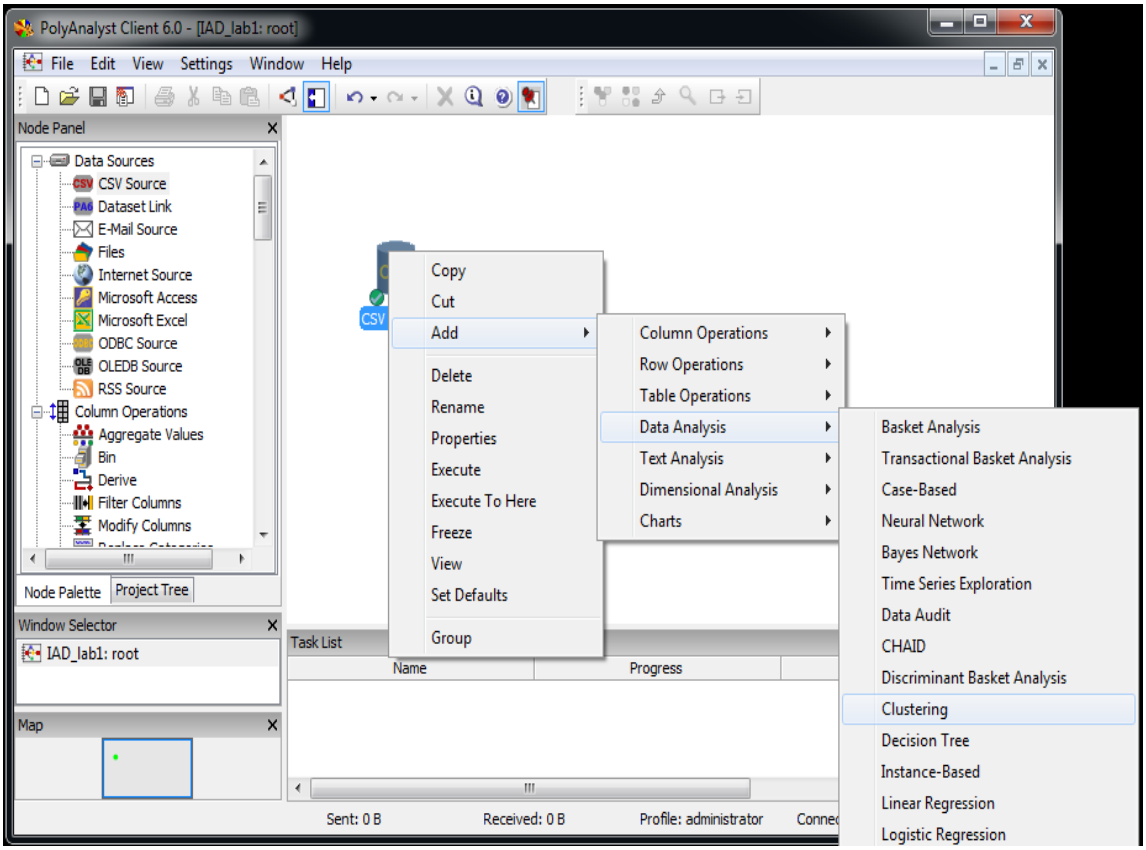


Рис. 5 . Додавання Кластерний аналіз до бази даних

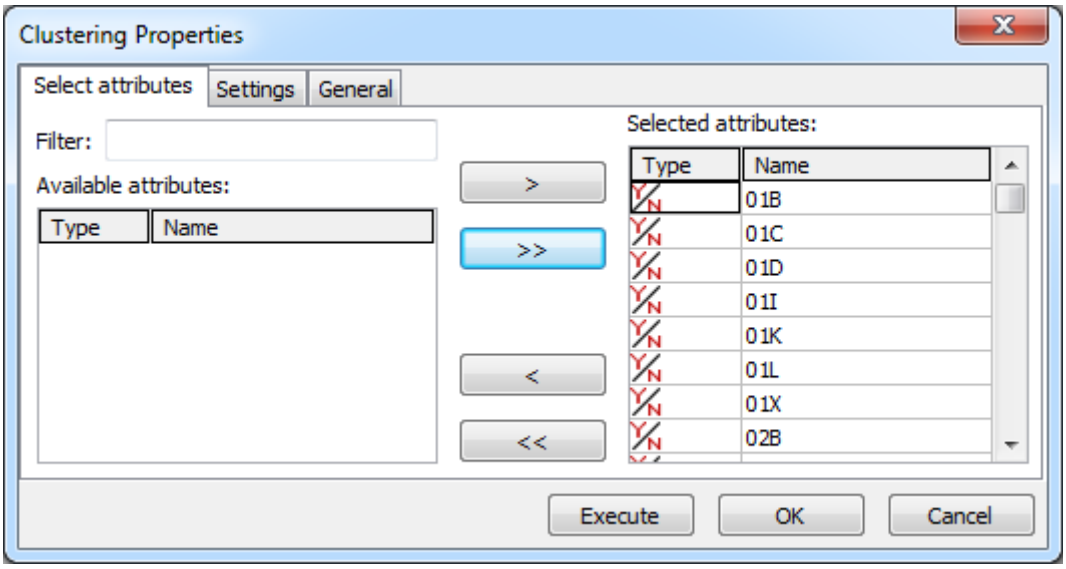


Рис..6 Додавання даних до кластерного аналізу

Після виконання кластеризації результати можна переглянути в вкладках Settings і Clustering. В вкладці Settings зображено конфігуріційні настройки для виконання і загалні дані процесу виконання. В вкладці Clustering зображено дані ,які згенеровані безпосередньо в результаті кластеризіції.

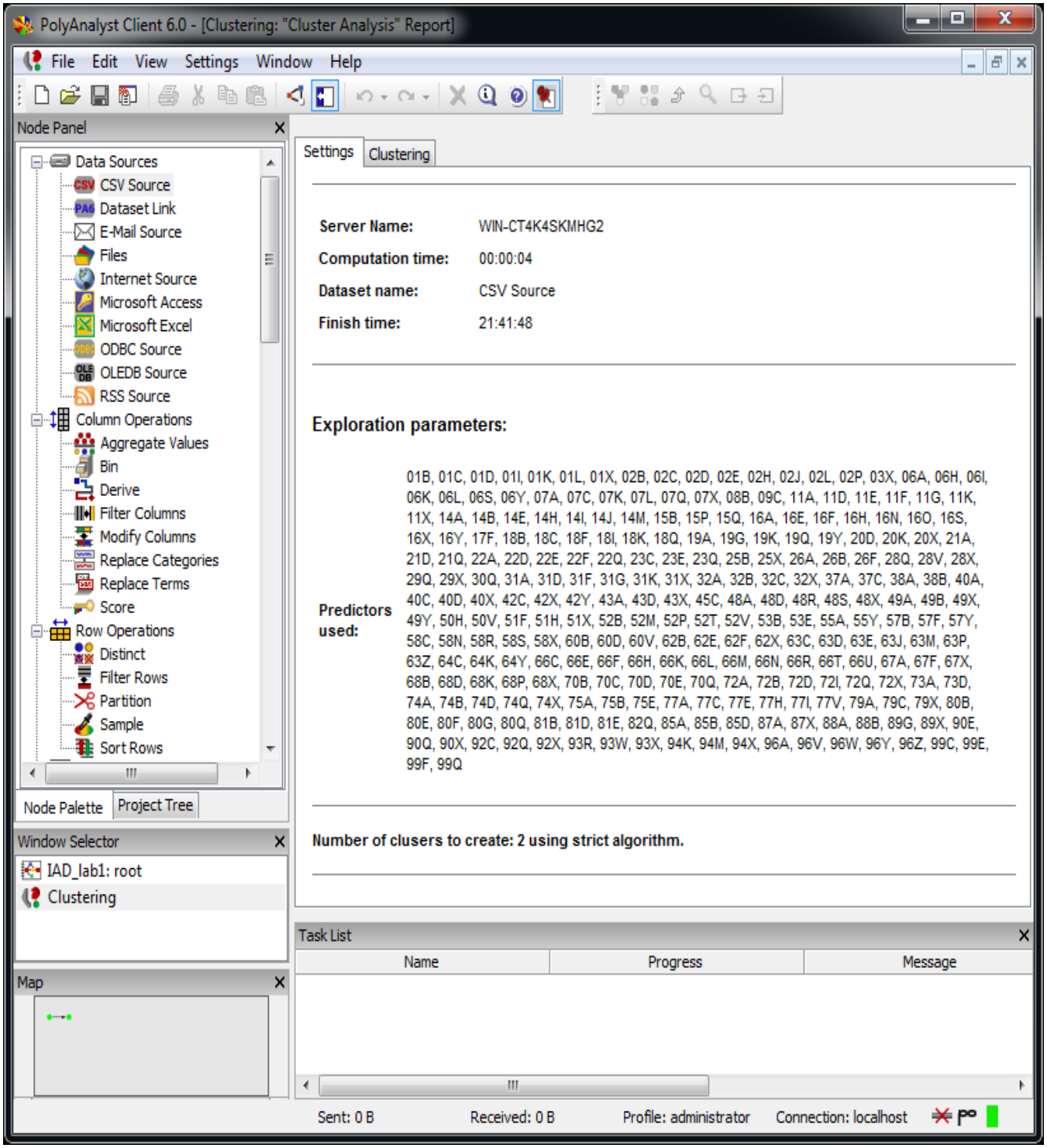


Рис. 7. Результат виконання кластеризації в вкладці Settings

На рис. 8. зображено кількість записів для кожного кластера:

Settings | Clustering

Clusters

Cluster	Cluster records
~1	144
~2	1,031

Рис..8. Результат виконання кластеризації в вкладці Clustering.
Частина Clusters.

Cluster Description

Attribute	Mean Value	Std. Dev	Categories Distribution	Magnitude	Distinction
01B			Category "NA" probability is 1	0.002168276	0.48
01C			Category "NA" probability is 0.999953	0.002028325	0.44
01D			Category "NA" probability is 0.992496	0.001703098	0.36
01I			Category "NA" probability is 0.97087 Category "no" probabilit	0.001514579	0.31
01K			Category "NA" probability is 0.892636 Category "no" probabil	0.001249249	0.27
01L			Category "NA" probability is 0.870238 Category "no" probabil	0.001191425	0.26
01X			Category "NA" probability is 0.898939 Category "no" probabil	0.001147452	0.24
02B			Category "NA" probability is 0.84854 Category "no" probabilit	0.001044872	0.22
02C			Category "NA" probability is 0.940353 Category "no" probabil	0.001131133	0.22
02D			Category "NA" probability is 0.849292 Category "no" probabil	0.0009723711	0.20
02E			Category "NA" probability is 0.659901 Category "no" probabil	0.001071832	0.21
02H			Category "NA" probability is 0.817669 Category "no" probabil	0.0009118459	0.18
02J			Category "NA" probability is 0.709364 Category "no" probabil	0.0009765589	0.19
02L			Category "NA" probability is 0.841305 Category "no" probabil	0.0008935567	0.17
02P			Category "NA" probability is 0.781511 Category "no" probabil	0.000876188	0.17
03X			Category "NA" probability is 0.746532 Category "no" probabil	0.0009051868	0.18
06A			Category "NA" probability is 0.675436 Category "no" probabil	0.001010461	0.19
06H			Category "NA" probability is 0.653539 Category "no" probabil	0.001048972	0.20
06I			Category "NA" probability is 0.789731 Category "no" probabil	0.0008412559	0.16
06K			Category "NA" probability is 0.790709 Category "no" probabil	0.0008315375	0.16

Рис. 9 .Результат виконання кластеризації в вкладці Clustering.
Частина Cluster Description.

На рис. 10 зображено дані згенеровані в результаті кластеризації:

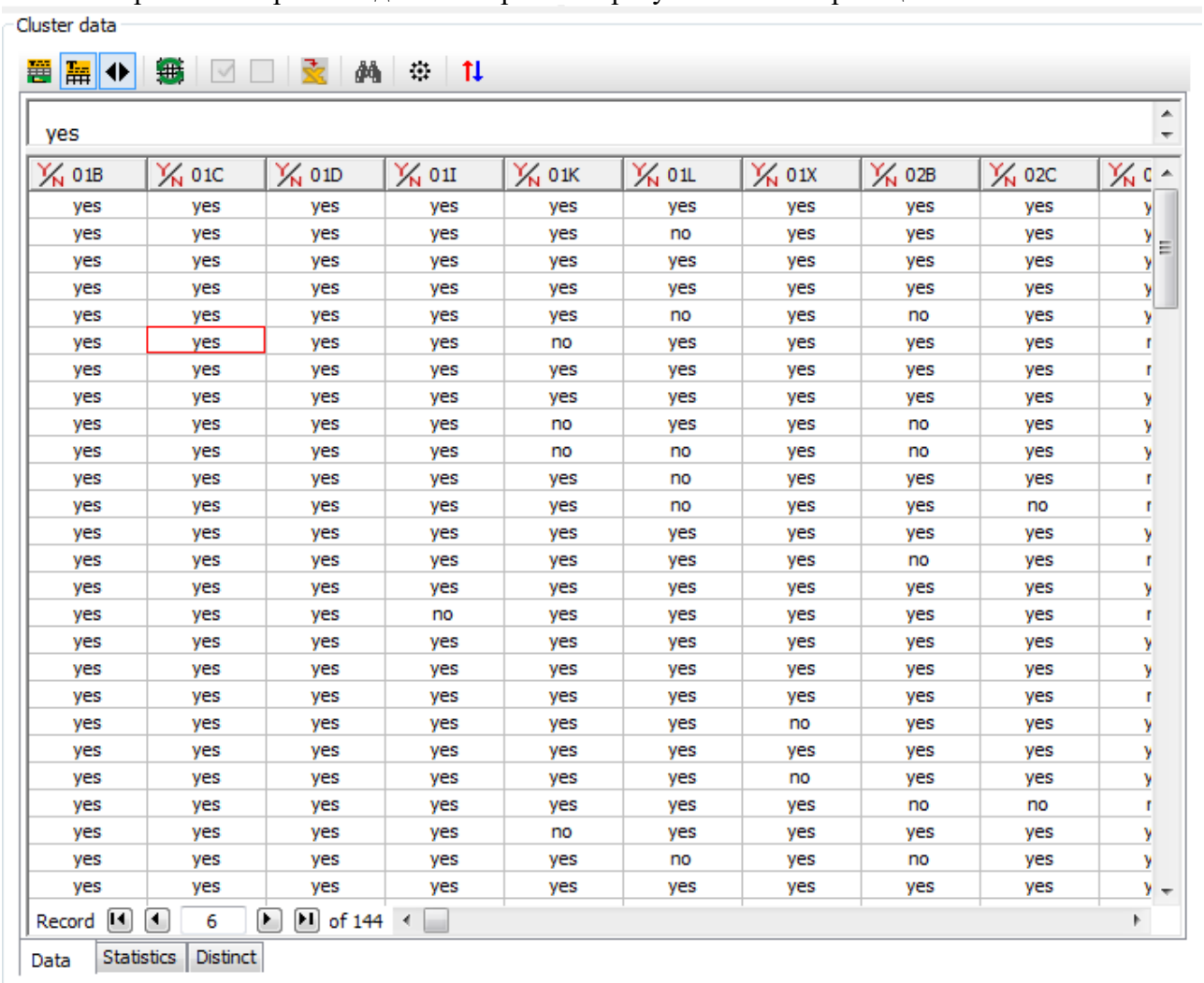


Рис. 10. Результат виконання кластеризації в вкладці Clustering.
Частина Cluster Data.

Завдання для виконання:

1. Ознайомлення із теоретичними відомостями.
2. Визначення множини характеристик, по яких будуть оцінюватися об'єкти у вибірці.
3. Обчислення значень тієї чи іншої міри схожості між об'єктами Здійснити кластеризацію даних.
4. Застосування одного з методів кластерного аналізу для створення груп схожих об'єктів.
5. Перевірка достовірності результатів кластеризації
6. Додати екранні форми результатів виконання.
7. Зробити висновки.

.

Лабораторна робота № 4

Тема : Методи генетичного пошуку

Мета роботи:

Вивчити основні методи генетичного пошуку.

Навчитися використовувати генетичні методи для розв'язку оптимізаційних задач.

Основні теоретичні відомості

У багатьох технічних задачах актуальною є проблема знаходження глобального оптимуму цільової функції в багатомірному просторі керованих змінних. Традиційні методи багатомірної оптимізації є методами локального пошуку та сильно залежать від вибору початкової точки пошуку. Для знаходження глобального оптимуму доцільно використовувати методи генетичного пошуку.

Генетичні методи засновані на аналогії з природними процесами селекції та генетичними перетвореннями, і поєднують комп'ютерні методи моделювання генетичних процесів у природних і штучних системах.

Традиційно до генетичних методів відносять генетичні алгоритми, генетичні стратегії, генетичне програмування та еволюційне програмування.

Генетичний пошук як метод оптимізації

Генетичний пошук містить у собі групу багатомірних, стохастичних, евристичних оптимізаційних методів, вперше запропонованих Д. Холландом у 1975 р. і заснованих на ідеї еволюції за допомогою природного відбору. Генетичні методи були отримані в процесі узагальнення й імітації в штучних системах таких властивостей живої природи, як природний відбір, пристосованість до змінюваних умов середовища, спадкування нащадками життєво важливих властивостей від батьків і т.ін.

Під стандартним генетичним методом розуміють метод для вирішення оптимізаційних задач вигляду:

$f(H) \rightarrow \min$,

де f – функція пристосованості (функція придатності, цільова функція, фітнесс-функція);

$H = \{0; 1\}^L$ – хромосома, що містить в закодованому вигляді параметри цільової функції;

L – кількість розрядів у хромосомі.

Генетичні методи в процесі пошуку використають деяке кодування множини параметрів замість самих параметрів, тому вони можуть ефективно застосовуватися для рішення задач оптимізації, визначених як на числових множинах, так і на кінцевих множинах довільної природи.

Аналогія генетичних методів з поняттями генетики

Основна концепція класичної генетики – *ген* (реально існуюча, незалежна, комбінуюча та розщеплююча при схрещуваннях одиниця спадковості) була введена І.Г. Менделем з метою пояснення спостережуваної статистики спадкування. Носіями генів у клітинному ядрі особини є нитковидні тіла – *хромосоми* (структурні елементи клітинного ядра біологічних організмів, що є носіями генів). У генетичних методах терміни "хромосома" і "особина" використаються як синоніми. Місце, що займає ген у хромосомі, називається *локусом*. Схематично можна уявити собі хромосому як прямолінійний відрізок, а локуси – як послідовні ділянки, на які цей відрізок розбитий. Гени приймають значення, які називаються *алелями*.

Дії генів проявляються в досить великих співтовариствах організмів, що схрещуються між собою. Такі співтовариства називають *популяціями*. Популяції характеризуються набором хромосом кожного з об'єктів, сукупність яких визначає *генофонд популяції*.

Таким чином, генетичні методи запозичили з біології понятійний апарат, ідею колективного пошуку екстремуму, способи представлення генетичної інформації, способи передачі генетичної інформації в послідовності поколінь (генетичні оператори), ідею про

переважне розмноження найбільш пристосованих особин.

Узагальнена схема роботи генетичних методів

Суть генетичного пошуку полягає в циклічній заміні однієї популяції наступною, більш пристосованою. Таким чином, популяція існує не тільки в просторі, але й у часі. Початкова популяція P_0 створюється на етапі ініціалізації генетичного пошуку.

Подальша робота генетичного методу представляє собою ітераційний процес виконання генетичних операторів відбору, схрещування й мутації. Генетичні оператори необхідні для того, щоб застосувати принципи спадковості й мінливості до популяції. Генетичні оператори мають властивість імовірності, тобто вони не обов'язково застосовуються до всіх рішень, що вносить додатковий елемент невизначеності в процес пошуку рішення.

Кожне рішення (хромосома) оцінюється мірою пристосованості. Пристосованість хромосоми визначається як обчислена цільова функція. Правила відбору прагнуть залишити тільки ті рішення, де досягається оптимум цільової функції. Найбільш пристосовані хромосоми одержують можливість відтворювати нащадків за допомогою схрещування з іншими хромосомами популяції. Це призводить до появи нових хромосом, які сполучають у собі деякі характеристики, наслідувані ними від батьків. Найменш пристосовані рішення з меншою ймовірністю зможуть відтворити нащадків, у результаті чого властивості, якими вони володіли, будуть поступово зникати з популяції в процесі еволюції.

Схрещування найбільш пристосованих хромосом приводить до того, що досліджуються найбільш перспективні ділянки простору пошуку. В остаточному підсумку популяція буде сходитися до оптимального рішення задачі.

Після схрещування іноді відбуваються мутації – спонтанні зміни в генах, які випадковим чином розкидають рішення по всьому простору пошуку.

У результаті схрещування й мутації розмір популяції збільшується. Однак для наступних перетворень необхідно скоротити число хромосом поточної популяції. Як правило, наступна популяція формується з нащадків, отриманих у поточній популяції в результаті схрещування й мутації, а також елітних хромосом, що володіють найкращою пристосованістю.

Узагальнений метод генетичного пошуку можна записати в такий спосіб.

Крок 1. Встановити лічильник ітерацій (часу): $t = 0$.

Крок 2. Згенерувати початкову популяцію хромосом $P(t)$.

Крок 3. Обчислити функцію пристосованості для всіх хромосом у популяції $f(P(t))$.

Крок 4. Перевірити умови закінчення пошуку (час, число ітерацій, значення функції пристосованості і т.ін.). Якщо критерії зупину задоволені, перейти до кроку 12.

Крок 5. Збільшити лічильник ітерацій (часу): $t = t + 1$.

Крок 6. Вибрати частину популяції (батьківські хромосоми) для схрещування P'.

Крок 7. Схрестити обрані батьківські хромосоми $P'(t)$.

Крок 8. Застосувати оператор мутації до хромосом $P'(t)$.

Крок 9. Обчислити нову функцію пристосованості популяції $f(P'(t))$.

Крок 10. Вибрати хромосоми, що вижили, виходячи з рівня пристосованості.

Крок 11. Перейти на крок 4.

Крок 12. Кінець.

У наш час запропоновано багато різних генетичних методів, і в більшості випадків вони мало схожі на наведений генетичний метод. Із цієї причини під терміном “генетичні методи” мається на увазі досить широкий клас методів, часом мало схожих один на одного.

Використання генетичного пошуку для рішення практичних задач передбачає:

– вибір методу представлення вхідних даних для генетичного пошуку (кодування параметрів, що оптимізуються);

– визначення цільової функції, що використовується для оцінки хромосом;

– вибір оператору відбору хромосом, що будуть використані для генерації нових рішень за допомогою схрещування й мутації;

– вибір методів одержання нових рішень (операторів схрещування й мутації);
– завдання параметрів пошуку, таких як кількість особин у популяції, імовірнісні характеристики генетичних операторів, максимально припустима кількість ітерацій генетичного пошуку, кількість елітних особин при використанні стратегії елітизму.

Моделі генетичного пошуку

Крім узагальненої схеми функціонування генетичного методу, розглянутої вище, використовують також інші технології генетичного пошуку. Вибір моделі генетичного методу залежить від типу розв'язуваної задачі.

Виділяють наступні методи й моделі генетичного пошуку:
– канонічні моделі (*репродуктивний план Холланда*, генетичний метод Девиса, генетичний метод Гольдберга);
– модель *Genitor (Д. Уітлі);*
– гібридні *генетичні* методи;
– модель CHC;
– генетичний метод зі змінним часом життя особин;
– мобільний генетичний метод;
– паралельні й багаторівневі генетичні методи (однопопуляційні генетичні методи, острівна модель, дрібноструктурні генетичні методи, ієрархічні гібриди);
– генетичний пошук зі зменшенням розміру популяції.

Ініціалізація та запуск генетичного пошуку

Кодування параметрів, що оптимізуються

Будь-який організм може бути представлений своїм *фенотипом*, що фактично визначає, чим є об'єкт у реальному світі, і *генотипом*, що містить всю інформацію про об'єкт на рівні хромосомного набору. При цьому кожний *ген*, тобто елемент інформації генотипу, має своє відображення у фенотипі. Таким чином, для розв’язку задач необхідно представити кожну ознаку об'єкта у формі, що підходить для використання в генетичному методі. Все подальше функціонування механізмів генетичного методу відбувається на рівні генотипу, що дозволяє обходитися без інформації про внутрішню структуру об'єкту, що й обумовлює широке застосування генетичного пошуку в самих різних задачах.

За методами представлення генів хромосоми можна умовно розділити на три групи:
1. *Бінарні хромосоми* – хромосоми, гени яких можуть приймати значення 0, або 1.
2. *Числові хромосоми* – гени можуть приймати значення в заданому інтервалі.

Числові хромосоми можна розділити на гомологічні та негомологічні.

Гомологічними називають хромосоми, що мають загальне походження, морфологічно та генетично подібні, і тому не утворюють неприпустимих рішень при застосуванні стандартних генетичних операторів. У гомологічних числових хромосомах кожний ген може приймати цілі значення в заданому інтервалі. Для різних генів можуть бути задані різні інтервали. Бінарна хромосома є гомологічною числовою хромосомою, кожний ген якої може приймати цілі значення в інтервалі [0, 1].

У *негомологічних* хромосомах гени можуть приймати значення в заданому інтервалі; при цьому інтервал однаковий для всіх генів, але в хромосомі не може бути двох генів з однаковим значенням. Для негомологічних хромосом застосовуються різні спеціальні генетичні оператори, що не створюють неприпустимих рішень. Негомологічні хромосоми, як правило, застосовуються при розв’язку задач комбінаторної оптимізації.

3. *Векторні хромосоми* – хромосоми, гени яких представляють собою вектор цілих чисел.

Ген у векторних хромосомах має властивості негомологічної хромосоми, тобто числа у векторі можуть приймати значення в заданому інтервалі, і вектор не може містити двох однакових чисел. Проте, хоча гени у векторних хромосомах негомологічні, самі векторні хромосоми є гомологічними.

Процес кодування параметрів, що оптимізуються, можна виконати в наступній послідовності кроків.

Крок 1. Визначення параметрів, що оптимізуються, – генів.
Крок 2. Вибір числа розрядів у кожному гені.
Крок 3. Вибір методу кодування.
Варто врахувати, що занадто велика довжина кодування прискорює процес збіжності всіх членів популяції до кращого знайденого рішення. Часто такий ефект є небажаним, оскільки при цьому більша частина простору пошуку залишається недослідженою. Передчасна збіжність може не привести до оптимального рішення, крім того, швидка збіжність до однієї області не гарантує виявлення декількох рівних екстремумів. До того ж застосування довгих кодувань зовсім не гарантує, що знайдене рішення буде мати необхідну точність, оскільки цього, в принципі, не гарантує сам генетичний метод.
Тому в питанні вибору оптимальної довжини кодування потрібно досягти деякого компромісного рішення – з одного боку довжина хромосоми повинна бути досить великою, щоб все-таки забезпечити швидкий пошук, з іншого боку – по можливості малою, щоб не допускати передчасної збіжності й залишити методу шанс відшукати кілька оптимальних значень.
Наведемо варіанти кодування генів у деяких задачах, розв'язуваних за допомогою генетичних методів:
– оптимізація функцій: гени – незалежні змінні;
– апроксимація: гени – параметри-константи апроксимуючих функцій;
– задача відбору інформативних ознак: гени ідентифікують значимість відповідних ним ознак (наприклад, якщо значення гену дорівнює одиниці, то відповідна йому ознака вважається інформативною);
– настроювання ваг штучної нейронної мережі: гени відповідають синоптичним вагам нейронів;
– штучне життя (Artificial Life): гени відповідають характеристикам особини (сила, швидкість, і т.ін.), також повинні бути незмінні гени, що позначають тип особини (рослина або тварина);
– задача про найкоротший шлях: гени – пункти пересування. Вся хромосома представляє собою маршрут з початкової точки в кінцеву, причому не завжди існуючий.
Невдалий вибір упорядкування та кодування бітів у хромосомі може викликати передчасну збіжність до локального оптимуму. Для подолання цього недоліку можна вибирати спосіб кодування, ґрунтуючись на додаткової інформації про задачу.
Варто відзначити, що використання різних варіантів кодування розподіляє точки в просторі пошуку по-різному. У найбільш розповсюдженій різновиді генетичного пошуку для представлення генотипу об'єкту використовуються *бітові рядки*. При цьому кожному атрибуту об'єкту у фенотипі відповідає один ген у генотипі об'єкта. Ген є бітовим рядком, найчастіше фіксованої довжини, що являє собою значення цієї ознаки.
Кількість розрядів r у гені для кодування ознаки визначається за формулою:

$$r = \mathrm{ceil}\left(\log_2 \left(\frac{w_{max} - w_{min}}{\varepsilon} \right) \right),$$

де ceil(x) – найближче більше або рівне x ціле число;
w_{max} і w_{min} – максимально й мінімально можливі значення ознаки (параметра, незалежної змінної);
ε – задана похибка визначення оптимального значення ознаки.
Розрядність хромосоми L визначається як сума розрядностей генів. У випадку, якщо задані однакові значення w_{max}, w_{min} і ε для всіх n генів, розрядність хромосоми може бути обчислена за формулою:
$L = n \cdot r$.
Після того, як обрані параметри, їх кількість та розрядність, необхідно вирішити, як безпосередньо записувати дані, тобто вибрати метод кодування. Можна використати *звичайне кодування* або *коди Грея*. Незважаючи на те, що використання кодів Грея призводить до кодування/декодування даних, вони дозволяють уникнути деяких проблем,

які з'являються в результаті звичайного кодування.
Перевага коду Грея в тому, що якщо два числа відрізняються на 1, то і їхні двійкові коди відрізняються тільки на один розряд, а у двійкових кодах не все так просто. Так, наприклад, числа 7 і 8 у бітовому поданні відрізняються в чотирьох позиціях ($7_{10}=0111_2$, $8_{10}=1000_2$), що затрудняє функціонування генетичного методу й збільшує час, необхідний для його збіжності, а в коді Грея ці числа відрізняються всього на одну позицію ($7_{10}=0100_Г$, $8_{10}=1100_Г$).
Варто відзначити, що кодувати й декодувати у коди Грея досить зручно. *Кодування/декодування з бінарного коду в код Грея* можна виконати в такий спосіб.
Крок 1. Скопіювати старший розряд кодуємого (декодуємого) числа в старший розряд декодуємого (кодуємого) числа.
Крок 2. Виконати перетворення за формулами:
– із двійкового коду в код Грея: $G[i]=\mathrm{XOR}(B[i+1],B[i])$;
– з коду Грея у двійковий: $B[i]=\mathrm{XOR}(B[i+1],G[i])$,
де $G[i]$ – i-ий розряд коду Грея;
$B[i]$– i-ий розряд бінарного коду.
Наприклад, послідовність чисел від 0 до 15 у двійковому коді: {0000, 0001 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1001 1010, 1011, 1100, 1101, 1110, 1111}, а в кодах Грея: {0000, 0001 0011, 0010, 0110, 0111, 0101, 0100, 1100, 1101 1111, 1110, 1010, 1011, 1001, 1000}.
Кодування ознак, яким відповідають числа із плаваючою комою.
Найпростіший спосіб кодування – використання бітового представлення. Однак такий варіант має ті ж недоліки, що й при кодуванні цілих чисел. Тому на практиці застосовується наступна послідовність дій.
Крок 1. Розбивається весь інтервал допустимих значень ознаки на ділянки з необхідною точністю.
Крок 2. Приймається значення гена як ціле число, що визначає номер інтервалу (використовуючи код Грея).
Крок 3. В якості значення параметра приймається число, що є серединою цього інтервалу.

Завдання цільової функції

Цільова функція – це функція, оптимум якої необхідно знайти. Генетичний метод вимагає, щоб хромосоми оцінювалися за допомогою цільової функції (фітнесс-функцій, функції пристосованості, функції оцінки) задачі.
Наприклад, для задачі апроксимації, цільовою функцією може бути середнє відхилення значень вихідного параметру, розрахованих за утвореною моделі, від його реальних значень.
Можна відзначити, що обчислення фітнесс-функції – один з найбільш важливих етапів генетичного пошуку.
Тому при виборі цільової функції потрібно враховувати наступне.
1. Функція пристосованості повинна бути адекватна задачі. Це означає, що для успішного пошуку необхідно, щоб розподіл значень фітнесс-функціх збігався з розподілом реальної якості рішень (не завжди “якість” рішення еквівалентна його оцінці за фітнесс-функцією).
2. Фітнесс-функція повинна мати рельєф. Крім того, рельєф повинен бути різноманітним. Це означає, що генетичний метод має мало шансів на успіх, якщо на поверхні фітнесс-функції є величезні “плоскі” ділянки, тому що це приводить до того, що більшість рішень (хромосом) у популяції при різних генотипах не будуть відрізнятися фенотипом. Тобто, незважаючи на те, що рішення розрізняються, вони мають однакову оцінку, а значить метод не має можливості вибрати краще рішення та вибрати напрямок подальшого розвитку.
3. Фітнесс-функція повинна вимагати мінімум ресурсів, тому що її обчислення є найбільш часто виконуваним етапом методу, і тому складність обчислення фітнесс-функції має

істотний вплив на швидкість роботи методу.
4. У випадку, якщо цільова функція містить ділянки, що представляють собою так зване "вузьке горло" (різкий стрибок або спад), необхідно врахувати, що генетичний пошук може не знайти глобального екстремуму, що розташований у вузькому горлі. Для підвищення якості генетичного пошуку при такій цільовій функції можна рівномірно формувати початкову популяцію на всьому інтервалі припустимих значень змінних.

Ініціалізація

Стандартні генетичні методи починають свою роботу з ініціалізації, тобто формування *початкової популяції* P_0 – кінцевого набору допустимих рішень задачі:
$P_0 = \{H_1, H_2, \ldots, H_N\}$,
де N – розмір популяції;
$H_j = \{h_{1j}, h_{2j}, \ldots, h_{Lj}\}$ – хромосома, що складається з L генів;
$\min_i \le h_{ij} \le \max_i$, $\min_i$ і $\max_i$ – мінімальне й максимальне значення i-го параметра в розв'язуваній за допомогою генетичного методу задачі.
Ці рішення можуть бути обрані випадковим образом або введені користувачем. Вибір початкової популяції не має значення для збіжності процесу в асимптотиці, однак, формування гарної початкової популяції (наприклад, із множини локальних оптимумів) може помітно скоротити час досягнення глобального оптимуму. Таким чином, при наявності необхідної інформації завдання початкової популяції користувачем є кращим.
Найчастіше розмір початкової популяції вибирається в інтервалі 20–100 особин.
Стратегія створення початкової популяції може бути різною. Відомі наступні стратегії.
1. *Стратегія "ковдри"* – вихідна множина містить всі можливі варіанти рішень. Стратегія "ковдри" має наступні недоліки:
– у багатьох випадках неможливо здійснити повний перебір;
– з погляду адаптивного розвитку не представляє цінності, тому що часто вже в першому поколінні рішень відшукується оптимальне рішення.
2. *Стратегія "фокусування"* – стартова множина рішень включає різновиди одного рішення.
Стратегія "фокусування" застосовується в тих випадках, коли є припущення, що деяке рішення є різновидом відомого субоптимального. Тоді шляхом поступових незначних змін існуючого рішення можна одержати більш якісне субоптимальне рішення.
Для більшості задач оптимізації неприйнятні стратегії "ковдри" (внаслідок проблематичності повного перебору) і фокусування (відсутня чітка залежність якості рішення від параметрів рішення).
3. *Стратегія "дробовика"* – генерується досить велика множина рішень. Дана стратегія може бути реалізована одним з трьох способів.
3.1. Рівномірне формування початкової популяції.
3.2. Випадкове формування початкової популяції.
3.3. Комплементарне формування початкової популяції, яке виконується за два кроки.
Крок 1. Сформувати випадковим чином першу половину початкової популяції.
Крок 2. Сформувати другу половину початкової популяції шляхом додавання хромосом, протилежних (одиниці заміняються нулями) хромосомам в першій половині популяції.

Відбір

Оператор відбору (вибору, селекції) представляє собою оператор, що на основі значення цільової функції вибирає хромосоми таким чином, щоб з ненульовою ймовірністю будь-який елемент популяції міг би бути обраний у якості одного з батьків при схрещуванні.
Найпоширенішими є наступні оператори відбору:
– пропорційний відбір (пропорційно-імовірнісний відбір);
– відбір ранжируванням;
– турнірний відбір;

– відбір з використанням порогу.

Пропорційний відбір

Даний вид відбору складається з наступної послідовності кроків.

Крок 1. Обчислити пристосованість кожної особини f_j.

Крок 2. Знайти середню пристосованість у популяції $f_{\text{ср}}$ як середнє арифметичне значень пристосованості всіх особин:

$$f_{cp} = \frac{1}{N}\sum_{j=1}^{N} f_j \ .$$

Крок 3. Для кожної особини обчислити відношення $P_s(j) = \dfrac{f_j}{f_{cp}}$.

Крок 4. Залежно від величини $P_s(j)$ сформувати масив особин, допущених до схрещування.

Формування масиву допущених до схрещування особин (крок 4) можна здійснити двома шляхами:

Перший шлях (стохастичний залишковий відбір): якщо $P_s(j) > 1$, то особина вважається добре пристосованою та допускається до схрещування.

Наприклад, якщо дріб $P_s(j) = 2{,}36$, то дана особина має подвійний шанс на схрещування й буде мати ймовірність рівну 0,36 третього схрещування. Якщо ж пристосованість дорівнює 0,54, то особина візьме участь у єдиному схрещуванні з імовірністю 0,54.

Другий шлях: після знаходження відносини $P_s(j)$ відбувається відбір (із заміщенням) всіх N особин для подальшої генетичної обробки, відповідно до величини $P_s(j)$.

Найпростіший пропорційний відбір – *рулетка* – відбирає особини за допомогою N запусків рулетки.

Колесо рулетки містить по одному сектору для кожного члена популяції. Розмір j-го сектору пропорційний відповідній величині $P_s(j)$. Особина одержує можливість створення нащадків, якщо випадково згенероване число в межах від 0 до 2π попадає в сектор, що відповідає цій особини. При такому відборі члени популяції з більш високою пристосованістю з більшою ймовірністю будуть частіше вибиратися, чим особини з низькою пристосованістю.

При реалізації відбору рулеткою доцільно замінити колесо рулетки інтервалом [0;1] у зв'язку з тим, що в такому випадку немає необхідності обчислювати ширину кожного сектора – у цьому випадку кожній особині ставиться у відповідність напівінтервал $[x_{j-1}; x_j)$, де $x_{j-1} - x_j = P_s(j)$, а $x_0 = 0$ (при цьому $x_N = 1$). У наступне покоління переходить особина з номером j, де j: $x_{\text{r nd}} \in [x_{j-1}; x_j)$, а число $x_{\text{r nd}}$ повертається щораз випадковою функцією з рівномірним розподілом щільності ймовірності на відрізку [0;1].

Схема рулетки може давати дуже великі помилки, у тому розумінні, що кінцеве число нащадків даної особини може сильно відрізнятися від очікуваного. Кінцеве число наближається до очікуваного тільки в популяціях дуже більших розмірів.

Відбір ранжируванням

Відбір ранжируванням виконується за 4 кроки.

Крок 1. Обчислити пристосованість кожної особини f_j.

Крок 2. Відсортувати (ранжирувати) популяцію по зростанню пристосованості особин.

Крок 3. Для кожної особини обчислити величину $P_s(j)$. Для цього використати один із двох видів ранжирування.

а) лінійне ранжирування: $P_s(j) = \dfrac{1}{N}\left(\eta_{\max} - (\eta_{\max} - \eta_{\min})\dfrac{j-1}{N-1}\right)$, де $\eta_{\max} \in [1; 2]$, $\eta_{\min} = 2 - \eta_{\max}$.

б) рівномірне ранжирування: $P_s(j) = \begin{cases} \dfrac{1}{\mu}, 1 \le i \le \mu, \\ 0, \ \mu < i \le N, \end{cases}$

де μ – деяке фіксоване число перших членів популяції.
Крок 4. Залежно від величини $P_s(j)$ відібрати певну частину особин для схрещування.

Турнірний відбір

Турнірний відбір реалізує k турнірів, щоб вибрати k особин. Кожний турнір складається із двох етапів.
Етап 1. Вибір m елементів з популяції.
Етап 2. Вибір кращої особини серед особин, відібраних на попередньому етапі.
Розмір групи особин, що відбираються для турніру, часто дорівнює 2. У цьому випадку говорять про *парний турнір*. Взагалі ж m називається *чисельністю турніру*.
Турнірний відбір має певні переваги перед пропорційним, тому що не втрачає своєї вибірковості, коли в ході еволюції всі елементи популяції стають приблизно рівними за значенням цільової функції.

Відбір з використанням порогу

Відбір з використанням порогу (відбір усіканням) виконується в наступній послідовності.
Крок 1. Обчислити пристосованість кожної особини f_j.
Крок 2. Відсортувати популяцію по зростанню пристосованості особин.
Крок 3. Задати поріг $\Pi \in [0;1]$. Поріг визначає, яка частка особин, починаючи з найпершої (самої пристосованої), буде брати участь у відборі. В принципі, поріг можна задати й числом, більшим за одиницю, тоді він буде просто дорівнює числу особин з поточної популяції, допущених до відбору.
Крок 4. Серед особин, що потрапили під значення порога, випадковим образом N раз вибирати саму везучу й записувати її в проміжний масив, з якого потім вибираються особини безпосередньо для схрещування.
Через те, що в цій стратегії використається відсортована популяція, час її роботи може бути більшим для популяцій великого розміру й залежати також від алгоритму сортування.

Схрещування

У теорії еволюції важливу роль відіграє те, яким чином ознаки батьків передаються нащадкам. У генетичних методах за передачу ознак батьків нащадкам відповідає оператор схрещування (кроссинговер, кроссовер, рекомбінація). Цей оператор моделює процес схрещування особин і визначає передачу ознак батьків нащадкам.
Метою оператору схрещування є породження з наявної множини рішень нової, у якій кожна хромосома буде нащадком деяких двох елементів попередньої популяції, тобто нести в собі частково інформацію кожного батька. Допускається ситуація, коли обидва батьки представлені одним елементом популяції.

Вибір батьківської пари

Вибираючи щораз для схрещування найбільш пристосовані особини, можна з певним ступенем впевненості стверджувати, що нащадки будуть або не набагато гіршими, ніж батьки, або кращими за них.
Існує кілька способів вибору батьківської пари.
Випадковий вибір батьківської пари (*панміксія*) – це найпростіший підхід, коли обидві особини, які утворять батьківську пару, випадковим чином вибираються із всієї популяції, причому будь-яка особина може стати членом декількох пар. Незважаючи на простоту, такий підхід універсальний для розв'язку різних класів задач. Однак він досить критичний до чисельності популяції, оскільки ефективність методу, що реалізує такий підхід, знижується з ростом чисельності популяції.
Крок 1. Для вибору пари батьків задається ймовірність схрещування P_c.
Крок 2. Довільним чином нумеруються всі представники вихідної популяції.
Крок 3. Вибір першого батька: починаючи з першого рішення, проглядається популяція

доти, поки випадково обране число з інтервалу [0, 1] не буде меншим, ніж P_c. Елемент, для якого виконується така умова, стає першим батьком.
Крок 4. Відбувається перегляд популяції, починаючи з наступному після першого батька рішення, поки знову випадково обране число не буде меншим, ніж P_c. Елемент, для якого виконується така умова, стає другим батьком.
Описаним способом складаються пари доти, поки не вибереться потрібна кількість пар батьків.
Конкретне значення P_c залежить від розв'язуваної задачі, і в загальному випадку лежить в інтервалі [0,6; 0,99].
Незважаючи на простоту, такий підхід універсальний для розв'язування різних класів задач. Однак він досить критичний до чисельності популяції, оскільки ефективність методу, що реалізує такий підхід, знижується з ростом чисельності популяції.
Інший метод випадкового вибору батьківської пари може бути представлений у вигляді наступної послідовності кроків.
Крок 1. Розбити популяцію випадковим чином на два масиви (підпопуляції) одного розміру.
Крок 2. Відсортувати кожну підпопуляцію.
Крок 3. Сформувати пари для схрещування з особин, що мають однаковий ранг (номер) у підпопуляціях.
Крок 4. Допустити до схрещування пари, для яких випадково згенероване в інтервалі [0;1] число буде перевищувати задану ймовірність схрещування.
Селективний спосіб вибору особин у батьківську пару полягає в тому, що батьками можуть стати тільки ті особини, значення пристосованості яких не менше середнього значення пристосованості по популяції, при рівній імовірності таких кандидатів утворити батьківську пару.
Такий підхід забезпечує більш швидку збіжність генетичного пошуку. Однак через швидку збіжність селективний вибір батьківської пари не підходить тоді, коли ставиться задача визначення декількох екстремумів, оскільки для таких задач метод, як правило, швидко збігається до одного з рішень.
Крім того, для деякого класу задач зі складним ландшафтом фітнесс-функції швидка збіжність може перетворитися в передчасну збіжність до квазіоптимального розв'язку. Цей недолік може бути частково компенсований використанням відповідного механізму відбору, який би "гальмував" занадто швидку збіжність методу.
Інші два способи формування батьківської пари – це *інбридинг* та *аутбридинг*. Обоє ці методи побудовані на формуванні пари на основі близького й далекого "споріднення", відповідно. Під "спорідненням" тут розуміється відстань між членами популяції як у сенсі евклідової (геометричної) відстані особин у просторі параметрів (для фенотипів), так і у сенсі відстані Хеммінгу між хромосомними наборами особин (для генотипів).
Евклідова відстань $R^{(jk)}$ між j-ою та k-ою особинами популяції визначається за формулою:

$$R^{(jk)} = \sqrt{\sum_{i=1}^{p}\left(x_i^{(j)} - x_i^{(k)}\right)^2},$$

де p – кількість параметрів (генів) особини;
$x_i^{(j)}$ – i-ий параметр у незакодованому вигляді j-ої особини.
Відстань Хеммінгу $H^{(jk)}$ між j-ою та k-ою особинами популяції визначається як кількість різних бітів в однакових позиціях j-ої та k-ої хромосом.
Інбридинг складається з двох етапів:
1. Перший член пари вибирається випадково.
2. Другим батьком з більшою ймовірністю буде максимально близька до першого особина.
Один з варіантів процедури інбридингу може бути реалізований у такий спосіб.
Крок 1. Вибрати випадковим чином першого батька.
Крок 2. Вибрати з поточної популяції випадковим чином групу з C хромосом (C = 1%–

15% від розміру популяції).
Крок 3. Розрахувати *Евклідову* відстань від хромосоми, отриманої на першому кроці, до кожної із *C* відібраних на другому кроці хромосом.
Крок 4. В якості другого батька вибрати найближчу до першого батька хромосому.
Аутбридинг формує батьківські пари з максимально далеких особин.
Використання генетичних інбридингу й аутбридингу є ефективним для багато екстремальних задач. Однак два цих способи по-різному впливають на поводження генетичного методу. Інбридинг можна охарактеризувати властивістю концентрації пошуку в локальних вузлах, що фактично призводить до розбиття популяції на окремі локальні групи навколо підозрілих на екстремум ділянок ландшафту. Аутбридинг, навпаки, спрямований на попередження збіжності методу до вже знайдених рішень, змушуючи метод переглядати нові, недосліджені області.

Оператори схрещування

При ***n-точковому*** схрещуванні:
1. Випадково обираються *n* точок розриву, що приводить до розбиття вихідних векторів на *n* + 1 частин різної довжини.
2. Обмінюються у вихідних хромосомах ділянки з парними номерами, а ділянки з непарними залишаються без змін:
n-точкове схрещування може застосовуватися для бінарних, векторних і гомологічних числових хромосом.
Класичним варіантом такого схрещування є одноточечне схрещування, при якому:
1. Випадковим чином визначається точка в середині хромосоми. Ця точка називається точкою розриву (точкою *схрещування,* crossover point).
2. В обраній точці обидві хромосоми діляться на дві частини й обмінюються ними. У результаті утворюються два нащадки.
Даний тип схрещування називається одноточечним, тому що при ньому батьківські хромосоми розрізаються тільки в одній випадковій точці.
При двохточечному схрещуванні в хромосомі випадково вибираються вже дві точки схрещування. Ліву точку будемо вважати першою, а праву – другою. Перший нащадок формується із частин першого батька, розташованих лівіше від першої точки схрещування й правіше від другої точки, і частини другого батька, розташованої між першою й другою точками схрещування. Другий нащадок формується із лівої та правої частин другого батька й центральної частини першого батька.
Обчислювальні експерименти показали, що навіть для простих функцій не можна говорити про перевагу того або іншого оператора. Більше того, використання механізму випадкового вибору одно- або двохточечного схрещування для кожної конкретної батьківської пари часом виявляється більше ефективним, ніж детермінований підхід до його вибору, оскільки досить важко апріорно визначити, який із двох операторів більше підходить для кожного конкретного виду функції пристосованості.
Однорідне схрещування (uniform crossover) генерує нащадка шляхом випадкової передачі йому генетичної інформації від батьків. Генерація нащадка виконується в такий спосіб.
Крок 1. Встановити лічильник бітів (генів) нащадка: $j = 1$.
Крок 2. Визначити хромосому з батьківської пари, що передасть значення свого *j*-го гена нащадкові: $n = \text{round}(\text{rand}[0;1])$, де rand[0;1] – випадково згенероване число в інтервалі [0;1]; round(*A*) – округлене значення числа *A*.
Крок 3. Виконати: $h_{jn} = h_{jn}$, де h_{jn} – значення *j*-го гена нащадка, h_{jn} – значення *j*-го гена *n*-го батька.
Крок 4. Виконати: $j = j + 1$.
Крок 5. Якщо $j > L$, де *L* – довжина хромосоми, тоді виконати перехід на крок 6. У противному випадку перейти на крок 2.
Крок 6. Кінець.
Рівномірне схрещування може застосовуватися для бінарних, гомологічних числових і

векторних хромосом. Таке схрещування зручно застосовувати в тому випадку, коли вже отримані індивіди з гарними спадкоємними ознаками, і їх необхідно закріпити в поточній популяції. Рівномірне схрещування виконується в наступній послідовності кроків.

Крок 1. Випадковим чином задається маска схрещування, що представляє собою рядок з нулів та одиниць, довжини, що дорівнює довжині хромосом.

Крок 2. Формування першого нащадка.

Одиниця на конкретній позиції в масці схрещування означає, що елемент, що знаходиться на тому ж місці в першого батька, необхідно помістити на це місце в першій дитині. Нуль на цій позиції в масці схрещування означає, що елемент, що знаходиться на тому ж місці в другого батька, необхідно помістити на це місце першій дитині.

Крок 3. Формування другого нащадка.

Якщо першого батька вважати другим, а другого – першим, то аналогічно кроку 2 можна одержати другого нащадка.

Варто відзначити, що маска схрещування може бути одна для всіх хромосом, або своя для кожної пари батьків.

При ***порівняльному схрещуванні*** в хромосомах батьків порівнюються всі біти. Якщо на однакових позиціях обох батьків знаходяться однакові біти (0 і 0 або 1 і 1), то нащадкам присвоюються ті ж значення відповідних бітів. Якщо на однакових позиціях батьків розташовані гени із різними значеннями, тоді значення відповідних генів нащадків визначають за допомогою генератору випадкових чисел.

Арифметичне (диференціальне) схрещування є найбільш вдалим для пошуку оптимуму функції багатьох дійсних змінних.

Нехай H_1 і H_2 – це два індивідууми в популяції, тобто дійсні вектори, від яких залежить цільова функція. Тоді нащадок $H_п$ обчислюється за формулою $H_п = H_1 + k \cdot (H_1 - H_2)$, де k – це деякий дійсний коефіцієнт (який може залежати від $||H_1 - H_2||$ – відстані між векторами). У цій моделі мутація – це додавання до індивідуума випадкового вектора малої довжини. Якщо вихідна функція неперервна, то ця модель працює добре, а якщо вона ще й гладка, те – відмінно.

При ***діагональному схрещуванні*** для схрещування R батьків випадковим чином вибирається $(R - 1)$ однакових точок схрещування в кожному з них. R нащадків отримують шляхом комбінування відповідних елементів батьків по діагоналі.

Крок 1. Вибрати випадковим чином R батьківських хромосом для схрещування.

Крок 2. Вибрати випадковим чином $(R - 1)$ точок схрещування в хромосомах, розділивши тим самим кожну з них на R сегментів.

Крок 3. Встановити: $r = 1$.

Крок 4. Сформувати r-го нащадка.

Крок 4.1 Встановити: $k = 1$.

Крок 4.2. Сформувати k-ий сегмент r-го нащадка, взявши g-ий сегмент із k-ої хромосоми-батька, де

$$g = \begin{cases} k + r - 1, & \text{якщо } k + r - 1 \le R, \\ k + r - 1 - R, & \text{якщо } k + r - 1 > R. \end{cases}$$

Крок 4.3. Виконати: $k = k + 1$.

Крок 4.4. Якщо $k \le R$, виконати перехід до кроку 4.2.

Крок 5. Виконати: $r = r + 1$.

Крок 6. Якщо $r \le R$, виконати перехід до кроку 4.

Крок 7. Кінець.

Слід зазначити, що різні типи схрещування мають загальну позитивну властивість: вони контролюють баланс між подальшим використанням уже знайдених гарних підобластей простору пошуку та дослідженням нових підобластей. Це досягається за рахунок неруйнування загальних блоків усередині хромосом-батьків і одночасного дослідження нових областей в результаті обміну частинами хромосом.

Спільне використання операторів відбору та схрещування призводить до того, що області простору з кращою в середньому оптимальністю, вміщують більше членів популяції, чим інші. Тому, оператор схрещування є найбільш критичним із всіх операторів генетичних методів з погляду одержання глобальних результатів.

У малих популяціях краще застосовувати більш руйнівні варіанти схрещування (багатоточечне та однорідне), а в великих популяціях краще працює двохточечне.

Мутація

Оператор мутації полягає в зміні генів у випадково обраних позиціях. На відміну від операторів відбору та схрещування, які використовуються для поліпшення структури хромосом, метою оператора мутації є диверсифікація, тобто підвищення розмаїтості пошуку й введення нових хромосом у популяцію для того, щоб більш повно досліджувати простір пошуку. Оскільки число членів популяції P звичайно вибирається значно меншим у порівнянні із загальним числом (2^L) можливих хромосом у просторі пошуку, то в силу цього вдається досліджувати лише його частину. Отже, мутація ініціює розмаїтість у популяції, дозволяючи переглядати більше рішень у просторі пошуку й виходити в такий спосіб з локальні екстремумів в процесі пошуку.

В результаті виконання мутації з ненульовою ймовірністю чергове рішення може перейти в будь-яке інше рішення.

Вибір хромосом для мутації відбувається в такий спосіб.

Крок 1. Нумеруються довільним чином всі хромосоми H_j вихідної популяції.

Крок 2. Починаючи з першої хромосоми, проглядається вся популяція, при цьому кожній хромосомі H_j ставляться у відповідність випадкові числа x_j з інтервалу [0;1).

Крок 3. Якщо число x_j виявляється меншим за ймовірність мутації $P_м$, то поточна хромосома H_j піддається мутації.

Серед рекомендацій з вибору ймовірності мутації нерідко можна зустріти варіанти $1/L$ або $1/N$, де L – довжина хромосоми, N – розмір популяції. Імовірність мутації значно менше ймовірності схрещування й рідко перевищує 1%.

Необхідно відзначити, що оператор мутації є основним пошуковим оператором і відомі такі методи, що не використають інших операторів крім мутації.

Проста мутація

Проста мутація використовується для бінарних, гомологічних числових і векторних хромосом. Суть її полягає у внутригенній мутації. При цьому в хромосомі випадковим чином вибирається ген, а потім відбувається його випадкова зміна.

Алгоритм простої мутації для бінарних і гомологічних числових хромосом може складатися з наступних кроків.

Крок 1. Скопіювати батьківську хромосому в хромосому- нащадка.

Крок 2. Вибрати випадковим чином ген для мутації.

Крок 3. У заданому інтервалі припустимих значень гена вибрати нове значення гена, що не дорівнює поточному.

Для ***векторних хромосом*** проста мутація відбувається шляхом внесення змін у порядок елементів усередині обраного гена.

Крок 1. Скопіювати батьківську хромосому в хромосому- нащадка.

Крок 2. Вибрати випадковим чином ген для мутації.

Крок 3. Вибрати випадковим чином точку мутації усередині мутуючого гена.

Крок 4. Зробити обмін значеннями між розрядами мутуючого гена, що перебувають безпосередньо ліворуч і праворуч від точки мутації.

Відомий також модифікований оператор простої мутації, що називається ***точковою мутацією***, який відрізняється тим, що в хромосомі мутує не один, а кілька генів із заданою ймовірністю. Такий оператор використовується для бінарних, гомологічних числових і векторних хромосом. Алгоритм даного оператору наведений нижче.

Крок 1. Скопіювати батьківську хромосому в хромосому- нащадка.

Крок 2. Встановити $i = 1$.

Крок 3. Випадковим чином згенерувати число хі з інтервалу [0;1).

Крок 4. Якщо число x_i виявляється меншим ймовірності мутації гену $P_{мг}$, то виконати мутацію гена h_i.

Крок 5. Встановити $i = i + 1$.

Крок 6. Якщо $i < (L + 1)$, де L – довжина хромосоми, то виконати перехід на крок 3.

Крок 7. Кінець.

Крім того, відома множина різних варіантів операторів мутації. Більшість із них використовують комбінації наступних ідей.

1. Так само як і схрещування, мутація може проводитися не тільки по одній випадковій точці. Можна вибирати деяку кількість точок у хромосомі для інверсії, причому їхнє число також може бути випадковим.

2. Піддається мутації відразу деяка група послідовних генів.

3. Спільне застосування випадкової мутації та часткової перебудови рішення алгоритмами локального пошуку. Застосування локального спуску дозволяє генетичному методу зосередитися тільки на локальних оптимумах. Множина локальних оптимумів може виявитися експоненціально більшим і на перший погляд здається, що такий варіант методу не буде мати великих переваг. Однак експериментальні дослідження розподілу локальних оптимумів свідчать про високу концентрацію їх у безпосередній близькості від глобального оптимуму. Це спостереження відоме як гіпотеза про існування "великої долини" для задач на мінімум або "центрального гірського масиву" для задач на максимум.

Мутація гомологічних числових хромосом

Такі види мутацій полягають в зміні обраного для мутації гена h_{ji} (або всієї хромосоми H_j) на деяку величину Δh_{ij}, розраховану за певними методами:

$h_{ij}' = h_{ij} + \Delta h_{ij}$,

де h_{ij} – ген до мутації; h_{ij}' – ген після мутації

1. ***Нерівномірна (non-uniform) мутація*** до обраного для мутації i-го гену h_{ji} хромосоми H_j застосовується за формулою:

$$h_{ji}' = \begin{cases} h_{ji} + \Delta(t, \max_i - h_{ji}) & \text{з ймовірністю } 0{,}5, \\ h_{ji} - \Delta(t, h_{ji} - \min_i) & \text{з ймовірністю } 0{,}5, \end{cases}$$

де $\Delta(t, y) = y \cdot \left(1 - r^{(1-t/T)^k}\right)$;

r = rand[0; 1] – випадково згенероване число в інтервалі [0; 1];

t – номер поточної ітерації;

T – максимальна кількість ітерацій;

k – параметр, що визначає ступінь однорідності (рівномірності);

$\min_i$ і $\max_i$ – мінімальне й максимальне значення i-го параметру в розв'язуваній за допомогою генетичного методу задачі.

Крім того, нерівномірна мутація i-го гену j-ої хромосоми h_{ji} може бути виконана за формулою:

$$h_{ji}' = \begin{cases} h_{ji} + \Delta(t, \max_i - h_{ji}), & \text{якщо } f(H_j + \Delta(t, \max_i - h_{ji})) \geq f(H_j - \Delta(t, h_{ji} - \min_i)), \\ h_{ji} - \Delta(t, h_{ji} - \min_i), & \text{якщо } f(H_j + \Delta(t, \max_i - h_{ji})) < f(H_j - \Delta(t, h_{ji} - \min_i)), \end{cases}$$

де $\Delta(t, y) = y \cdot w_i(t)$; $w_i(t)$ – коефіцієнт, що залежить від відношення t/T.

Наприклад, коефіцієнт $w_i(t)$ може бути заданий формулою:

$$w_i(t) = r\left(1 - \frac{t}{T}\right)^{k_i},$$

де r = rand[0; 1] – випадково згенероване число в інтервалі [0; 1];

$k_i > 0$ – параметр, що задає користувач.

2. ***Випадкова мутація*** обраного гена h_{ji} полягає в зміні його значення на величину Δh_{ij}, розраховану за формулою:

$\Delta h_{ij} = \text{rand}[\min_i \cdot r \cdot q(t); \max_i \cdot r \cdot q(t)]$,

де rand[a; b] – випадково згенероване число в інтервалі [a; b];

$\min_i$ і $\max_i$ – мінімальне й максимальне значення i-го гену;

r = rand[0; 1] – випадково згенероване число в інтервалі [0; 1];

$q(t) = \dfrac{\ln T - \ln t}{\ln T}$.

3. ***Гауссовська (нормальна) мутація*** до обраного для мутації i-го гену j-ої хромосоми h_{ji}

застосовується за формулою:
$h_{ij}' = h_{ij} + \varepsilon$,
де ε – випадкове число, отримане за нормальним розподілом (Коші, або будь-якому іншому розподілу) з нульовим середнім і $\sigma_i = \frac{T-t}{T} \cdot \frac{\max_i - \min_i}{3}$.
Число ε може бути додане до одного гена. Можливий варіант додавання випадкового вектора до всієї хромосоми.
4. ***Мутація*** обраного гена h_{ij} ***на основі квадратичної апроксимації***.
Крок 1. Обчислити значення фітнесс-функції при $h_{ij} + \Delta h_{ij}$ і при $h_{ij} - \Delta h_{ij}$: $f(h_{ij} + \Delta h_{ij})$ і $f(h_{ij} - \Delta h_{ij})$. Значення Δh_{ij} можуть бути обчислені за формулами знаходження Δ і ε, аналогічними нерівномірній та Гауссовській мутації.
Крок 2. Апроксимувати точки h_{ij}, $h_{ij} + \Delta h_{ij}$ і $h_{ij} - \Delta h_{ij}$ у параболу.
Крок 3. Знайти мінімальне значення отриманої кривій $f_{\text{параб min}}$ і відповідне значення точки в просторі ознак, що відповідає мінімальному значенню параболи $h_{ij\,\min}$.
Крок 4. Присвоїти: $h_{ij} = h_{ij\,\min}$.
Важливо відзначити, що описані оператори мутації можуть застосовуватися й для бінарних хромосом, попередньо перетворених до реальних числових значень із погляду розв'язуваної задачі. Після застосування описаних вище операторів мутації для таких хромосом їх необхідно знову перетворити до бінарного вигляду, застосувавши використовуваний метод кодування.

Формування нового покоління

Після схрещування та мутації необхідно створити нову популяцію. Види операторів формування нового покоління (репродукції, редукції) практично збігаються з видами операторів відбору батьків, що передбачають формування проміжного масиву особин, допущених до схрещування.
При формуванні нового покоління необхідно вирішити проблему: які з нових особин увійдуть у наступне покоління, а які – ні. Для цього застосовують один із двох способів.
Перший спосіб полягає в тому, що нові особини (нащадки) займають місця своїх батьків. Після чого починається наступний етап, у якому нащадки оцінюються, відбираються, дають потомство й поступаються місцем своїм нащадкам.
Недоліком даного способу є можливість втрати найбільш пристосованої особини попереднього покоління. Одним зі способів вирішення даної проблеми може бути використання *принципу "елітизму"*, що полягає в тому, що особини з найбільшою пристосованістю гарантовано переходять у нову популяцію. Їхнє число може бути від 1 і більше. Кількість елітних особин *KI*, які гарантовано перейдуть у наступну популяцію, може бути обчислена за формулою:
$KI = (1 - S_O) \cdot N$,
де S_O – ступінь відновлення популяції, що перебуває в діапазоні [0,95;1,0]; N – розмір популяції.
Використання принципу "елітизму" дозволяє прискорити збіжність генетичного методу. Недолік використання даної стратегії в тому, що підвищується ймовірність попадання методу в локальний мінімум.
Другий спосіб заснований на тому, що створюється проміжна популяція, яка містить у собі як батьків, так і їхніх нащадків. Члени цієї популяції оцінюються, а потім з них вибираються N найкращих, які й увійдуть у наступне покоління.
Другий варіант є більше оптимальним, але він вимагає сортування масиву розміром $2N$.
Другий варіант формування нового покоління можна реалізувати за допомогою *принципу витиснення*, що носить двохкритеріальний характер – те, чи буде особина з репродукційної групи заноситися в популяцію нового покоління, визначається не тільки величиною її пристосованості, але й тим, чи є вже у популяції наступного покоління особина з аналогічним хромосомним набором. Із всіх особин з однаковими генотипами перевага спочатку віддається тим, чия пристосованість вище. Таким чином, досягаються дві мети: по-перше, не губляться кращі знайдені рішення з різними хромосомними наборами, а по-друге, у популяції постійно підтримується достатня генетична розмаїтість.

Витиснення в цьому випадку формує нову популяцію скоріше з далеко розташованих особин, замість особин, що групуються біля поточного знайденого рішення.

Критерії зупинення

Одним з важливих етапів у генетичних методах є визначення критеріїв зупину. Очевидно, еволюція – нескінченний процес, у ході якого пристосованість особин поступово підвищується. Примусово зупинивши цей процес через досить довгий час після його початку й обравши найбільш пристосовану особину в поточному поколінні, можна одержати не абсолютно точну, але близьку до оптимальної відповідь.

Як правило, в якості критерію зупинення застосовується *обмеження на максимальну кількість ітерацій* функціонування методу (тобто обмеження на кількість поколінь). Кількість популяцій може бути будь-якої, але частіше за все обирають 50-100 популяцій. Якщо в якості критерію зупину обирається максимальна кількість ітерацій, то задається кількість ітерацій T, в результаті чого цикл генетичного пошуку виконується T раз.

Зупинення роботи генетичного методу може відбутися також у випадку, якщо *популяція вироджується*, тобто якщо практично немає розмаїтості в генах особин популяції. Виродження популяції називають передчасною збіжністю.

Якщо в якості критерію зупиннення обирається виродження популяції, то задаються кількість ітерацій T і відсоток поліпшення значення кращої хромосоми F.

Починаючи із циклу $T + 1$, у генетичних операторах обчислюються значення цільової функції для кожної хромосоми останньої популяції, і вибирається із цих значень найкраще значення цільової функції $f_{best_{T+t}}$. З попередніх T поколінь вибирається найкраще значення цільової функції $f_{best_{T+t-1}}$. Після чого обчислюється відсоток поліпшення F' по формулі:

$$F' = \frac{100 \cdot (f_{best_{T+t}} - f_{best_{T+t-1}})}{f_{best_{T+t-1}}}.$$

Якщо значення F' виявляється меншим, ніж F, то критерій зупинення вважається досягнутим, в противному випадку виконується наступний цикл генетичного пошуку.

Прийнятне значення цільової функції $f_{\text{п}}$ також може використовуватися в якості критерію зупину. Якщо в процесі функціонування генетичного методу значення цільової функції f деякої особини досягло значення $f_{\text{п}}$ з визначеною заздалегідь заданою точністю ε, то метод зупиняється. При цьому розв'язком задачі є отримане значення цільової функції $f_{\text{п}}$.

Генетичний пошук в пакеті Matlab

Для використання генетичних методів, у середовищі Matlab передбачена функція ga:

```
[x, Fmin] = ga(@fitnessfun, n, options)
```

Функція ga знаходить мінімум Fmin функції fitnessfun, що має n параметрів, а також вектор x, що мінімізує цільову функцію.

Параметри роботи функції ga задаються в змінній options, що представляє собою наступну структуру:

```
options =
        PopulationType: 'doubleVector'
        PopInitRange: [2x1 double]
        PopulationSize: 20
        EliteCount: 2
        CrossoverFraction: 0.8000
        MigrationDirection: 'forward'
        MigrationInterval: 20
        MigrationFraction: 0.2000
        Generations: 100
        TimeLimit: Inf
```

```
FitnessLimit: -Inf
StallLimitG: 50
StallLimitS: 20
InitialPopulation: []
InitialScores: []
PlotInterval: 1
CreationFcn: @gacreationuniform
FitnessScalingFcn: @fitscalingrank
SelectionFcn: @selectionstochunif
CrossoverFcn: @crossoverscattered
MutationFcn: @mutationgaussian
HybridFcn: []
Display: 'final'
PlotFcns: []
OutputFcns: []
Vectorized: 'off'
```

Для зміни значень параметрів функції ga використовується команда gaoptimset, а для одержання поточних параметрів – функція gaoptimget.
Наприклад, для встановлення розміру популяції в 100 особин необхідно використати наступну команду:

```
options = gaoptimset('PopulationSize', 100)
```

Функція gaoptimset дозволяє також передавати параметри в обрані функції, що реалізують основні генетичні оператори. Наприклад, для використання функції мутації mutationgaussian з параметрами scale = 0.5 (середньоквадратичне відхилення) і shrink = 0.75 (параметр скорочення) можна використати таку команду:

```
scale = 0.5; shrink = 0.75;
options = gaoptimset('MutationFcn',{@mutationgaussian,scale,shrink})
```

Приклад
Дано наступні обмеження нерівності і нижні межі

$$\begin{bmatrix} 1 & 1 \\ -1 & 2 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \le \begin{bmatrix} 2 \\ 2 \\ 3 \end{bmatrix},$$

$$x_1 \ge 0, \quad x_2 \ge 0,$$

Наступний код шукає мінімум функції, Fitness_function , яка описана у файлі Fitness_function.m

```
A = [1 1; -1 2; 2 1];
b = [2; 2; 3];
lb = zeros(2,1);
[x,fval,exitflag] = ga(@Fitness_function,…
2,A,b,[],[],lb)
```

Результат роботи:

```
Optimization terminated: average change in the fitness value less than options.TolFun.

x =
   0.6670    1.3340
fval =
  -8.2258
exitflag =
    1
```

Текст Fitness_function.m:

```
function y = Fitness_function(x)

y = 0.5*x(1)^2+x(2)^2-x(1)*x(2)-2*x(1)-6*x(2);
```

Matlab, починаючи з версії 7.0, містить візуальний інтерфейсний модуль для роботи з генетичними методами – Genetic Algorithm Tool (GAT), що входить до складу бібліотеки Genetic Algorithm and Direct Search Toolbox. Запуск GAT відбувається за командою:Gatool або OPTIMTOOL(ga).

Завдання до роботи

1. Ознайомитися з основними теоретичними відомостями за темою роботи.
2.Вивчити роботу функції ga пакету Matlab.
3. Розробити за допомогою пакету Matlab програмне забезпечення, що реалізує 2 методи генетичного пошуку. Основні генетичні оператори для реалізації генетичних методів обрати з таблиці 1.1 відповідно до варіанту.

Інші параметри, необхідні для генетичного пошуку, обрати самостійно. Вибір параметрів обґрунтувати.

4 Виконати тестування розробленого програмного забезпечення за допомогою вирішення задач оптимізації тестових функцій. Тестові функції y_i (не менше п'яти) для виконання тестування програми обрати самостійно. Вибір тестових функцій обґрунтувати.

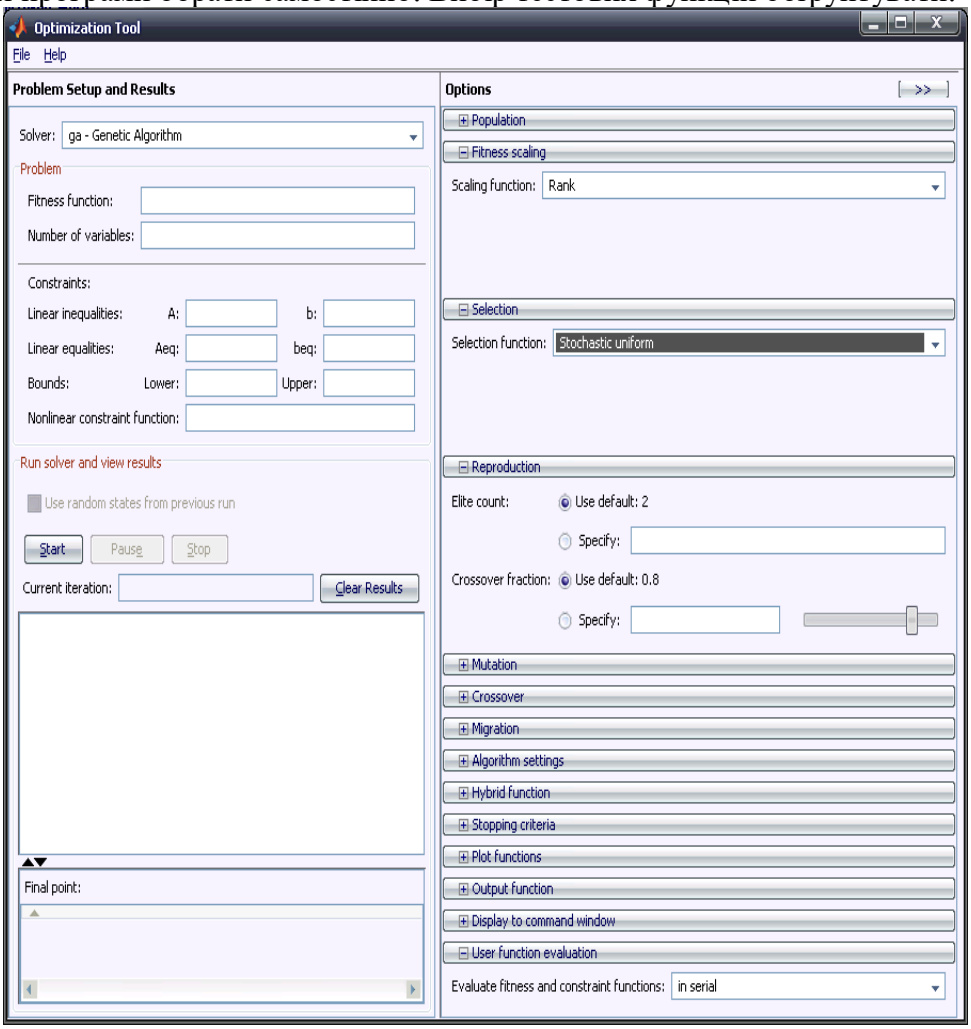


Рис. 1. Візуальний інтерфейсний модуль для роботи з генетичними методами

5 Порівняти одержані результати оптимізації різних функцій за допомогою обох реалізованих генетичних методів. Результати порівняльного аналізу звести до таблиці, попередньо розробивши систему критеріїв порівняння методів генетичного пошуку.

Таблиця 1.1

Генетичні оператори для виконання завдання

№ варіанту	№ задачі	Генетичні оператори		
		Відбір	Схрещування	Мутація
1	1	пропорційний	однорідне	гауссовська
	2	ранжирування	рівномірне	нерівномірна
2	1	рулетка	арифметичне	проста
	2	пороговий	порівняльне	випадкова
3	1	турнірний	одноточечне	гауссовська
	2	ранжирування	діагональне	нерівномірна
4	1	пропорційний	двохточечне	проста
	2	пороговий	рівномірне	нерівномірна
5	1	рулетка	однорідне	гауссовська
	2	ранжирування	порівняльне	нерівномірна
6	1	турнірний	арифметичне	проста
	2	пороговий	діагональне	випадкова
7	1	пропорційний	одноточечне	гауссовська
	2	ранжирування	рівномірне	випадкова
8	1	рулетка	двохточечне	проста
	2	пороговий	порівняльне	нерівномірна
9	1	турнірний	однорідне	гауссовська
	2	ранжирування	діагональне	випадкова
10	1	пропорційний	арифметичне	проста
	2	пороговий	рівномірне	випадкова

6 Оформити звіт з роботи.
7 Дати відповіді на контрольні питання.

Зміст звіту

1. Тема та мета роботи.
2. Короткі теоретичні відомості.
3. Текст розробленого програмного забезпечення з коментарями, а також текст програми для тестування розроблених генетичних методів.
4. Графіки та аналітичні вирази обраних тестових функцій.
5. Результати роботи програмного забезпечення (таблиця порівняльного аналізу розроблених генетичних методів).
6. Висновки, що містять відповіді на контрольні запитання, а також відображують результати виконання роботи та їх критичний аналіз.

Контрольні питання

1 Порівняйте методи генетичного пошуку з іншими методами оптимізації.
2 Які методи відносять до генетичних?
3 В чому переваги генетичних методів?

4 Проаналізуйте умови ефективного використання методів генетичного пошуку.
5 Назвіть особливості генетичних методів.
6 Які недоліки генетичного пошуку та в чому вони полягають?
7 Дайте визначення основних термінів, що відносяться до теорії генетичного пошуку: популяція, розмір популяції, число поколінь, хромосома, ген, локус, алель, фенотип, генотип.
8 Проаналізуйте узагальнену схему роботи генетичних методів.
9 Наведіть послідовність виконання узагальненого генетичного пошуку.
10 Які параметри необхідно визначати для роботи генетичних методів?
11 Виконайте порівняльний аналіз канонічних моделей генетичного пошуку.
12 В чому полягають особливості моделі Genitor?
13 Що таке гібридний еволюційний метод? Які існують стратегії взаємодії класичних та генетичних методів?
14 Назвіть відмінності моделі CHC від класичних генетичних методів.
15 Які особливості генетичного методу із змінним часом життя хромосом?
16 Порівняйте мобільний еволюційний метод з класичними методами генетичного пошуку. Для чого призначені оператори CUT та SPLICE?
17 Проаналізуйте паралельні та багаторівневі генетичні методи.
18 Наведіть послідовність виконання генетичного пошуку із зменшенням розміру популяції.
19 Які існують способи кодування параметрів, що оптимізуються, при використанні генетичних методів?
20 Що таке фітнесс-функція?
21 Порівняйте стратегії створення початкової популяції.
22 Виконайте порівняльний аналіз операторів відбору (пропорційний відбір, відбір за допомогою ранжирування, турнірний відбір та відбір з використанням порогу).
23 Які способи формування батьківської пари використовуються в генетичних методах?
24 Проаналізуйте оператори схрещування (*n*-точкове, рівномірне, порівняльне, арифметичне, діагональне).
25 Для чого призначений оператор мутації? Які оператори мутації використовуються в генетичних методах?
26 Яким чином відбувається формування нового покоління?
27 Які критерії зупину використовуються при еволюційному пошуку?
28 Для чого призначена теорема схем? Дайте визначення основних понять, що використовуються в теоремі схем. В чому полягає теорема Холланда про схеми?
29 Порівняйте генетичні стратегії з генетичними алгоритмами та методом імітації відпалу.
30 Виконайте порівняльний аналіз генетичного та генетичного програмування.
31 Проаналізуйте внутрішню структуру функції ga пакету Matlab: основні змінні, параметри, методи та допоміжні функції, їх призначення та використання.
32 Які параметри можна використовувати в функції ga? Яким чином вони задаються? Як отримати поточні параметри функції ga?
33 роаналізуйте візуальний модуль для роботи з методами генетичного пошуку gatool: призначення, використання, параметри, візуальні компоненти, методи представлення результатів генетичного пошуку.

ЛАБОРАТОРНА РОБОТА № 5

Тема: Аналіз даних за допомогою програмного комплексу WizWhy

Мета: навчитись аналізувати дані за допомогою WizWhy

Теоретичні відомості:

Сенс обробки даний в WizWhy полягає в наступному: по деякому алгоритму програма відшукує в даних логічні закономірності, які «представляє» за допомогою мови правила «якщо - то».

Наприклад (умовний приклад), якщо відповідь на питання 1 - 3 (перша умова), і на питання 4 - 1 (друга умова), і на питання 15 - 1 (третя умова), то відповідь на питання 20 з такої-то ймовірністю - 2.

Завантаження даних

1. WizWhy не вміє читати файли excel, тому перш ніж скористатися наявними даними їх потрібно конвертувати у формат, зрозумілий WizWhy, наприклад, в. Dbf. Відкриваємо файл з даними в Excel, Файл -> Зберегти як -> DBF4 (dBASE IV) (*. Dbf)
2. Вікно програми при запуску виглядає приблизно так. Щоб завантажити дані, вибираємо тип файлу джерела (в нашому випадку dBase), карлючка 1 і натискаємо кнопку Open Data of Type, рисунок 1. У результаті у вкладці Basic Data з'являться потрібні дані (рисунок 2).

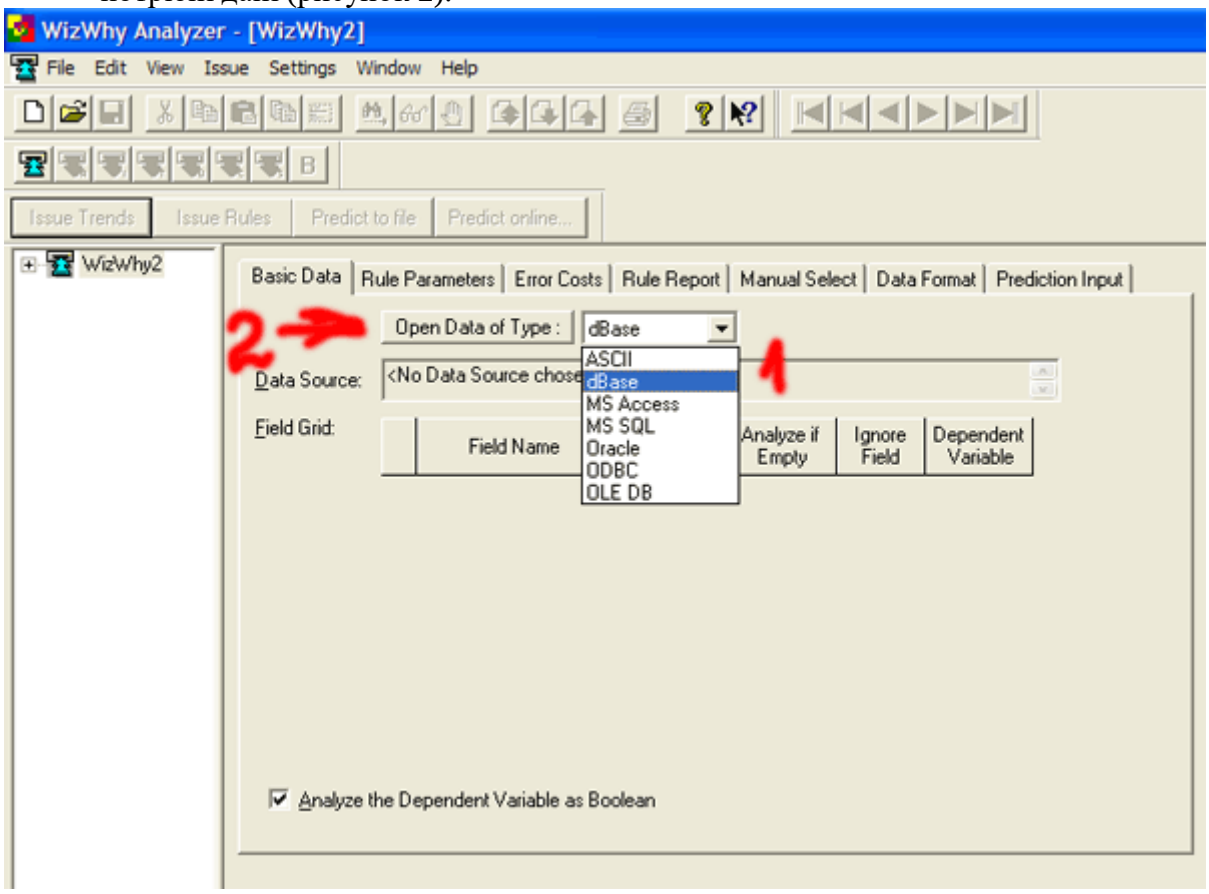


Рисунок 1

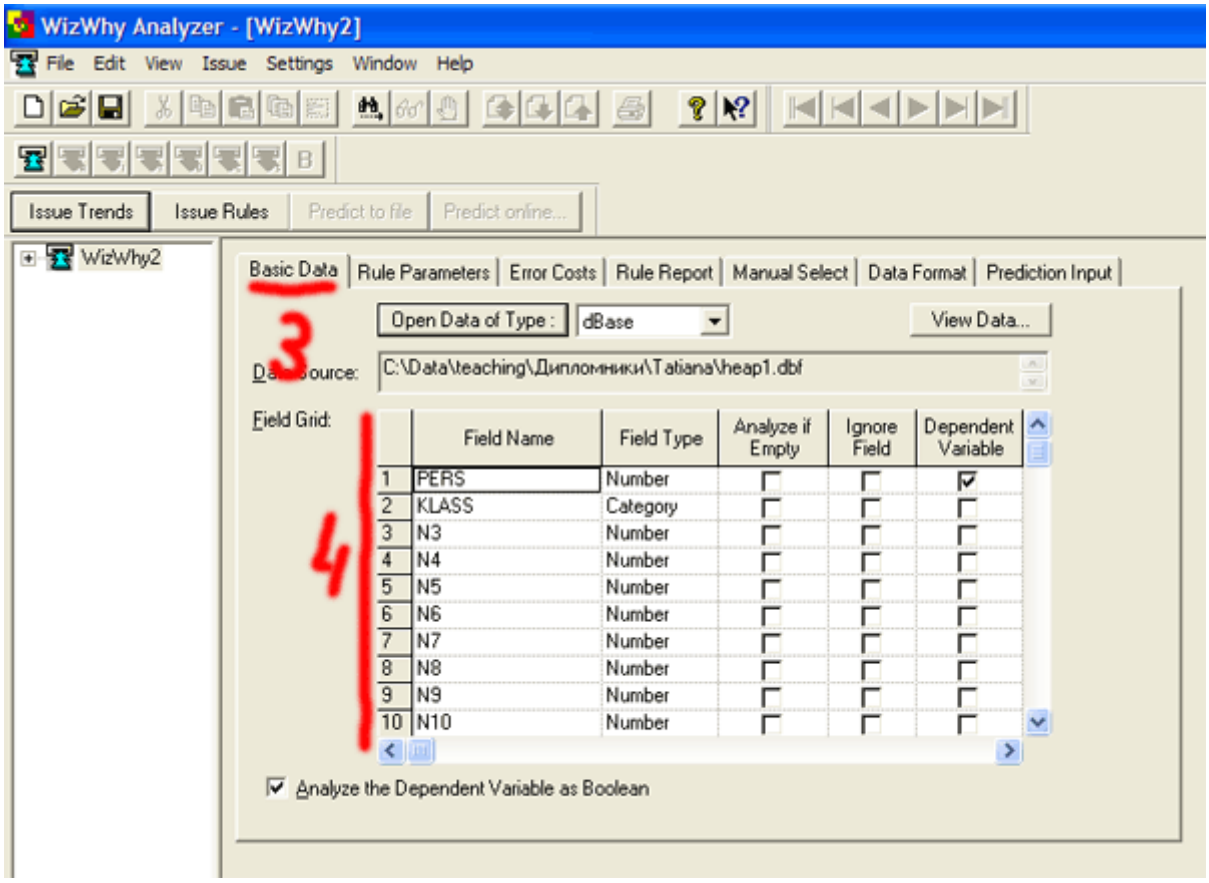

WizWhy Analyzer - [WizWhy2]
File Edit View Issue Settings Window Help
Issue Trends
Issue Rules
Predict to file
Predict online...
WizWhy2
Basic Data
Rule Parameters
Error Costs
Rule Report
Manual Select
Data Format
Prediction Input
Open Data of Type :
dBase
View Data...
Data Source:
C:\Data\teaching\Дипломники\Tatiana\heap1.dbf
Field Grid:
Field Name
Field Type
Analyze if Empty
Ignore Field
Dependent Variable
1 PERS Number
2 KLASS Category
3 N3 Number
4 N4 Number
5 N5 Number
6 N6 Number
7 N7 Number
8 N8 Number
9 N9 Number
10 N10 Number
Analyze the Dependent Variable as Boolean

Рис. 2

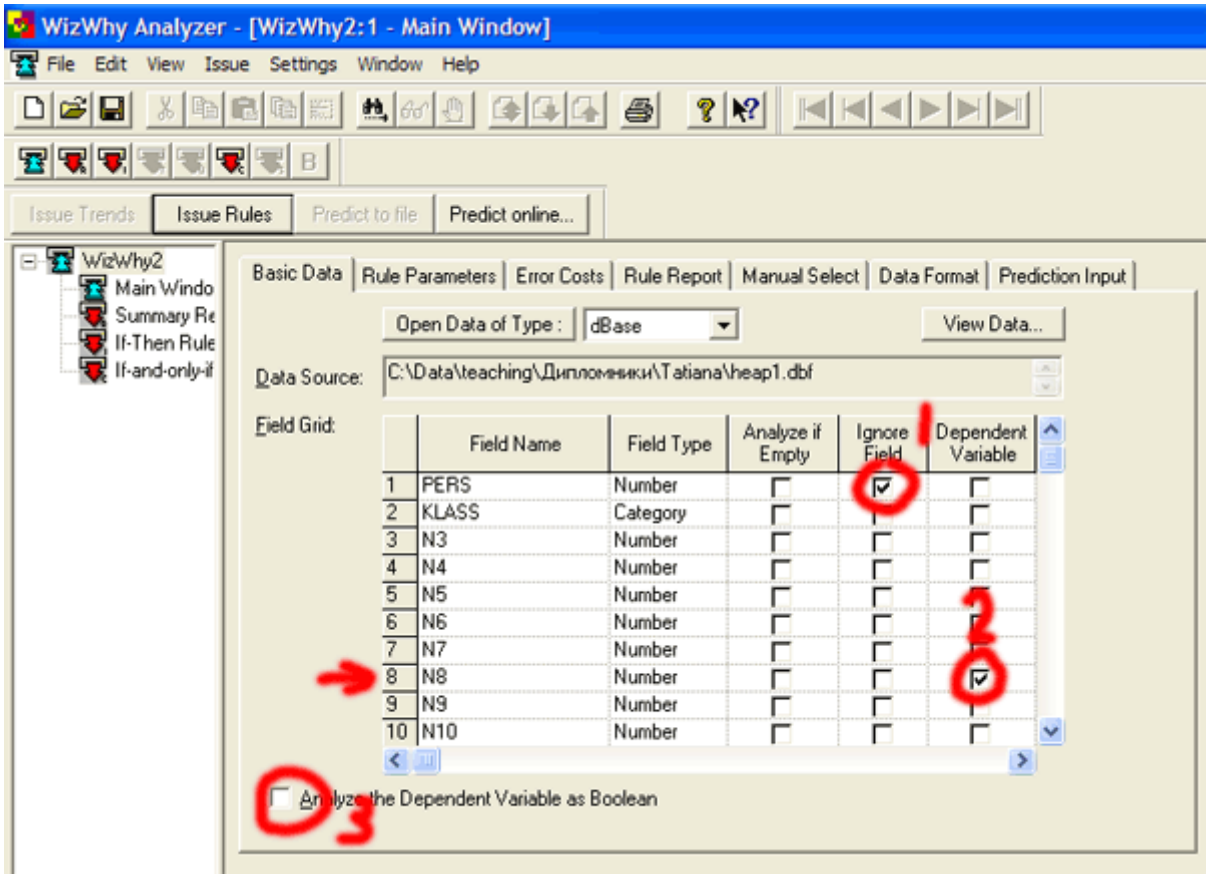

WizWhy Analyzer - [WizWhy2:1 - Main Window]
File Edit View Issue Settings Window Help
Issue Trends
Issue Rules
Predict to file
Predict online...
WizWhy2
Main Windo
Summary Re
If-Then Rule
If-and-only-if
Basic Data
Rule Parameters
Error Costs
Rule Report
Manual Select
Data Format
Prediction Input
Open Data of Type :
dBase
View Data...
Data Source:
C:\Data\teaching\Дипломники\Tatiana\heap1.dbf
Field Grid:
Field Name
Field Type
Analyze if Empty
Ignore Field
Dependent Variable
1 PERS Number
2 KLASS Category
3 N3 Number
4 N4 Number
5 N5 Number
6 N6 Number
7 N7 Number
8 N8 Number
9 N9 Number
10 N10 Number
Analyze the Dependent Variable as Boolean

Рис 3.

Характеристики результуючих правил

Переходимо до наступної вкладки. При знятій галочці Analyze the Dependent Variable as Boolean у нас тільки два параметри, які можна налаштувати: Minimim number of cases in a rule і Maximim number of conditions in a rule. У даному прикладі встановлено перший параметр рівним 20 (або 10% від усієї вибірки,) та максимальне число умов (незалежних змінних) рівним 3. Коли більше - важко інтерпретувати результати.

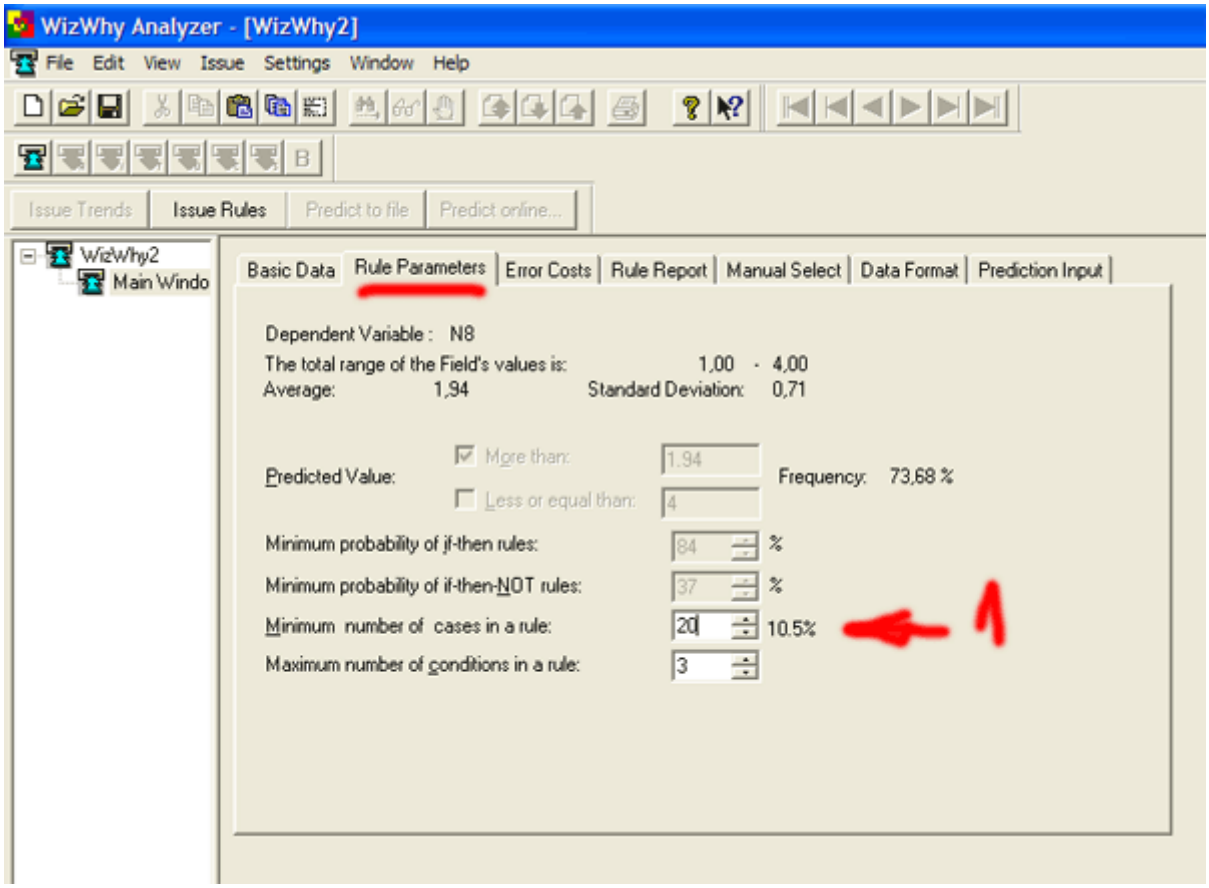


Рис. 4

Якби галочка Analyze the Dependent Variable as Boolean була залишена, WizWhy дозволив би нам налаштувати і інші параменти аналізу (рисунок 5). Можна будувати правила, які б передбачали значення залежної змінної більше 2 (1, 3) і менше 2 (3, 4) (тоді треба було б виставити налаштування як показано карлючкою 1).

Minimum probability of if-then rules і Minimum probability of if-then-NOT rules (мінімальна ймовірність правила if-then і if-then-NOT) - це параметри, що задають необхідну точність правил.Тобто Minimum probability of if-then rules = 27% означає, що ми вказуємо WizWhy, що нас цікавлять тільки такі правила, які помиляються не частіше, ніж в 73% випадків. Аналогічно для if-then-NOT.

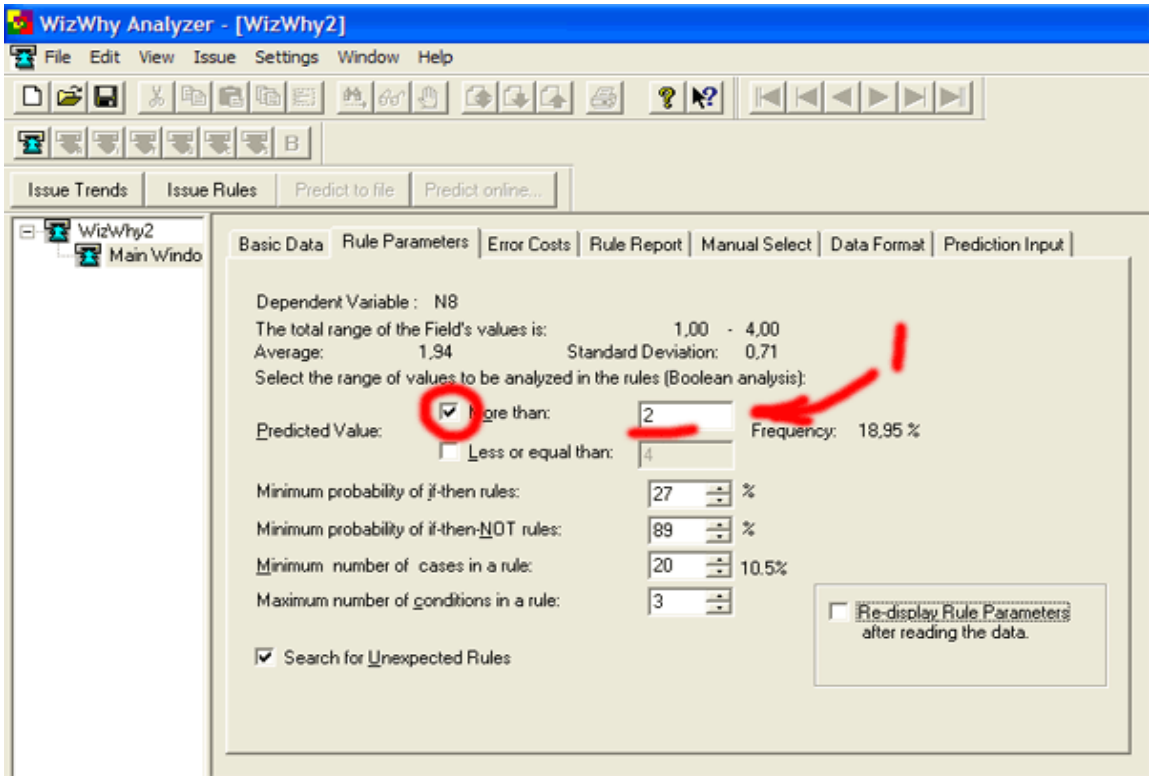


Рис. 5

Ціна помилки

Якби галочка Analyze the Dependent Variable as Boolean знята, WizWhy самостійно встановлює ціни помилок (рисунок 6).

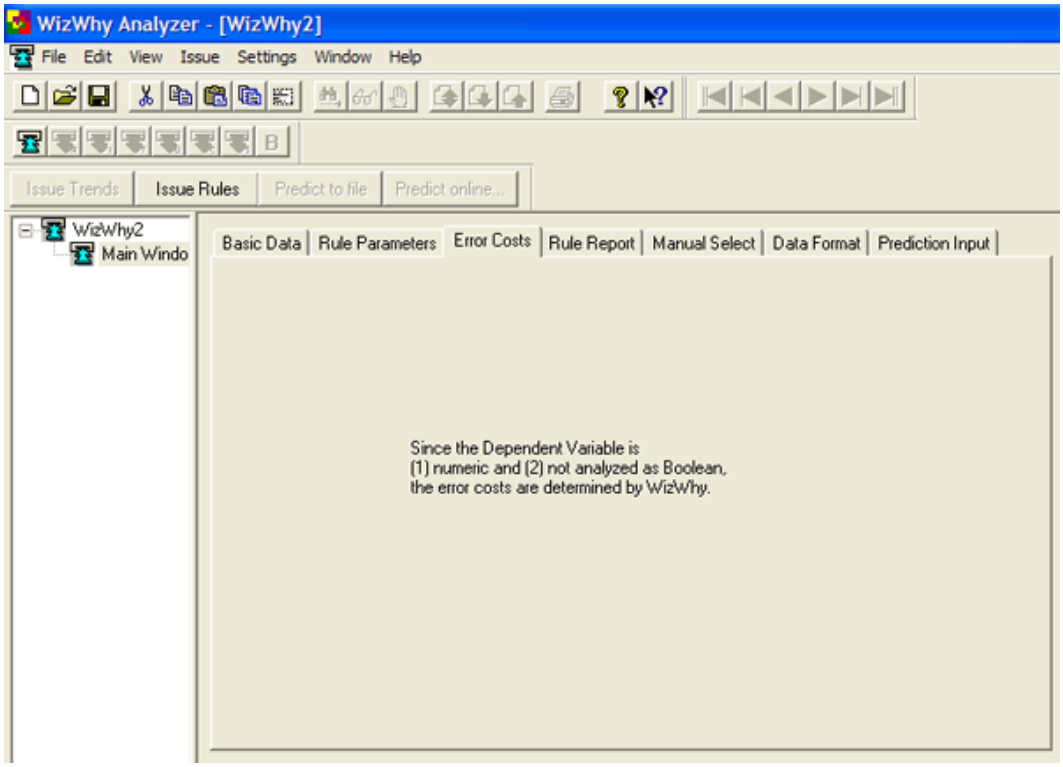


Рис. 6

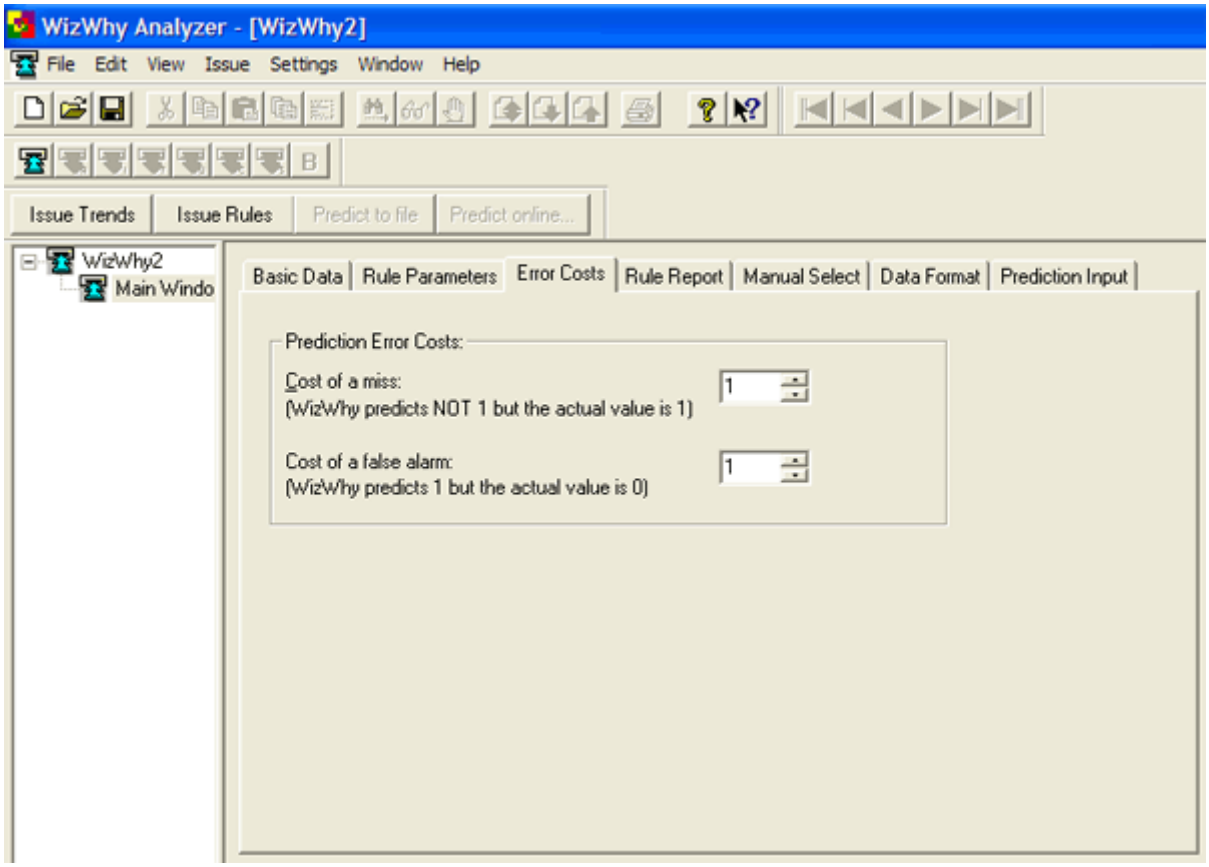


Рис. 7

Параметри виводу результатів

Вкладка Rule Report (Рисунок 8) контролює відображення результатів. Maximum rules to be displayed - максимальна кількість правил, яке WizWhy вам покаже. За замовчуванням 100.

Отримані правила сортуються в звіті (Sort rules by) по:

- significance level - рівню достовірності (= 1-імовірність помилки)
- (Error) probability - імовірність помилки
- number of cases - число спостереження, які "потрапляють під правл"

Крім того, для кожного правила WizWhy вказує конкретні записи "попадають під" та "не попадають під" правило. Кількість таких прикладів задається параметрами Rule is in effect та Rule is NOT in effect.

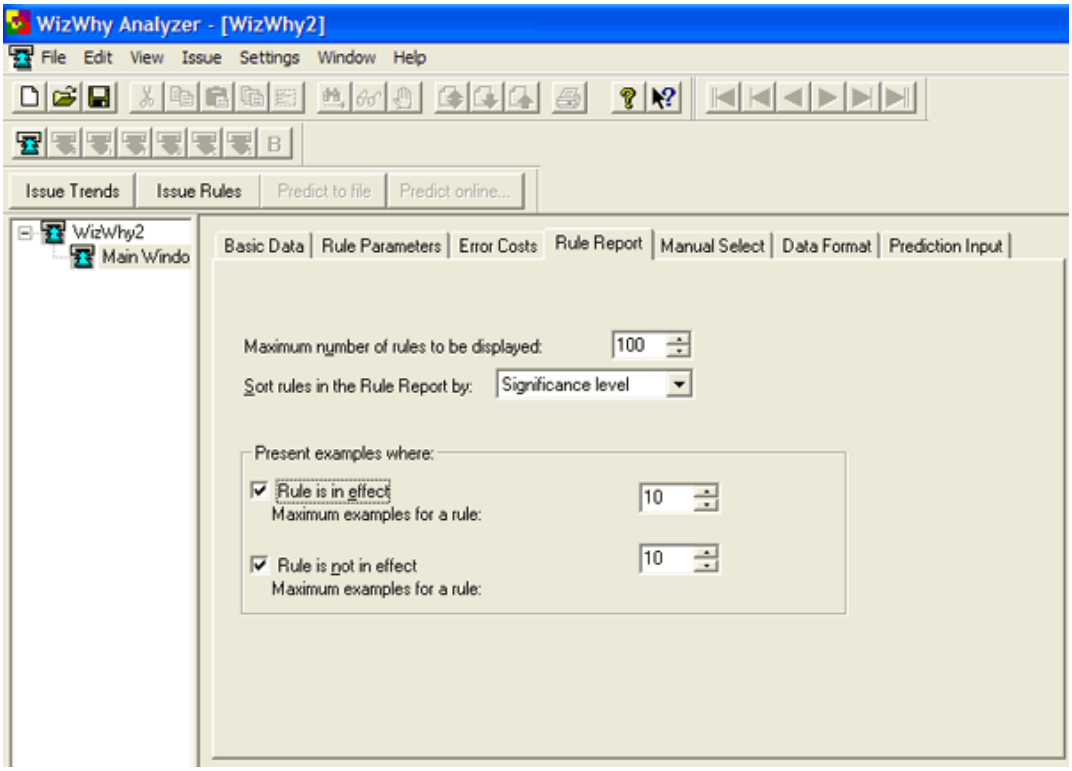


Рис. 8

Запускаємо аналізатор

Після того, як налаштування зроблені, клікаєм по кнопці Issue Rules Про те, що аналіз йде вас буде повідомляти індикатор (2)! -) Дочекайтеся результатів!

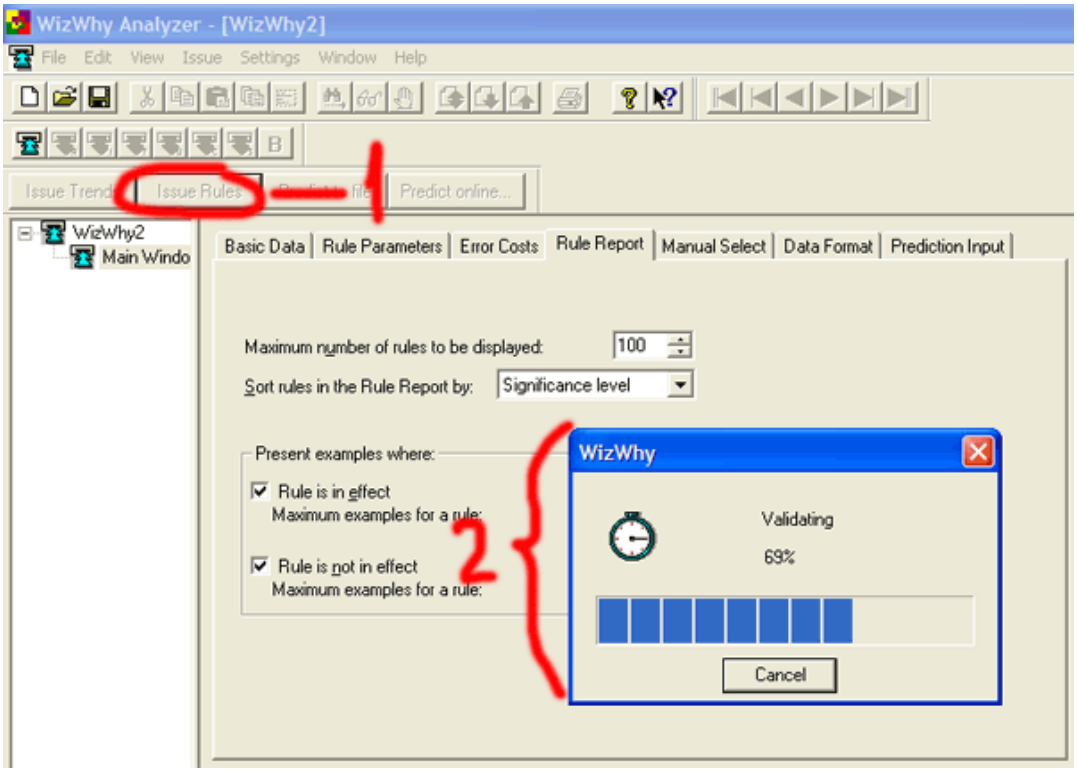


Рис. 9

Інтерпретуємо результати

Після того, як WizWhy завершить всі необхідні розрахунки, на екран буде подана приблизно така картинка (рисунок 10). Якщо не така, то клікніть по елементу If-then-rules

report в лівому вікні (рисунок 10). В середньому вікні будуть відображені ті самі правила, які потрібно інтерпретувати.

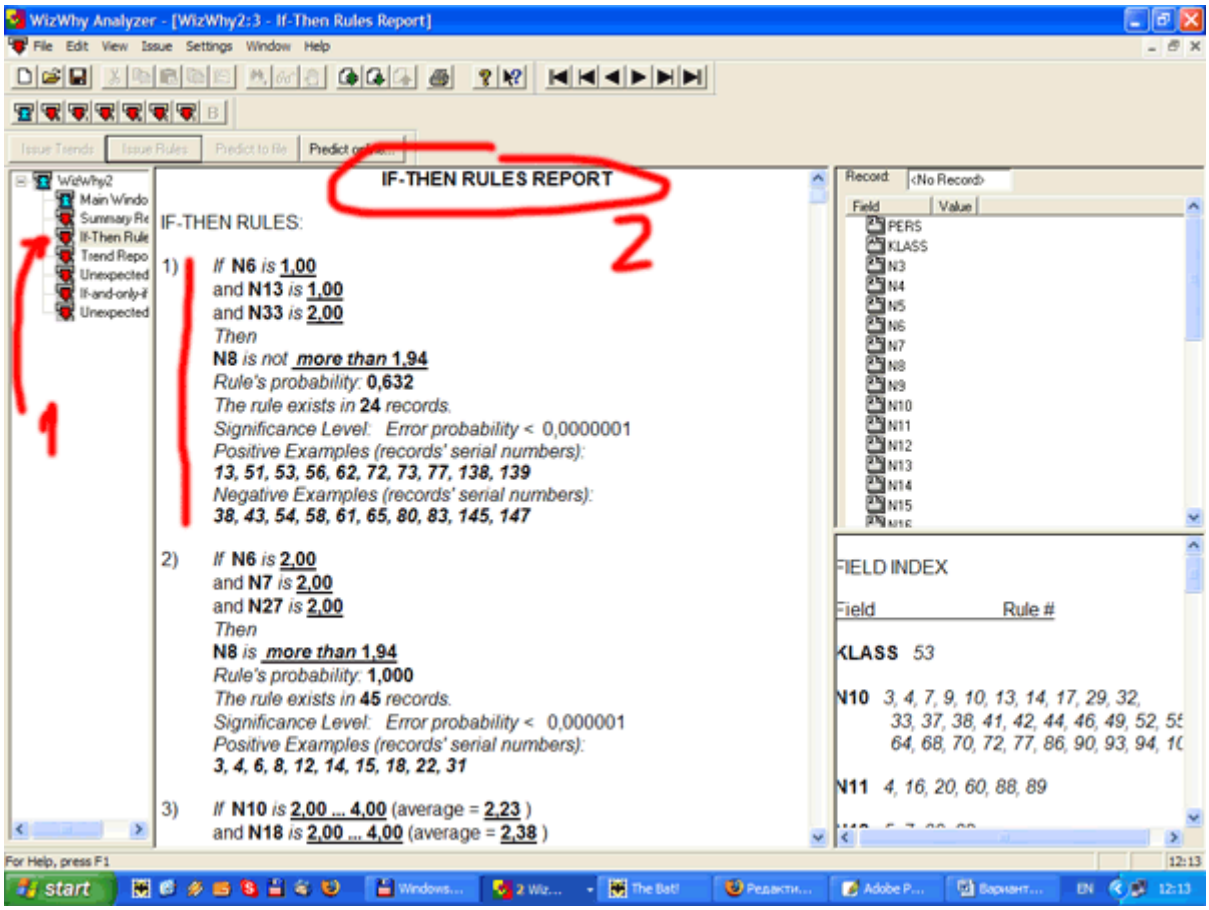


Рис. 10

"Читати" ці правила досить легко. Для прикладу розшифруємо Перше отримане.

```
If N6 is 1,00
       and N13 is 1,00
       and N33 is 2,00
       Then
       N8 IS NOT More Than 1,94
       Rule's probability: 0,632
       The rule exists in 24 records.
       Significance Level: Error probability <0,0000001
       Positive Examples ( records 'serial numbers):
       13, 51, 53, 56, 62, 72, 73, 77, 138, 139
       Negative Examples (records 'serial numbers):
       38, 43, 54, 58, 61, 65, 80, 83, 145, 147
```

Перекладається приблизно так:

```
Якщо відповідь на запитання 4 (пам'ятаєте про зміщення?) - 1
(найвища важливість)
       і відповідь на запитання 11 - 1
       і відповідь на питання 31 - 2
       Тоді
       відповідь на запитання 8 (пророкує змінна) більше ніж
1,94
```

```
(т. к. в настройках ми зняли галочку Analyze the
Dependent Variable as Boolean)
тобто 2 або 3 або 4
вірогідність помилки <1%
приклади рядків, для яких правило справедливо:
13, 51, 53 і т. д.
приклади рядків, для яких правило не справедливо :
38, 43, 54 і т. д.
```

Переналаштовуємо параметри аналізу наступним чином:

- Ставимо галочку Analyze the Dependent Variable as Boolean у вкладці Basic Data;
- Ставимо галочку Less or equal than у вкладці Rule Parametrs і задаємо цьому параметру значення 2.

Таким чином, ми даємо завдання WizWhy знайти правила, що пророчать відповідь 1 або 2 на запитання 6. Всі інші налаштування не змінюємо.

Запускаємо аналіз. Дивимося результати. Наприклад першим видатним WizWhy правило:

```
If N18 is 1,00 ... 2,00 (average = 1,67)
       and N33 is 2,00
       Then
       N8 IS Less Than OR Equal To 2,00
       Rule's probability: 0,989
       The rule exists in 90 records.
       Significance Level: Error probability is almost 0
       Positive Examples (records 'serial numbers):
       10, 13, 14, 28, 30, 31, 33, 38, 40, 41
       Negative Examples (records 'serial numbers):
       186
Якщо відповідь на запитання 16 (пам'ятаєте про зміщення?) 1 або
2
       і відповідь на питання 31 - 2,
       То
       Відповідь на питання 1 або 2
       З дуже високою ймовірністю
       Правило "накриває" 90 записів.
```

Завдання для виконання :

1. Ознайомлення із теоретичними відомостями.
2. Завантаження даних
3. Налаштування параметрів

3.1 Передбачуваня змінна
3.2 Характеристики результуючих правил
3.3 Ціна помилки
3.4 Параметри виводу результатів
3.5 Додаткові параметри налаштування
3.6 Запуск аналізатора

4. Інтерпретуючі результати
5. Збереження результатів
6. Зробити екранні форми.
7. Висновок.

ЛІТЕРАТУРА :

1. Великі дані. [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Великі_дані
2. Data Mining – Cluster Analysis [Электронный ресурс]. – Режим доступа: https://www.tutorialspoint.com/data_mining/dm_cluster_analysis.htm
3. 515K Hotel Reviews Data in Europe [Режим електронного доступу] https://www.kaggle.com/jiashenliu/515k-hotel-reviews-data-in-europe?select=Hotel_Reviews.csv
4. The Python Package Index (PyPI) is a repository of software for the Python programming language [Режим електронного доступу] https://pypi.org/
5. NumPy and Pandas Tutorial – Data Analysis with Python https://cloudxlab.com/blog/numpy-pandas-introduction/
6. PolyAnalyst 6.5 Technical capabilities and system requirements, Megaputer intelligence, 2019.
7. Інтелектуальный аналіз тексту URL: https://uk.wikipedia.org/wiki/Інтелектуальний_аналіз_тексту
8. Big Data і блокчейн. [Електронний ресурс]. – Режим доступу: https://forklog.com/big-data-i-blokchejn-proryv-v-oblasti-analiza-dannyh/

Навчально-методичне видання

Гончар Людмила Іванівна
Марценюк Євгенія Олексіївна

МЕТОДИЧНІ ВКАЗІВКИ
ДО ВИКОНАННЯ ЛАБОРАТОРНИХ РОБІТ

з дисципліни

«ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ»

для магістрів спеціальності 121
Інженерія програмного забезпечення

Підписано до друку 21.09.2022 р.
Формат 60х84/16. Папір офсетний.
Друк офсетний. Зам. № 22-929
Умов.-друк. арк. 3,9. Обл.-вид. арк. 4,1.
Тираж 30 прим.

Віддруковано ФО-П Шпак В. Б.
Свідоцтво про державну реєстрацію В02 № 924434 від 11.12.2006 р.
м. Тернопіль, бульвар Просвіти, 6/4. тел. 097 299 38 99.
E-mail: tooums@ukr.net

Свідоцтво про внесення суб'єкта видавничої справи до державного реєстру видавців, виготовлювачів і розповсюджувачів видавничої продукції ДК № 7599 від 10.02.2022 р.