Міністерство освіти і науки, молоді та спорту України
Тернопільський національний економічний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

«До захисту допущено»
Завідувач кафедри
комп'ютерної інженерії
к.т.н., доц. О.М.Березький
_____
"____"_____20__ р.

ДИПЛОМНИЙ ПРОЕКТ
освітньо-кваліфікаційного рівня "Спеціаліст"
зі спеціальності 7.05010201 "Комп'ютерні системи та мережі"

на тему:

СИСТЕМА ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ОЦІНКИ КРЕДИТНИХ РИЗИКІВ

Студент групи КСМзс-51_____Тимош В.В.
(підпис)
Керівник:
д.е.н., професор _____ Ріппа С.П.
(підпис)

Нормоконтроль
к.т.н., доцент _____Васильків Н.М.
(підпис)

Консультант
з охорони праці
доцент _____Сапожник Г.В.
(підпис)

2012

Міністерство освіти і науки, молоді та спорту України
Тернопільський національний економічний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії
спеціальність 7.05010201 – "Комп'ютерні системи та мережі"

"Затверджую"
завідувач кафедри
комп'ютерної інженерії
к.т.н., доц. О.М.Березький
_____
"____"_____20__ р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ СТУДЕНТА
Тимош Віталій Володимирович

1. **Тема проекту:** "Система підтримки прийняття рішень оцінки кредитних ризиків" затверджена наказом університету № ___від "___"_____20__ р.
2. **Термін здачі студентом закінченого проекту** "___"_____20__ р.
3. **Вихідні дані для проекту:** Технічне завдання.
4. **Перелік задач, які мають бути вирішені:**
   - розглянути існуючі типи беєсівських мереж;
   - виконати огляд сучасних підходів та методів побудови топології і ймовірнісного висновку;
   - визначити найбільш прості та перспективні напрямки вдосконалення існуючих підходів;
   - розробити евристичний алгоритм побудови беєсівських мереж за навчальними даними;
   - розробити алгоритм побудови ймовірнісного висновку в беєсівських мережах, за навчальними даними;
   - розробити архітектуру інформаційної системи підтримки прийняття рішень оцінки кредитних ризиків на основі байєсівських мереж;
   - реалізувати програмно запропоновану систему, ґрунтуючись на сучасних підходах щодо побудови топології та імовірнісного висновку в байєсівських мережах.

5. **Перелік графічного матеріалу** (з точним вказанням обов'язкових креслень)
   - СППР. Схема структурна
   - Архітектура СППР. Схема структурна
   - БМ. Схема взаємозв'язків функціональних елементів
   - Структура системи кредитного скорингу у вигляді БМ

6.Консультанти по проекту (із зазначенням розділів):

Розділ	Консультант	Підпис
Охорона праці	Сапожник Г.В.	

КАЛЕНДАРНИЙ ПЛАН

№	Назва розділів дипломного проекту	Термін виконання	Позначки керівника про виконання завдань
1	Технологія інтелектуального аналізу даних	15.09.2011 – 5.11.2011	
2	Структура та алгоритми побудови байєсівської мережі	6.11.2011 – 31.01.2012	
3	Система підтримки прийняття рішень оцінки кредитних ризиків на основі байєсівських мереж	1.02.2012 – 14.04.2012	
4	Охорона праці	15.04.2012 – 23.04.2012	

Завдання прийняв до виконання _____
(підпис)

Керівник дипломного проекту _____
(підпис)

АНОТАЦІЯ

Дипломний проект на тему “Система підтримки прийняття рішень оцінки кредитних ризиків” на здобуття освітньо-кваліфікаційного рівня “Спеціаліст” зі спеціальності 7.05010201 “Комп’ютерні системи та мережі” написаний обсягом 127 сторінок і містить 47 ілюстрацій, 24 таблиці, 6 додатків та 24 джерела інформації.

Для побудови структури байєсівської мережі запропоновано евристичний алгоритм навчання лінійної складності за статистичними даними.

Розроблено алгоритм побудови точного ймовірнісного висновку в байєсівських мережах за навчальними даними.

Розроблена і програмно реалізована оригінальна система підтримки прийняття рішень оцінки кредитних ризиків на основі байєсівських мереж, яка ґрунтується на запропонованих алгоритмах побудови структури та ймовірнісного висновку.

ANNOTATION

The diploma project “Decision-taken Support System for evaluation of credit risks” on acquiring educational by qualification “Specialist” degree, speciality 7.05010201 Computer Systems and Networks, has the general volume of 127 pages and contains 47 illustrations, 24 tables, 6 additions and 24 sources of information.

To construct a Bayesian network structure is proposed heuristic algorithm for learning linear complexity statistics.

The algorithm of constructing an accurate probabilistic conclusion in Bayesian networks on the basis of training data is made.

Original software credit risk assessment decision-taken support system based on Bayesian networks is designed and implemented. All this is based on the proposed algorithm for constructing the structure and probabilistic conclusion.

Технічне завдання

1. НАЙМЕНУВАННЯ ТА ОБЛАСТЬ ЗАСТОСУВАННЯ

1.1 Система підтримки прийняття рішень оцінки кредитних ризиків

1.2 Область застосування – інтелектуальний аналіз даних.

2. ОСНОВА ДЛЯ РОЗРОБКИ

Основою для розробки є завдання на дипломний проект, затверджене кафедрою комп'ютерної інженерії факультету комп'ютерних інформаційних технологій Тернопільського національного економічного університету.

3. ПРИЗНАЧЕННЯ РОЗРОБКИ

Метою роботи є розробка системи підтримки прийняття рішень оцінювання ризиків кредитування фізичних осіб.

4. ДЖЕРЕЛА РОЗРОБКИ

Джерелами даної розробки є матеріали навчальної та реферативної наукової літератури, технічна документація, існуючі програмні системи, журнали, науково-дослідні роботи вітчизняних та закордонних вчених.

5. ТЕХНІЧНІ ВИМОГИ

5.1 Вимоги до апаратних засобів

5.1.1 Функціональні вимоги до апаратних засобів.

5.1.1.1 Система повинна працювати на IBM-сумісних робочих станціях.

5.1.1.2 Мінімальні вимоги до робочих станцій:

– тактова частота процесора не менше 300 Мгц;

– 4 Мбайт вільного місця на жорсткому диску для програмних модулів системи та 4 Мбайт для навчальних даних і формування баз даних (останнє значення не фіксоване і може змінюватися в залежності від умов експлуатації);

– оперативна пам'ять 32 Мбайт і більше;

– операційна система Windows 98/2000/NT/XP;

– пристрій для забезпечення безперебійної роботи комп'ютера для можливості автономної роботи програми;

– клавіатура та комп'ютерна мишка;

– пристрій для запису даних і результатів на оптичні носії для резервного копіювання файлів баз даних.

5.2 Вимоги до програмної системи

5.2.1 Функціональні вимоги до програмної системи

5.2.1.1 Оператор системи повинен мати змогу виконувати наступні функції:

– формування конкретних процедур обробки даних та прогнозування (формулювання вимог);

– вибір та формулювання критеріїв розв'язку задачі;

– виконання задач моделювання і прогнозування;

– формування результатів.

5.2.1.2 Вхідна інформація отримується шляхом:

– введення опратором інформації, даних, команд, параметрів і запитів в систему;

– завантаження даних з файлу, вказавши шлях до файлу на жорсткому диску;

5.2.1.3 Вихідна інформація:

– вихідна інформація повинна подаватись у зручному для сприйняття вигляді;

– формування результатів повинно відбуватись у реальному часі;

– вихідна інформація виводиться у текстовому, табличному, графічному (графіки, діаграми) форматах;

5.2.2 Вимоги до надійності.

5.2.2.1 Передбачити контроль введеної інформації.

5.2.2.2 Розробити комплекс заходів контролю коректності дій користувача під час роботи з системою.

5.2.2.3 Забезпечити можливість відновлення роботи системи після збоїв.

5.2.3 Вимоги до програмного забезпечення:

5.2.3.1 Операційна система сімейства Windows;

5.2.3.2 Microsoft Excel;

5.2.3.3 Сумісність з сучасними форматами даних:

– вхідна інформація подається у форматах «*.txt»;

– вихідна інформація подається у вигляді таблиць у форматі «*.xls».

5.2.4 Вимоги до програмної документації

5.2.4.1 Коди програмних модулів повинні містити необхідні для їх розуміння коментарі;

5.2.4.2 Розроблене програмне забезпечення повинно включати довідкову систему

5.2.5 Вимоги експлуатації

5.2.5.1 Кліматичні вимоги до експлуатації, при яких забезпечується робота програми повинні відповідати кліматичним умовам експлуатації наявних технічних засобів.

5.2.5.2 Вимоги до кваліфікації та численності персоналу

– мінімальна кількість персоналу, необхідного для роботи програми – одна штатна одиниця – кінцевий користувач програми – оператор.

5.2.6 Вимоги до захисту:

5.2.6.1 Мінімальна довжина пароля – 8 символів.

6. ВИМОГИ ОХОРОНИ ПРАЦІ

В розділі “Охорона праці ” дипломного проекту повинен бути даний аналіз умов праці в приміщенні де працює розробник програмного засобу.

7. ПОРЯДОК КОНТРОЛЮ І ПРИЙОМКИ

7.1 Представлення дипломного проекту на попередній захист

7.2 Представлення дипломного проекту на захист

ПЕРЕЛІК СКОРОЧЕНЬ

АБМ – алгебраїчна байєсівська мережа

БД – база даних

БЗ – база знань

БМод – база моделей

БМ – байєсівська мережа

БМД – байєсівська мережа довіри

ВМВ – впорядкована множина вершин

ГПА – головна підсистема аналізу

ЗВІ – значення взаємної інформації

ІАД – інтелектуальний аналіз даних

КГ – функція Купера-Герсковича

ЛІМ – лінійна імовірнісна модель

МЛКГ – модифікована логарифмічна функція Купера-Герсковича

МНК – метод найменших квадратів

ОМД – опис мінімальною довжиною

ОПР – особа, що приймає рішення

СД – сховище даних

СППР – система підтримки прийняття рішень

ТУЙ – таблиця умовних ймовірностей

ТУН – тест на умовну незалежність

ШНМ – штучна нейронна мережа

ВСТУП

Станом на березень 2011 року згідно котировок CDS (Credit Default Swaps) Україна була першим кандидатом на дефолт в світі. Однією з причин такого низького рейтингу є слабкий рівень впровадження та використання скорингових систем у банківській сфері, що обумовило велику кількість проблемних кредитів.

Задача побудови моделі оцінювання ризиків кредитування фізичних осіб є надзвичайно актуальною, оскільки на протязі останніх років швидкими темпами розвивається кредитування фізичних осіб. При цьому збільшується кількість дефолтів (неповернення кредитів). За деякими продуктами (наприклад, споживчі кредити на купівлю товарів) об’єми втрат складають близько 15% [1]. За даними рейтингової компанії Standard and Poor's (станом на кінець 2011 року) кожен третій виданий в Україні кредит є проблемним. Standard and Poor's відносить банківську систему України до останньої, самої слабкої групи № 10. Окрім України, у цю групу входять Венесуела, Ямайка та Болівія. Для порівняння, Білорусія входить в 9-у групу, а Росія в 7-у.

Отже, метою роботи є розробка системи підтримки прийняття рішень оцінювання ризиків кредитування фізичних осіб.

Для успішного досягнення поставленої мети необхідно виконанти наступні завдання:

- розглянути існуючі типи беєсівських мереж;
- виконати огляд сучасних підходів та методів побудови топології і ймовірнісного висновку;
- визначити найбільш прості та перспективні напрямки вдосконалення існуючих підходів;
- розробити евристичний алгоритм побудови беєсівських мереж за навчальними даними;
- розробити алгоритм побудови ймовірнісного висновку в беєсівських мережах, за навчальними даними;
- розробити архітектуру інформаційної системи підтримки прийняття рішень

оцінки кредитних ризиків на основі байєсівських мереж;

– реалізувати програмно запропоновану систему, ґрунтуючись на сучасних підходах щодо побудови топології та імовірнісного висновку в байєсівських мережах.

1 ТЕХНОЛОГІЯ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ

1.1 Аналіз методів та алгоритмів інтелектуального аналізу даних

1.1.1 Дерева рішень

Метод дерев рішень (decision trees або answer tree) – одним із самих популярних методів розв'язання завдань класифікації та прогнозування. Іноді цей метод інтелектуального аналізу даних (ІАД) називають також деревами вирішальних правил, деревами класифікації та регресії.

Якщо залежна, тобто цільова змінна приймає дискретні значення, то за допомогою методу дерева рішень вирішується задача класифікації.

Якщо ж залежна змінна приймає безперервні значення, то дерево рішень установлює залежність цієї змінної від незалежних змінних, тобто вирішується задача чисельного прогнозування.

Уперше дерева рішень були запропоновані Ховілендом (Hoveland) та Хантом (Hunt) наприкінці 50-х років минулого століття. Самою ранньою роботою у якій викладається суть дерев рішень являється робота Ханта "Експерименти в індукції" ("Experiments in Induction") [2], яка була опублікована в 1966 році.

У найбільш простому виді дерево рішень – це спосіб подання правил в ієрархічній, послідовній структурі. Основа такої структури – відповіді "Так" або "Ні" на ряд запитань.

На рисунках 1.1, 1.2 і 1.3 наведені приклади дерев рішень, задачею яких є дати відповідь на питання: "Чи видавати кредит?" Для того щоб вирішити задачу, необхідно визначити ймовірність дефолту клієнта банку. Для цього необхідно відповісти на ряд запитань, що знаходяться у вузлах (вершинах) цього дерева, починаючи з його кореня.

Як приклад, розглянемо дерево рішень, наведене на рисунку 1.1. Змінна "Поручитель" – вершина перевірки, тобто умова. Якщо відповідь позитивна – "Так", то здійснюється перехід до верхньої частини дерева до вершини "Вік", при негативній – до нижньої частини дерева.

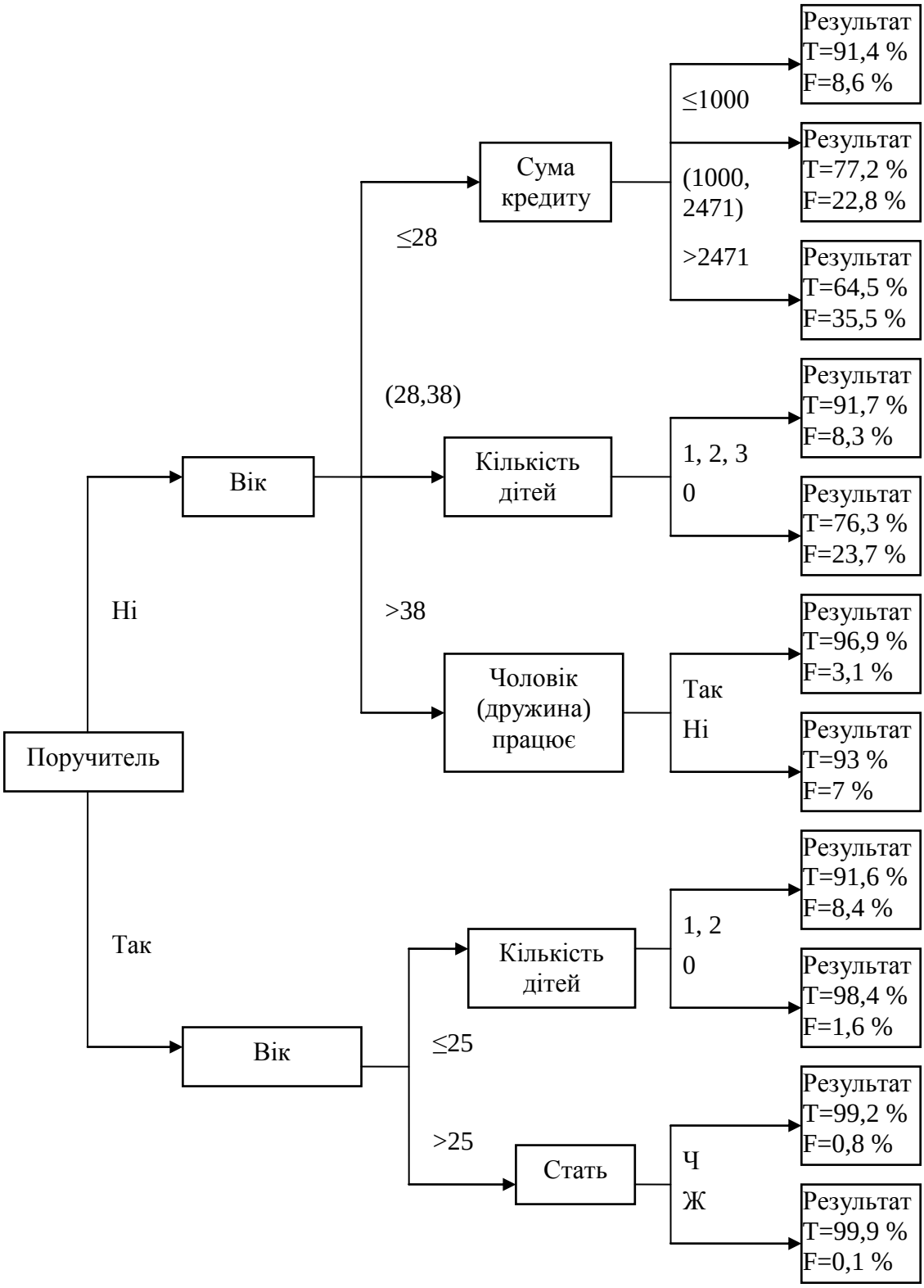


Рисунок 1.1 – Система кредитного скорингу у вигляді дерева рішень, побудованого за методом Chaid

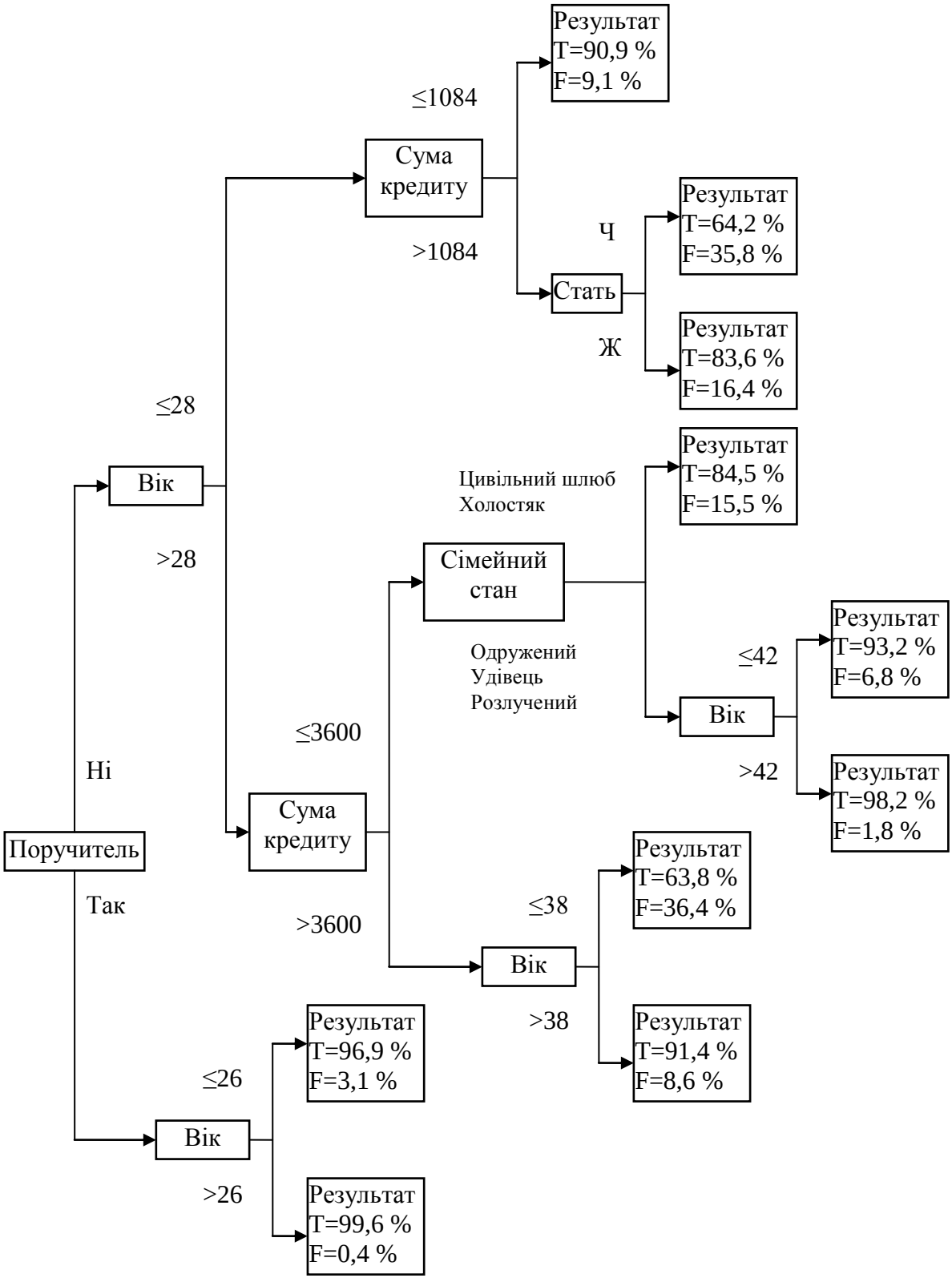


Рисунок 1.2 – Система кредитного скорингу у вигляді дерева рішень, побудованого за методом CART

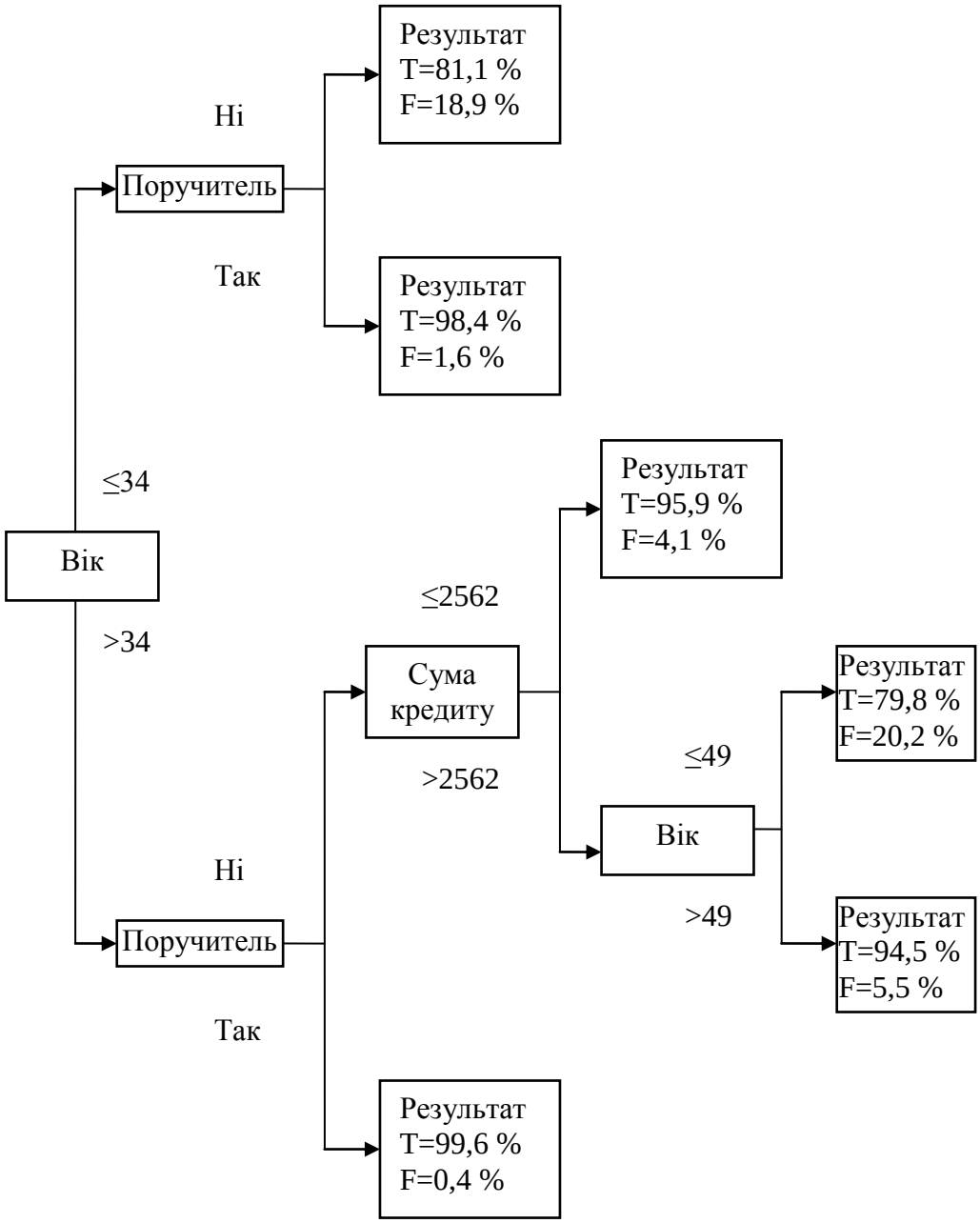


Рисунок 1.3 – Система кредитного скорингу у вигляді дерева рішень, побудованого за методом QUEST

Таким чином, внутрішній вузол (вершина) дерева – це вузол перевірки певної умови. Далі йде наступне питання, і так далі до тих пір, поки не буде досягнуто кінцевого вузла (вершини) дерева, який представляє вузол рішення. Для нашого дерева кінцевим вузлом є вершина “Результат”, що приймає два значення: “T” та “F” (T = *true* та F = *false*), де “F” – ймовірність неповернення клієнтом

кредиту виданого банком (дефолту).

В результаті проходження від кореня дерева “Поручитель” до його кінцевої вершини “Результат” розв’язується задача класифікації. На основі отриманих значень ймовірностей “T” та “F” визначається ступінь кредитоспроможності клієнта і приймається рішення щодо видачі або відмови у наданні кредиту.

Типовий алгоритм побудови топології (структури) дерев рішень складається з двох етапів: “побудова” або “створення” дерева (tree building) та “скорочення” дерева (tree pruning). При побудові дерева вирішуються питання вибору критерію розщеплення та зупинки процедури навчання (якщо це передбачено алгоритмом). У ході етапу скорочення дерева вирішується питання відсікання деяких його гілок.

CHAID (CHi-squared Automatic Interaction Detector). Це найбільш відомий метод побудови дерев рішень, згідно з яким для одержання оптимальної розбивки використається критерій зв'язку між категоріальними змінними χ^2 (у випадку, якщо цільова змінна є кількісною, використається $F-$критерій). Дані для аналізу цільової змінної та змінні-фактори можуть бути як кількісними, так і категоріальними, однак кількісні змінні-фактори при побудові дерева перетворюються в категоріальні.

Exhaustive CHAID. Даний метод представляє собою модифікацію методу CHAID. Його перевагою є те, що в процесі побудови дерева аналізується більша кількість можливих видів розбиття, а недоліком – більші витрати часу на виконання.

CART (Classification and regression trees). Метод, відомий також як "метод побудови дерев регресії й класифікації". На відміну від двох описаних вище методів, він грунтується не на статистичних критеріях, а на зменшенні неоднорідності сегментів (вузлів). Добре працює у тому випадку, якщо всі змінні – кількісні. У методі можуть бути використані як кількісні, так і категоріальні цільові змінні та змінні-фактори.

QUEST (Quick, Unbiased and Efficient Statistical Tree). У цьому методі для вибору факторів застосовують різні критерії, в залежності від типу потенційного фактору. Він дозволяє уникати зсувів, пов'язаних з вибором факторів з більшою

кількістю категорій, але цільова змінна в цьому випадку повинна бути категоріальною. Змінні-фактори можуть бути як кількісними, так і категоріальними.

На рисунках 1.1, 1.2 та 1.3 наведені дерева рішень, побудовані за методами CHAID, CART та QUEST. У таблиці 1.1 наведені характеристики побудованих моделей. З цієї таблиці видно, що серед дерев рішень найкращим виявилося побудоване за методом CHAID.

Таблиця 1.1 – Таблиця похибок методів дерев рішень

Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
За умови 90% ймовірності повернення кредиту				
CHAID	76	275	351	0.895
CART	68	299	367	0.891
QUEST	47	546	593	0.823
За умови 80% ймовірності повернення кредиту				
CHAID	178	0	178	0.947
CART	98	142	240	0.928
QUEST	178	0	178	0.947

Опис алгоритму C4.5 побудови дерев рішень

Нехай задана множина навчальних прикладів T і кожний приклад характеризується m атрибутами. Один з цих атрибутів повинен вказувати на належність прикладу до певного класу, тобто один з атрибутів повинен бути обраний як ознаки класу. Нехай ознака класу приймає значення $C_1,...,C_k$. Класи повинні бути дискретними, тобто мати скінчену кількість значень. Кожен приклад повинен однозначно відноситися до конкретного класу. Випадки, коли приклади належать до класу з ймовірнісними оцінками, виключаються.

Процес побудови дерева здійснюється зверху вниз. Спочатку створюється корінь дерева, потім нащадки кореня і т. д..

На першому кроці є тільки корінь (порожнє дерево) і множина навчальних прикладів T. Потрібно розбити множину T на підмножини. Це можна зробити шляхом вибору одного з атрибутів як перевірочного. В результаті розбиття отримують n (по кількості значень атрибута) підмножин та відповідно створюють n нащадків кореня, кожному з яких поставлена у відповідність своя підмножина, отримана при розбивці множини T. Далі ця процедура рекурсивно застосовується до всіх підмножин (нащадків кореня) і так далі.

Розглянемо докладніше критерій вибору атрибуту, за яким повинне піти розгалуження. Існує m можливих варіантів (за числом атрибутів), з яких ми повинні обрати потрібний. Деякі алгоритми виключають повторне використання атрибута при побудові дерева, але в C 4.5 це обмеження відсутнє.

Нехай виконується перевірка X (в якості перевірки може бути обрано будь-який атрибут), що приймає n значень $A_1,...,A_n$. Тоді розбивка T по X надає нам підмножини $T_1,...,T_n$.

Нехай $freq(C_j,S)$ – кількість прикладів з деякої множини S, що відповідає класу C_j. Тепер ймовірність того, що випадково обраний приклад з множини S буде належати до класу C_j, обчислюється за формулою:

$$P = \frac{freq(C_j,S)}{|S|}.$$

Оцінка середньої кількості інформації, необхідної для визначення класу прикладу з множини T обчислюється, як ентропія цієї множини за формулою:

$$Info(T) = -\sum_{j=1}^{k}\left(\left(\frac{freq(C_j,T)}{|T|}\right)\cdot\left(\log_2\left(\frac{freq(C_j,T)}{|T|}\right)\right)\right),$$

де $|T|$ – кількість елементів в множині T.

Таку ж оцінку, але після розбиття множини T по X, дає вираз:

$$Info_X(T) = \sum_{i=1}^{n}\left(\frac{|T_i|}{|T|}\cdot Info(T_i)\right).$$

Як критерій вибору атрибута використовують теоретико-інформаційний

критерій:

$$Gain(X) = Info(T) - Info_X(T).$$

Можуть застосовуватися також інші статистичні критерії, наприклад в алгоритмі CART використовується індекс Gini (на честь італійського економіста Corrado Gini), який оцінює “відстань” між розподілами класів:

$$Gini(C) = 1 - \sum_j p_j^2,$$

де $C -$ поточний вузол;

$p_j -$ ймовірність класу j у вузлі C.

Далі в алгоритмі C4.5 для всіх атрибутів обчислюється теоретико-інформаційний критерій. Обирається атрибут, що максимізує даний вираз. Цей атрибут перевіряється в поточному вузлі дерева, а потім за цим атрибутом виконується подальша побудова дерева. Тобто у вузлі буде перевірятися значення за цим атрибутом і подальший рух по дереву буде здійснюватися в залежності від характеру отриманої відповіді.

Такі ж міркування застосовують стосовно отриманих підмножин $T_1,...,T_n$, а процес побудови дерева здійснюють рекурсивно до тих пір, поки у вузлі не виявляться приклади з одного класу.

Якщо в процесі роботи алгоритму отримано вузол, асоційований з порожньою множиною (тобто жоден приклад не потрапив в даний вузол), то він позначається як лист, а як рішення обирається клас, який найчастіше зустрічається у безпосереднього предка цього листа.

1.1.2 Кластерний аналіз

В 1939 р. Тріон (Tryon) ввів новий термін “кластерний аналіз”. На відміну від задач класифікації, кластерний аналіз не вимагає апріорних припущень стосовно даних. Задачу кластеризації об'єктів можна розглядати як процес виявлення в багатовимірній матриці даних істотного порядку, завдяки чому стає можливим виділення кластерів – "щільних" скупчень об'єктів, що досліджуються. Застосування кластерного аналізу паралельно розвивалось в декількох напрямах: в

біології, психології та інших науках, а тому більшість методів має по дві та більше назв. На сьогодні кластерний аналіз містить більше 100 різних алгоритмів [1].

Для більш інформативного представлення табличних даних застосовують діаграми розсіювання. На рисунку 1.4 наведено приклад діаграми розсіювання. Об'єкти (приклади), які за значеннями (факторами) "схожі" один на одного, належать до одного кластеру (групи).

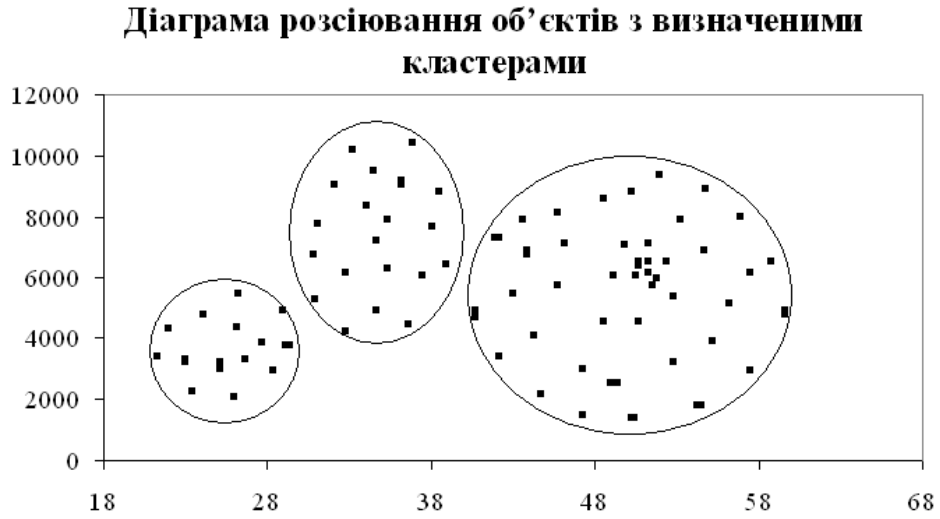


Рисунок 1.4 – Приклад діаграми розсіювання об'єктів

Критерієм визначення подібності та розбіжності кластерів є відстань між точками на діаграмі розсіювання. Для визначення міри відстані між кластерами (міри близькості) застосовують Манхеттенську відстань (хеммінгове або сіті-блокове), Чебишева, Мінковського, Блока, коефіцієнт кореляції Пірсона (у випадку двох змінних), процент неузгодженості (у випадку категоріальних даних), квадратична відстань Евкліда та інші. Але серед найвідоміших – відстань Евкліда, яка обчислюється за формулою:

$$D = \sqrt{(x_{1_i} - x_{1_j})^2 + \ldots + (x_{N_i} - x_{N_j})^2},$$

де $x_i^T = (x_{1_i}, \ldots, x_{N_j})$ та $x_j^T = (x_{1_j}, \ldots, x_{N_j})$ – координати $i-$го та $j-$го об'єктів відповідно у $N-$вимірному просторі.

Необхідно зауважити, що при цьому змінні повинні вимірюватися у шкалах, які зрівнюються між собою. Вибір масштабу в кластерному аналізі має велике

значення. Нехай для об’єктів A та B, ознака x_1 перебуває у діапазоні від 100 до 700, а ознака x_2 у діапазоні від 0 до 1. При розрахунку величини відстані між точками на діаграмі розсіювання ознака x_1 буде практично повністю домінувати над ознакою x_2.

Для розв’язання цієї задачі виконують стандартизацію змінних або нормування, тобто всі змінні приводять до єдиного діапазону. Найбільш розповсюджені такі види стандартизації:

1. Приведення змінних до діапазону значень від -1 до 1, за допомогою лінійного перетворення.

2. Приведення змінних до діапазону значень від 0 до 1.

3. Максимум = 1; значення змінних ділять на їхній максимум.

4. Середнє = 1; значення змінних ділять на їх середнє.

5. Стандартне відхилення = 1; значення змінних ділять на стандартне відхилення.

6. *Z*-шкали; від значень змінних віднімається їх середнє і ці значення ділять на стандартне відхилення.

Кожний кластер має такі математичні характеристики: центр, радіус, середньоквадратичне відхилення і розмір кластера.

Центр кластера – це середнє геометричне місце об’єктів (точок на діаграмі розсіювання) у просторі змінних.

Радіус кластера – максимальна відстань точок (об’єктів) на діаграмі розсіювання від центра кластера.

Інколи виникає ситуація, коли появляється перекриття кластерів. У цьому випадку неможливо за допомогою математичних процедур однозначно віднести об'єкт до одного із двох кластерів. Такі об'єкти називають спірними.

Спірний об'єкт – це об'єкт, який у міру подібності може бути віднесений до декількох кластерів.

Розмір кластеру може бути визначений або за радіусом кластера, або за середньоквадратичним відхиленням об'єктів для цього кластера. Об'єкт

відноситься до кластера, якщо відстань від об'єкта до центра кластера менша або дорівнює радіусу кластера. Якщо ця умова виконується для двох і більше кластерів, об'єкт є спірним. Неоднозначність таких випадків може бути усунута експертом або аналітиком.

Методи кластерного аналізу поділяються на ієрархічні та неієрархічні.

До ієрархічних методів кластерного аналізу, в свою чергу, відносять такі:

1. Ієрархічні агломеративні методи (AGNES – agglomerative nesting). Цей клас методів характеризується послідовним об'єднанням вихідних елементів і відповідним зменшенням числа кластерів.

2. Ієрархічні дивізівні методи (DIANA – divisive analysis). Ця група методів є логічною протилежністю агломеративним методам. На початку роботи алгоритму всі об'єкти належать одному кластеру, який на наступних кроках поділяється на менші кластери.

3. Ієрархічні алгоритми, пов'язані з побудовою дендрограм (від грецького *dendron* – дерево), які є результатом ієрархічного кластерного аналізу. Дендрограма описує близькість окремих точок і кластерів один до одного на діаграмі розсіювання. Дендрограма містить n рівнів, кожний з яких відповідає одному із кроків процесу послідовного укрупнення кластерів.

Розглянемо методи кластерного аналізу.

Після того, як кожний об’єкт відносять до певного кластеру, виникає запитання – як визначити відстані між кластерами? Існують різні правила, які називають методами об'єднання або зв'язку для двох кластерів [3].

До ієрархічних методів належать (таблиці 1.2–1.6):

– Зв'язок між групами (between-groups linkage). В цьому методі як міру відстані між кластерами застосовують середнє значення всіх відстаней між всіма можливими парами об’єктів з обох кластерів. Інформація, необхідна для знаходження міри, обчислюється на основі всіх теоретично можливих пар спостережень.

– Зв'язок всередині груп (within-groups linkage). Міра відстані між двома кластерами розраховується на підставі всіх можливих пар спостережень, що

належать обом кластерам, при цьому враховуються також пари спостережень, які утворюються всередині кластерів.

Таблиця 1.2 – Таблиця похибок ієрархічних методів кластеризації із використанням квадратичної міри Евкліда

Міра відстані між кластерами – квадратична Евклідова				
Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
Зв'язок між групами	174	93	267	0.92
Зв'язок всередині груп	123	490	613	0.817
Ближнього сусіда	178	17	195	0.942
Віддалених сусідів	174	94	268	0.92
Зважений центроїдний	174	93	267	0.92
Медіан	153	207	360	0.892
Варда	146	296	442	0.868

Таблиця 1.3 – Таблиця похибок ієрархічних методів кластеризації із використанням простої міри Евкліда

Міра відстані між кластерами – Евклідова				
Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
Зв'язок між групами	174	93	267	0.92
Зв'язок всередині груп	125	466	591	0.823
Ближнього сусіда	178	17	195	0.943
Віддалених сусідів	174	94	268	0.92
Зважений центроїдний	174	93	267	0.92
Медіан	175	78	253	0.924
Варда	123	520	643	0.808

Таблиця 1.4 – Таблиця похибок ієрархічних методів кластеризації із використанням коефіцієнта Пірсона

Міра відстані між кластерами – коефіцієнт кореляції Пірсона				
Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
Зв'язок між групами	177	26	203	0.939
Зв'язок всередині груп	171	370	541	0.838
Ближнього сусіда	178	1	179	0.947
Віддалених сусідів	177	71	248	0.926
Зважений центроїдний	163	772	935	0.721
Медіан	177	26	203	0.939
Варда	163	876	1039	0.69

Таблиця 1.5 – Таблиця похибок ієрархічних методів кластеризації із використанням міри Чебишева

Міра відстані між кластерами – Чебишева				
Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
Зв'язок між групами	174	93	267	0.92
Зв'язок всередині груп	119	502	621	0.814
Ближнього сусіда	178	17	195	0.942
Віддалених сусідів	133	425	558	0.833
Зважений центроїдний	174	93	267	0.92
Медіан	172	119	291	0.913
Варда	78	1194	1272	0.62

Таблиця 1.6 – Таблиця похибок ієрархічних методів кластеризації із використанням міри Мінковича

Міра відстані між кластерами – Мінковича				
Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
Зв'язок між групами	174	93	267	0.92
Зв'язок всередині груп	125	466	591	0.823
Ближнього сусіда	178	17	195	0.942
Віддалених сусідів	174	94	268	0.92
Зважений центроїдний	174	93	267	0.92
Медіан	175	78	253	0.924
Варда	123	520	643	0.808

– Метод ближнього сусіда (nearest neighbor) або одиночний зв'язок. Відстань між двома кластерами визначається відстанню між двома найбільш близькими об'єктами (найближчими сусідами) у різних кластерах. Цей метод дозволяє виділяти кластери як завгодно складної форми за умови, що різні частини таких кластерів з'єднані ланцюжками близьких один до одного елементів.

– Метод найбільш віддалених сусідів (furthest neighbor) або повний зв’язок. Відстані між кластерами визначаються найбільшою відстанню між будь-якими двома об'єктами в різних кластерах.

– Незважений центроїдний метод або метод незваженого попарного центроїдного усереднення (unweighted pair-group method using the centroid average). Як відстань між двома кластерами в цьому методі береться відстань між їхніми центрами ваги.

– Зважений центроїдний метод або метод зваженого попарного центроїдного усереднення (WPGMC – weighted pair-group method using the centroid average). Цей метод схожий на попередній, але для врахування різниці між розмірами кластерів (числа об'єктів у них), використовуються вагові коефіцієнти.

– Метод медіан (median clustering) – той же центроїдний метод, але центр

об'єднаного кластера обчислюється як середнє всіх об'єктів.

– Метод Варда (Ward's method). За відстань між кластерами береться приріст суми квадратів відстаней об'єктів до центрів кластерів, яку одержують в результаті їх об'єднання. На відміну від інших методів кластерного аналізу, для оцінювання відстаней між кластерами, тут використовують методи дисперсійного аналізу. На кожному кроці алгоритму об’єднують такі два кластери, що приводять до мінімального збільшення цільової функції, тобто внутрігрупової суми квадратів. Цей метод спрямований на об'єднання близько розташованих кластерів і “прагне” створювати кластери малого розміру.

– Метод незваженого попарного арифметичного середнього (unweighted pair-group method using arithmetic averages) [3]. За відстань між двома кластерами береться середня відстань між всіма парами об'єктів у них.

– Метод зваженого попарного арифметичного середнього (WPGMA – weighted pair-group method using arithmetic averages). Цей метод схожий на метод незваженого попарного середнього, але в якості вагового коефіцієнта використовується розмір кластера (число об'єктів, що належать кластеру).

У випадку великої кількості спостережень використовують неієрархічні методи, що представляють собою ітеративні методи. Нові кластери утворюються завдяки дробленню вихідної сукупності до тих пір, поки не буде виконане правило зупинки. Така неієрархічна кластеризація складається в результату поділу набору даних на певну кількість окремих кластерів. Існує два підходи. Перший полягає у визначенні границь кластерів як найбільш щільних ділянок у багатовимірному просторі вихідних даних, тобто кластер визначається там, де є велике “згущення об’єктів”. Другий підхід полягає в мінімізації міри розходження об'єктів (таблиця 1.7).

Алгоритм *k*-середніх (*k*-means) або алгоритм швидкого кластерного аналізу. Повний опис алгоритму можна знайти в роботі Хартігана та Вонга. На відміну від ієрархічних методів, які не вимагають попередніх припущень щодо числа кластерів, для отримання можливості використання цього методу необхідно висунути гіпотезу стосовно найбільш ймовірної кількісті кластерів.

Таблиця 1.7 – Таблиця похибок неієрархічних методів кластеризації

Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
K-середніх для ітерацій та класифікації	116	564	680	0.797
K-середніх тільки для класифікації	172	119	291	0.914
Двокроковий	78	1245	1323	0.605

Алгоритм k-середніх будує k кластерів, розташованих на як можливо більших відстанях один від одного. Вибір числа k може грунтуватися на результатах попередніх досліджень, теоретичних міркуваннях або інтуїції.

Загальна ідея алгоритму така: спостереження узгоджуються з кластерами таким чином, щоб середні значення кластерів (для всіх змінних) максимально можливо відрізнялися один від одного.

Алгоритм k-медіан (PAM – partitioning around medoids) – модифікація алгоритму k-середніх. На відміну від k-середніх цей алгоритм менш чутливий до шумів і викидів даних, оскільки на медіану викиди даних майже не впливають.

Алгоритм WaveCluster – алгоритм кластеризації на основі хвильових перетворень. На початку роботи алгоритму дані узагальнюють шляхом накладання на простір даних багатовимірної решітки.

На подальших кроках алгоритму аналізуються не окремі об’єкти, а узагальнені характеристики об’єктів, що потрапили до одного осередку решітки. На наступних кроках для визначення кластерів застосовується хвильове перетворення до узагальнених даних.

Двокроковий метод.

Алгоритм BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) запропонований Жангом (Zhang) в роботі [4]. У цьому алгоритмі реалізовано двокроковий процес кластеризації.

На першому етапі формується попередній набір кластерів. На другому етапі

по відношенню до виявлених кластерів застосовують інші алгоритми кластеризації. В [4] наведена аналогія, що описує цей алгоритм. Якщо кожен елемент даних уявити собі як бусинку, що лежить на поверхні стола, то кластери бусинок можна “замінити” тенісними кульками і перейти до більш докладного вивчення кластерів тенісних кульок. Кількість бусинок може виявитися досить великою, однак діаметр тенісних кульок можна підібрати таким чином, щоб на другому етапі можна було визначити дійсну складну форму кластерів, застосувавши традиційні алгоритми кластеризації.

1.1.3 Факторний аналіз

Факторний аналіз – сукупність методів багатовимірного статистичного аналізу, які застосовуються до вивчення взаємозв'язків між значеннями змінних. Він виник на початку XX століття в психології. За допомогою факторного аналізу можливе виявлення прихованих (латентних) змінних (факторів), що відповідають за наявність лінійних статистичних зв’язків (кореляцій) між спостережуваними змінними.

Цілі факторного аналізу:

1 – скорочення числа змінних;

2 – визначення взаємозв'язків між змінними, їхня класифікація.

Факторний аналіз може використовуватися як для вирішення завдань скорочення розмірності даних, так і для вирішення завдань класифікації. Критерії або головні фактори, виділені в результаті факторного аналізу, містять у стислому вигляді інформацію про існуючі зв’язки між змінними. За допомогою факторного аналізу велике число змінних зводиться до меншого числа незалежних величин, які називаються факторами. Фактор, в “стислому” вигляді, містить інформацію про декілька змінних. В одному факторі поєднуються змінні, що сильно корелюють між собою.

1.1.3.1 Послідовність виконання факторного аналізу

Спочатку відбувається стандартизація заданих значень змінних (наприклад

за z – перетворенням); потім за допомогою стандартизованих значень розраховують кореляційні коефіцієнти Пірсона між заданими змінними. Вихідним елементом для подальших обчислень є кореляційна матриця. Для цієї кореляційної матриці визначають власні значення і відповідні їм власні вектори, для визначення яких використовують оціночні значення діагональних елементів матриці (відносні дисперсії простих факторів).

Власні значення сортують за зменшенням; зазвичай обирають стільки факторів, скільки є власних значень, більших за одиницю. Власні вектори, що відповідають цим власним значенням, утворюють фактори. Елементи власних векторів називаються факторними навантаженнями. Їх можна розуміти як коефіцієнти кореляції між відповідними змінними і факторами.

Для вирішення задачі визначення факторів існують такі методи факторного аналізу:

1 – метод головних компонентів (principal component analysis);

2 – незважений метод найменших квадратів (unweight least square);

3 – узагальнений метод найменших квадратів (generalized least square);

4 – метод максимальної правдоподібності (maximum likelihood);

5 – факторизація головної осі (principal axis factoring);

6 – альфа факторизація (alpha factoring);

7 – факторизація образу (image factoring).

Найбільш відомим є метод головних компонентів.

1.1.3.2 Задача обертання факторів.

Описаний порядок виконання факторного аналізу не дає однозначного розв’язку задачі визначення факторів. Пошук однозначного рішення називають задачею обертання факторів.

Серед існуючих методів обертання факторів найбільш відомі:

1 – ортогональне обертання *varimax*, згідно з яким відбувається мінімізація кількості змінних з високим факторним навантаженням;

2 – ортогональне обертання *quartimax*, при якому відбувається мінімізація кількості факторів, необхідних для пояснення змінної;

3 – ортогональне обертання *equamax* – компроміс між попередніми двома методами;

4 – косокутне обертання – *direct oblimin;*

5 – метод *promax* – комбінація ортогонального та косокутного.

Найбільш розповсюдженим є метод варімаксу (varimax), оскільки він полегшує інтерпретацію факторів. Альтернативою прямокутному (ортогональному) обертанню є косокутне обертання. У цьому випадку після обертання осі не зберігають прямий кут по відношенню одна до іншої. Якщо при прямокутному обертанні кореляція між факторами відсутня, то при косокутному обертанні цей принцип порушується – фактори можуть корелювати між собою.

1.1.4 Ієрархічний кластерний аналіз із попереднім факторним аналізом

Кластеризацію даних можна виконувати відносно факторів. Для цього одним з методів факторного аналізу скорочують кількість змінних процесу, а потім застосовують один з ієрархічних методів.

Для вирішення задачі скорочення кількості змінних в задачі кластеризації клієнтів банку застосовано найбільш популярний метод факторного аналізу – метод головних компонентів. При цьому для вирішення задачі обертання факторів застосовано метод варімаксу. В таблиці 1.8 наведені значення отримані значення оберненої матриці. За цими значенням можна визначити з яких змінних (атрибутів) складається кожен з факторів. Фактор-1 складається з змінних “стать”, “вік”, “сімейний стан” та “тип трудозайнятості”. Фактор-2 з змінних “сімейний стан”, “кількість дітей” та “чоловік (дружина) працює”. Фактор-3 з змінних “освіта”, “поручитель” та “сума кредиту”.

В таблицях 1.9–1.11 наведені отримані значення похибок ієрархічних методів кластеризації із попереднім факторним аналізом. Для обчислення відстані між кластерами застосована квадратична міра Евкліда. Для вирішення задачі обертання факторів – метод варімаксу (див. таблицю 1.8). Для обчислення значень факторів за атрибутами – метод регресії (див. таблицю 1.9), Бартлета (Bartlett) (див. таблицю 1.10) та Андерсона-Рубіна (Anderson-Rubin) (див. таблицю 1.11).

Таблиця 1.8 – Таблиця значень оберненої матриці компонентів, отримана за методом варімаксу з нормалізацією Кайзера

Компоненти	Фактор		
	1	2	3
Стать	-0.538	0.085	-0.223
Вік	0.811	0.171	-0.036
Сімейний стан	0.577	0.566	-0.024
Кількість дітей	-0.153	0.769	-0.033
Чоловік (дружина) працює	0.093	0.697	-0.014
Освіта	-0.158	-0.075	0.537
Тип трудозайнятості	0.543	-0.266	-0.369
Поручитель	0.121	-0.141	0.529
Сума кредиту	0.085	0.181	0.626

Таблиця 1.9 – Таблиця похибок ієрархічних методів

Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
Зв'язок між групами	177	37	214	0.936
Зв'язок всередині груп	107	602	709	0.788
Ближнього сусіда	178	1	179	0.947
Віддалених сусідів	173	188	361	0.892
Зважений центроїдний	178	107	285	0.915
Медіан	166	276	442	0.868
Варда	94	754	848	0.747

Таблиця 1.10 – Таблиця похибок ієрархічних методів

Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
Зв'язок між групами	177	37	214	0.936

Продовження таблиці 1.10

Зв'язок всередині груп	107	802	909	0.728
Ближнього сусіда	178	1	179	0.947
Віддалених сусідів	173	188	361	0.892
Зважений центроїдний	178	107	285	0.915
Медіан	186	276	462	0.862
Варда	94	754	848	0.7467

Таблиця 1.11 – Таблиця похибок ієрархічних методів

Назва методу	Похибка			Точність моделі
	1-го роду	2-го роду	Загальна	
Зв'язок між групами	177	37	214	0.936
Зв'язок всередині груп	107	602	709	0.788
Ближнього сусіда	178	1	179	0.947
Віддалених сусідів	173	188	361	0.892
Зважений центроїдний	178	107	285	0.915
Медіан	166	276	442	0.868
Варда	94	754	848	0.747

На рисунку 1.5 наведено діаграму розсіювання даних до кластеризації у розрізі “вік-сума кредиту”. Відносно даних виконана операція зменшення кількості змінних за методом факторного аналізу – методом головних компонентів. Після цього за методом "Ближнього сусіда" виконана кластеризація об’єктів (клієнтів банку). На рисунку 1.6 наведена діаграма розсіювання вірно та невірно класифікованих клієнтів.

Коректне застосування методів кластерного аналізу щодо вирішення задач класифікації та кластеризації ускладнюється різмаїттям існуючих комбінації алгоритмів та методик. За отриманими результатами обчислювальних експериментів, наведених в таблицях 1.2–1.7 та 1.9–1.11, визначено найкращий метод. За критерієм загальної точності моделі для класифікації

кредитоспроможності клієнтів банку найкращим виявився метод ближнього сусіда з використанням як міри відстані коефіцієнта кореляції Пірсона або із попереднім факторним аналізом за методом головних компонентів. Більш-менш прийнятна загальна точність моделі одержана при застосуванні методів зв'язку між групами, найбільш віддалених сусідів, зваженого центроїдного, медіан та *к*-середніх. Низьку точність показали методи зв'язку всередині груп та Варда, а найгіршим виявився двокроковий метод.

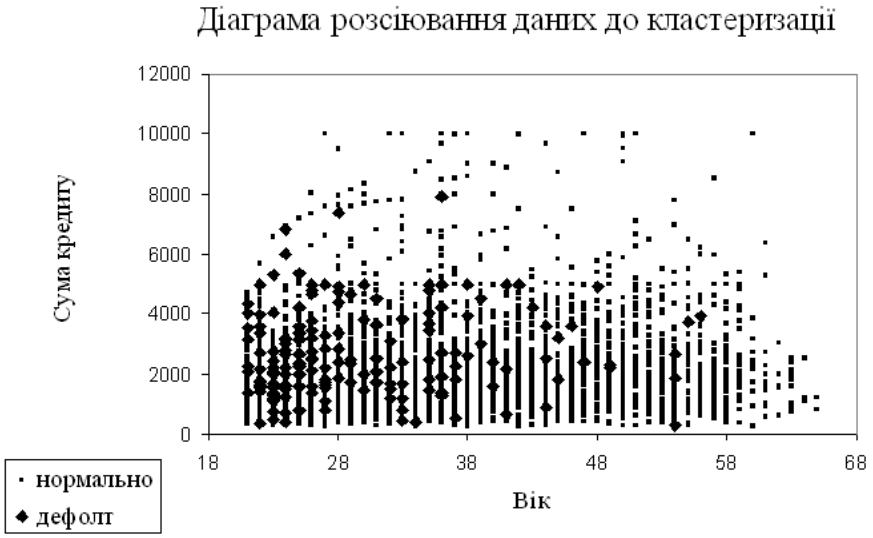


Рисунок 1.5 – Діаграма розсіювання клієнтів банку за віком та сумою кредиту, до кластеризації

Рисунок 1.6 – Діаграма розсіювання за віком та сумою кредиту вірно та невірно класифікованих клієнтів за методом "Ближнього сусіда" із попереднім факторним аналізом за методом головних компонентів

1.1.5 Штучні нейронні мережі

Одним з широковідомих методів ІАД є штучні нейронні мережі (ШНМ). Незважаючи на велику розмаїтість варіантів існуючих нейронних мереж, всі вони мають загальні риси. Всі вони складаються з великого числа зв'язаних між собою однотипових елементів – нейронів, які імітують нейрони головного мозку. На рисунку 1.7 показана схема нейрона для задачі оцінювання кредитоспроможності.

З рисунку 1.7 видно, що штучний нейрон складається із сінапсів, що зв'язує входи нейрона з ядром; ядра нейрона, яке здійснює обробку вхідних сигналів, і аксона, що зв'язує нейрон з нейронами наступного шару. Кожен сінапс має ваговий коефіцієнт $w_1,...,w_9$, який визначає, наскільки відповідний вхід нейрона впливає на його стан. Стан нейрона визначається за формулою:

$$S = \sum_{i=1}^{n} x_i \cdot w_i \,. \qquad (1.1)$$

де n – кількість входів нейронів,

x_i – значення i – го входу нейрону;

w_i – вага i – го сінапса.

Після цього обчислюється значення аксона нейрона за формулою:

$$Y = f(\quad),$$

де f – деяка функція, що називається активаційною.

Найчастіше, як активаційна функція, використовується так званий сігмоїд, що має такий вигляд: $f(x) = \dfrac{1}{1+e^{-\alpha \cdot x}}$, де α – параметр логістичної функції. Параметр α впливає на пологість сігмоїда та зберігається у локальній пам’яті нейрона разом з вагами входів. Основна перевага цієї функції полягає в тому, що вона диференційована на всій осі абсцис і має нескладну похідну:

$$f'(x) = \alpha \cdot f(x) \cdot (1 - f(x)).$$

При зменшенні параметра α сігмоїд стає пологішим, вироджуючись у горизонтальну лінію на рівні 0,5 при $\alpha = 0$. При збільшенні α сігмоїд швидко наближається до функції одиничного стрибка.

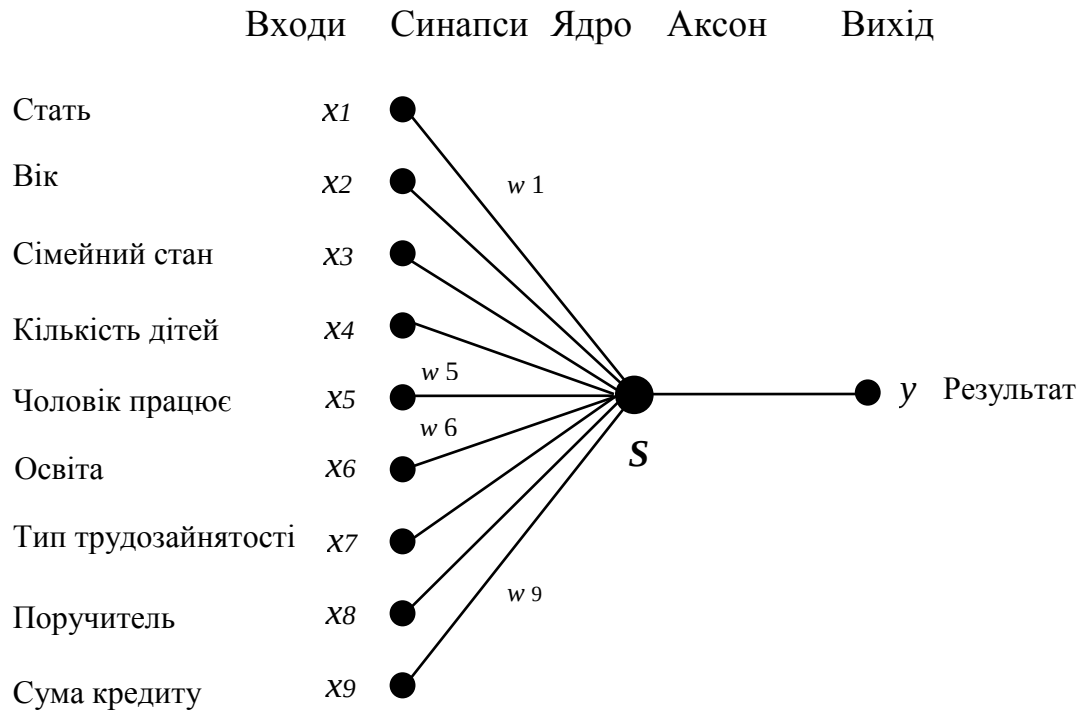


Рисунок 1.7 – Схема нейрона

Розглянемо нейронні мережі зворотного розповсюдження.

Назва цього типу ШНМ походить від back propagation завдяки використанню алгоритму навчання, у якому похибка поширюється від вихідного шару до вхідного, тобто в напрямі, протилежному напряму поширення сигналу при нормальному функціонуванні мережі [5].

Нейронна мережа зворотного поширення складається з декількох шарів нейронів, при цьому кожен нейрон $i-$го шару пов'язаний з кожним нейроном $i+1$ шару, тобто мова йде про повнозв’язні ШНМ.

У загальному випадку задача навчання ШНМ зводиться до знаходження функціональної залежності $Y = F(X)$ де $X-$ вхідний, а $Y-$ вихідний вектори. Таке завдання, при обмеженому наборі вхідних даних, має нескінченну кількість рішень. Для обмеження простору пошуку при навчанні ставиться задача мінімізації цільової функції похибки ШНМ із використанням методу найменших квадратів (МНК):

$$E(w) = \frac{1}{2} \cdot \sum_{j=1}^{p} (y_j - d_j)^2,$$

де y_j – значення j – го виходу ШНМ;

d_j – цільове значення j – го виходу;

p – кількість нейронів у вихідному полі.

Навчання ШНМ виконується за методом градієнтного спуску, тобто на кожній ітерації зміна ваги виконується за формулою:

$$\Delta w_{ij} = -\eta \cdot \frac{\partial E}{\partial w_{ij}}, \tag{1.2}$$

де η – параметр швидкості навчання;

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial S_j} \cdot \frac{\partial S_j}{\partial w_{ij}}, \tag{1.3}$$

де y_j – значення j – го виходу ШНМ;

S_j – зважена сума вхідних сигналів обчислена за формулою (1.1).

При цьому множник $\frac{\partial S_j}{\partial w_{ij}} = x_i$, де x_i – значення i – го входу нейрону. Перший множник формули (1.3) можна записати в наступній формі:

$$\frac{\partial E}{\partial y_i} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial S_k} \cdot \frac{\partial S_k}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{\partial y_k}{\partial S_k} \cdot w_{jk}^{(n+1)},$$

де k – кількість нейронів в $n+1$ шарі.

Якщо ввести допоміжну змінну $\delta_j^{(n)} = \frac{\partial E}{\partial y_j} \cdot \frac{\partial y_j}{\partial S_j}$, то рекурентна формула для визначення n – го шару на основі інформації про $n+1$ шари записується так:

$$\delta_j^{(n)} = \left[\sum_k \delta_k^{(n+1)} \cdot w_{jk}^{(n+1)} \right] \cdot \frac{\partial y_j}{\partial S_j}, \tag{1.4}$$

обчислення значень останнього шару робиться за формулою:

$$\delta_j^{(N)} = \left(y_i^{(N)} - d_i \right) \cdot \frac{\partial y_i}{\partial S_i}. \tag{1.5}$$

Тепер формула (1.2) запишеться так:

$$\Delta w_{ij}^{(n)} = -\eta \cdot \delta_j^{(n)} \cdot x_j^n. \tag{1.6}$$

Повний алгоритм навчання мережі має вигляд:

1. Подати на вхід ШНМ один з образів і визначити значення виходів нейронів нейронної мережі.

2. Обчислити значення для вихідного шару ШНМ за формулою (1.5) та обчислити зміну ваг вихідного шару N за формулою (1.6).

3. Обчислити значення за формулами (1.6) і (1.4) відповідно, а також $w_{ij}^{(N)}$ для інших шарів ШНМ, $n = N-1,...,1$.

4. Скорегувати всі ваги ШНМ за формулою:

$$w_{ij}^{(n)}(t) = w_{ij}^{(n)}(t-1) + \Delta w_{ij}^{(n)}(t).$$

5. Якщо помилка неприйнятна, то необхідно перейти на 1-й крок.

Розглянемо алгоритм resilient propagation.

Одним з серйозних недоліків алгоритму зворотного розповсюдження є занадто довгий за часом процес навчання. Існує кілька алгоритмів, що дозволяють прискорити процес навчання, наприклад: (1) QuickProp, (2) метод спряжених градієнтів, (3) Левенберга-Маркара, (4) Resilient Propagation та інші. Як альтернативу алгоритму зворотного розповсюдження серед методів ШНМ в даному підрозділі розглянуто метод Resilient Propagation, який був запропонований в 1993 році Рідміллером (Riedmiller) та Брауном (Braun).

На відміну від стандартного алгоритму зворотного розповсюдження, Resilient Propagation використовує тільки знаки частинних похідних для підстроювання вагових коефіцієнтів. Алгоритм використовує так зване “навчання за епохами”, коли корекція ваг відбувається після пред'явлення мережі всіх прикладів з навчальної вибірки.

Для визначення величини корекції використовується правило:

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^{+}\Delta_{ij}^{(t)}, & \dfrac{\partial E^{(t)}}{\partial w_{ij}} \cdot \dfrac{\partial E^{(t-1)}}{\partial w_{ij}} > 0 \\ \eta^{-}\Delta_{ij}^{(t)}, & \dfrac{\partial E^{(t)}}{\partial w_{ij}} \cdot \dfrac{\partial E^{(t-1)}}{\partial w_{ij}} < 0 \end{cases}. \qquad (1.7)$$

$$0 < \eta^{-} < 1 < \eta^{+}$$

Якщо на поточному кроці частинна похідна для відповідної ваги w_{ij} змінила свій знак, то остання зміна була значимою, і алгоритм проскочив локальний мінімум, отже величину зміни необхідно зменшити на η^- та повернути попереднє значення вагового коефіцієнта, фактично необхідно зробити "відкат":

$$\Delta w_{ij}(t) = \Delta w_{ij}(t) - \Delta_{ij}^{(t-1)}. \tag{1.8}$$

Якщо знак часткової похідної не змінився, то потрібно збільшити величину корекції на η^+ для досягнення більше швидкої збіжності. Зафіксувавши множники η^- та η^+, можна відмовитися від глобальних параметрів настроювання ШНМ. Рекомендовані значення для $\eta^- = 0{,}5$ та $\eta^+ = 1{,}2$, але немає ніяких обмежень на використання інших значень для цих параметрів.

Для того щоб не допустити занадто великих або малих значень ваг, величину корекції обмежують зверху максимальним $\Delta_{\max}$ а знизу мінімальними $\Delta_{\min}$ значеннями величини корекції, які, за замовчуванням, встановлюють рівними $\Delta_{\max} = 50$ та $\Delta_{\min} = 10^{-6}$. Початкові значення для всіх Δ_{ij} рекомендується встановлювати такими: $\Delta_{ij} = 0{,}1$.

Для обчислення значення корекції ваг використається наступна формула:

$$\Delta w_{ij}(t) = \begin{Bmatrix} -\Delta_{ij}^{(t)}, & \dfrac{\partial E^{(t)}}{\partial w_{ij}} > 0 \\ +\Delta_{ij}^{(t)}, & \dfrac{\partial E^{(t)}}{\partial w_{ij}} < 0 \\ 0, & \dfrac{\partial E^{(t)}}{\partial w_{ij}} = 0 \end{Bmatrix}. \tag{1.9}$$

Якщо похідна додатна, тобто помилка зростає, то ваговий коефіцієнт зменшується на величину корекції, інакше – збільшується. Після цього корегуються ваги:

$$w_{ij}(t+1) = w_{ij}(t) + \Delta w_{ij}(t). \tag{1.10}$$

Алгоритм:

1. Ініціалізувати величину корекції Δ_{ij}.

2. Пред'явити всі приклади з вибірки і обчислити частинні похідні.

3. Обчислити нове значення Δ_{ij} за формулами 1.9 та 1.7.

4. Скорегувати ваги за формулою 1.10.

5. Якщо умова зупинки не виконана, то перейти до кроку 2.

Даний алгоритм сходиться в 4-5 разів швидше у порівнянні із стандартним алгоритмом зворотного розповсюдження.

1.1.6 Бінарні регресійні моделі дискретного вибору

У 1805 році опублікована перша робота, присвячена методу найменших квадратів, з якої розпочався формальний розвиток регресійного аналізу, хоча насправді метод був запропонований Гауссом ще в 1795 році. Моделі дискретного вибору призначені для пояснення дискретного вибору результату, тобто для розв'язання задачі оцінювання кредитоспроможності позичальника. Це означає, що $y = 1$ (клієнт кредитоспроможний) якщо $probability \geq 0{,}5$ або $y = 0$ (дефолт) в інших випадках. Коефіцієнти наведених нижче рівнянь обчислені із використанням системи Eviews 3.

Далі в рівняннях використані такі змінні: x_1 – стать; x_2 – вік; x_3 – сімейний стан; x_4 – кількість дітей; x_5 – чоловік (дружина) працює; x_6 – освіта; x_7 – тип трудозайнятості; x_8 – поручитель; x_9 – сума кредиту; y – результат (кредитоспроможний?).

Лінійна імовірнісна модель (ЛІМ) представляє собою різновид множинної регресії, загальне призначення якої складається в аналізі зв'язку між декількома незалежними змінними (регрессорами або предикторами) і залежною змінною:

$$probability = 0{,}7545 - 0{,}0358 \cdot x_1 + 0{,}0024 \cdot x_2 + 0{,}01 \cdot x_3 + 0{,}0121 \cdot x_4 + \\ + 0{,}0214 \cdot x_5 + 0{,}0142 \cdot x_6 + 0{,}0037 \cdot x_7 + 0{,}1112 \cdot x_8 - 1{,}6732 \cdot x_9 .$$

Головний недолік лінійної імовірнісної моделі полягає в тому, що значення ймовірності виходить за діапазон від нуля до одиниці, тому замість неї найчастіше

використовують логіт та пробіт моделі.

Логіт модель. Ще в 1838 році бельгійський математик Верхулст (Verhulst) запропонував ідею використання логістичної функції для демографічних досліджень, але широкої популярності ця модель набула лише в 1925 році завдяки практичним дослідженням Юла (Yule), Перла (Pearl) та Ріда (Reed). Логіт-модель оцінювання кредитоспроможності фізичних осіб має вигляд:

$$probability = \frac{\exp(z)}{1+\exp(z)} = \frac{1}{1+(\exp(z))^{-1}} = \frac{1}{1+\exp(-z)},$$

де

$$z = -0{,}9563 \cdot x_1 + 0{,}0695 \cdot x_2 + 0{,}1426 \cdot x_3 + 0{,}1875 \cdot x_4 + \\ + 0{,}2264 \cdot x_5 + 0{,}2727 \cdot x_6 + 0{,}0509 \cdot x_7 + 3{,}0315 \cdot x_8 - 0{,}0005 \cdot x_9.$$

Пробіт модель запропонована в 1935 році біологом Бліссом (Bliss), але широкого поширення здобула лише на початку 70-х років минулого століття завдяки поширенню мейнфреймів, які здатні вирішувати задачі нелінійної максимізації. Для задачі, що вирішується, логіт модель записується наступним чином:

$$probability = \int_{-\infty}^{z} \frac{1}{\sqrt{2 \cdot \pi}} \cdot \exp\left(\frac{-t^2}{2}\right) \cdot dt,$$

де

$$z = -0{,}2677 - 0{,}4469 \cdot x_1 + 0{,}035 \cdot x_2 + 0{,}0918 \cdot x_3 + 0{,}1097 \cdot x_4 + \\ + 0{,}1148 \cdot x_5 + +0{,}1436 \cdot x_6 + 0{,}1539 \cdot x_7 + 1{,}4139 \cdot x_8 - 0{,}0002 \cdot x_9.$$

1.2 Байєсівська мережа як інструмент інтелектуального аналізу даних

1.2.1 Виникнення байєсівських мереж

В четвертому столітті до нашої ери Аристотель (384-322 до н.е.) створив формальну логіку, яка на той час стала противагою діалектиці Платона (428-348 до н.е.). На протязі століть відбувалася еволюція логіки як науки, вона змінювалася та збагачувалася. В період раннього середньовіччя (7-11 сторіччя) антична логіка сприймалася через призму християнства. Пізніше, у 12-13 століттях, після того як

всі твори Аристотеля були канонізовані церквою, з’явилася несхоластична логіка. Перша спроба *машинізації процесу логічного висновку* була здійснена Раймондом Луллієм (1235-1315 р.). Виникнення теорії ймовірностей в середньовіччі, як науки із перших спроб математичного аналізу азартних ігор (орлянка, кості, рулетка). Леонардо да Вінчі (1452-1519 р.) та Френсіс Бекон (1561-1626 р.) заново винайшли античну ідею індукції та дедуктивного методу. В 17-19 століттях відбувалось поступове зародження дискретної математики як науки завдяки таким вченим як Блез Паска́ль (Blaise Pascal ,1623-1662), П'єр Ферма (Pierre de Fermat, 1601-1665), Жозе́ф-Луї́ Лагра́нж (Giuseppe Lodovico Lagrangia; 1736-1813) та Еварист Галуа (Évariste Galois, 1811-1831). Також суттєвий вплив на розвиток логіки здійснили Рене Декарт (1596-1650 р.), Готфрід Лейбніц (1646-1716 р.) та Джордж Буль (1815-1864 р.). В 20 сторіччі індуктивна логіка з одного боку трансформувалася в ймовірнісну логіку, а з іншого боку – вийшла за межі логіки у власному змісті, знайшовши в істотно збагаченому виді нове життя в сучасній математичній статистиці та теорії планування експерименту.

БМ з’явилися на стику двох наук: теорії ймовірностей та теорії графів (розділ дискретної математики), рисунок 1.8. Термін “байєсівська мережа” (Bayesian Network) був запропонований Джуді Перлом в 1985 році, з метою акцентування трьох аспектів [6]: об’єктивного природи вхідних даних; отримання достовірної інформації при застосуванні теореми Байєса; ідея застосування причин та наслідків, запропонована в 1763 році в посмертній роботі Томаса Байєса [7].

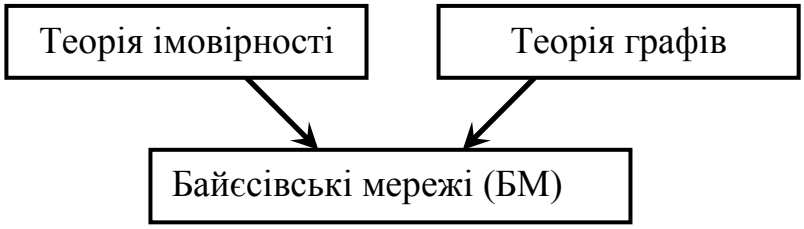


Рисунок 1.8 – БМ на стику двох наук

Преподобний Томас Байєс (1702–1761 роки) одним з перших зацікавився ймовірністю настання подій у майбутніх випробуваннях, грунтуючись на інформації про минулі випробування. Саме теорема Байєса пов’язує апріорні та

апостеріорні ймовірності причин після спостереження за наслідками. В 1763 році в посмертній роботі “Опис методу розв’язання задачі в рамках доктрини випадкових подій” (“An essay towards solving a problem in the doctrine of chances”), опублікованій його товаришем Ричардом Прайсом (Richard Price, 1723-1791) в п’ятдесят третьому номері лондонського журналу філософського королівського об’єднання [7], була вперше застосована ймовірність в індуктивному сенсі та встановлені основи ймовірнісного судження та висновку. Своє застосування теорема Байєса знайшла в теорії ймовірностей, одразу після опублікування в 1763 р.

Ідея впровадження БМ полягає в представленні причинно-наслідкових зв’язків процесу у вигляді графа. Треба зауважити, що ідея представлення причинно-наслідкових зв’язків у вигляді графів розглядалася набагато раніше. В 1921 р. біолог–генетик Сьюел Райт (Sewall Wright, 1889-1988 р.) запропонував пат-аналіз (path analysis) – статистичний метод, який дозволяє оцінити ступінь взаємовпливу змінних у причинно-наслідковій моделі. Для цього він об’єднав лінійні регресійні моделі та спрямовані графи, пізніше ця ідея була розвинута Гербертом Сімоном (Herbert Simon, 1916-2001) та Хьюбертом Балоком (Hubert Blalock).

До впровадження терміну “байєсівська мережа” Джуді Перлом в 1985 році, БМ застосовувалися під назвою каузальних мереж (causal network), тобто мережі з причинно-наслідковими зв’язками. Байєсівськими вони стали завдяки застосуванню в каузальних мережах теореми Байєса.

1.2.2 Переваги застосування байєсівських мереж

На відміну від інших методів ІАД, застосування байєсівських мереж для аналізу процесів різної природи, діяльності людини та функціонування технічних систем дозволяє враховувати та використовувати будь-які вхідні дані – експертні оцінки і статистичну інформацію. В свою чергу змінні можуть бути дискретними і неперервними, а характер їх надходження при аналізі та прийнятті рішення може бути як в режимі реального часу так і у вигляді статистичних масивів інформації і

баз даних. При цьому, завдяки використанню представлення взаємодії між факторами процесу у вигляді причино-наслідкових зв’язків в мережі, у порівнянні з іншими методами ІАД, досягаються максимально високий рівень візуалізації та, як наслідок, чітке розуміння суті взаємодії факторів процесу між собою. Іншими перевагами БМ є можливості врахування невизначеностей статистичного, структурного і параметричного характеру, а також формування висновку за допомогою різних методів – наближених і точних. Загалом, можна сказати, що БМ – це високоресурсний метод ймовірнісного моделювання процесів довільної природи з невизначеностями різних типів, який забезпечує можливість достатньо точного опису їх функціонування, оцінювати прогнози та будувати системи управління.

1.3 Постановка задачі

Завданням дипломного проекту постає розробка та реалізація інформаційної системи підтримки прийняття рішень для інтелектуального аналізу даних на основі байєсівських мереж. Для цього в роботі необхідно розв’язати такі задачі:

1. Розглянути існуючі типи БМ, а також виконати огляд сучасних підходів та методи щодо побудови топології і ймовірнісного висновку. Визначити найбільш прості та перспективні напрямки.

2. Розробити евристичний алгоритм побудови БМ за навчальними даними на основі ОМД функції. Реалізований алгоритм повинен бути не складнішим за існуючі аналоги.

3. Розробити алгоритм побудови ймовірнісного висновку в БМ, за навчальними даними. Програмна реалізація алгоритму повинна бути не гіршою існуючих аналогів за швидкодією та складністю застосування.

4. Розробити архітектуру інформаційної системи підтримки прийняття рішень оцінки кредитних ризиків на основі байєсівських мереж та реалізувати її програмно, ґрунтуючись на сучасних підходах щодо побудови топології та імовірнісного висновку в БМ.

2 СТРУКТУРА ТА АЛГОРИТМИ ПОБУДОВИ БАЙЄСІВСЬКОЇ МЕРЕЖІ

2.1 Побудова структури байєсівської мережі

Багато компаній роками накопичують бізнес-інформацію, сподіваючись, що вона допоможе їм у прийнятті рішень. Чим конкретніша інформація, тим кориснішою вона є для прийняття рішень. Інтелектуальний аналіз даних (Data Mining) – це технологія виявлення схованих взаємозв'язків всередині великих баз даних. Більшість інструментів інтелектуального аналізу даних грунтується на двох технологіях: машинне навчання (machine learning) і візуалізація (візуальне подання інформації). Байєсівські мережі якраз і поєднують у собі ці дві технології.

Цей розділ присвячено аналізу існуючих методів розв’язання задачі вибору оптимальної структури БМ. Оптимальною у тому сенсі, що обрана структура максимально правдоподібно відповідає процесу який моделюється, або навчальним даним. Взагалі в англомовній літературі в широкому сенсі розуміння термін "побудова БМ" означає реалізацію таких процесів:

1) пошук оптимальної структури БМ, тобто спрямованого ациклічного графа, що найбільш адекватно відповідає навчальним даним або досліджуваному процесу;

2) обчислення значень таблиць умовних ймовірностей БМ для вузлів цього графа.

Більшість існуючих методів побудови структури БМ можна умовно розділити на дві категорії [8]:

1) на основі оціночних функцій (search & scoring);

2) застосовуючи тест на умовну незалежність (dependency analysis). Більшість із існуючих методів зустрічаються з наступними проблемами:

1. Наявність впорядкованої множини вершин (ВМВ). У більшості методів, особливо старих, вважається, що ВМВ задана, але на практиці при роботі з реальними даними і це дуже часто не відповідає дійсності.

2. Низька обчислювальна ефективність. Деякі сучасні методи працюють без використання ВМВ, замість неї використовується тест на умовну незалежність

(ТУН). Але в цьому випадку необхідно виконати експоненціальну кількість таких тестів, що призводить до зменшення ефективності роботи методу у зв'язку із зростанням об'єму обчислювань.

3. Проблема побудови великих БМ. Існує декілька методів здатних побудувати структуру БМ з декілька сотень вершин, використовуючи навчальну вибірку з мільйонів записів. До таких методів відносяться Tetrad II та SopLeq.

2.1.1 Методи побудови байєсівських мереж

Методи на основі оціночних функцій (search & scoring)

Чу і Ліу (Chow and Liu) в 1968 році запропонували алгоритм для побудови БМ у вигляді дерева [60], заснований на використанні значень взаємної інформації між вершинами. В якості рішення метод видає структуру зі значенням спільного розподілу ймовірностей мережі, найбільш відповідного навчальним даним Побудова структури БМ здійснюється за $O(N^2)$ кроків, де N – кількість вершин мережі, однак цей алгоритм не працює для багатозв'язаних БМ.

В 1988 році Рібан і Перл (Rebane and Pearl) запропонували вдосконалений змінений алгоритм Чу і Ліу для побудови БМ у вигляді полі-дерева. Купер і Герскович (Cooper and Herskovits) в 1990 році розробили алгоритм Кутато (Kutato). На етапі ініціалізації алгоритму вважається, що всі вершини БМ незалежні, після чого обчислюється ентропія цієї мережі. Потім виконується додавання дуг між вершинами в мережі таким чином, щоб мінімізувати ентропію БМ. Для роботи алгоритму потрібна наявність ВМВ.

Купер і Герскович в 1992 році запропонували широко відомий алгоритм К2, який виконує пошук структури з максимальним значення функції Купера-Герсковича (КГ). Для роботи алгоритма потрібна наявність ВМВ.

В 1994 році був запропонований алгоритм HGC. Цей алгоритм суттєво відрізняється від інших алгоритмів застосовуючи оціночні функції, тим що вперше саме в ньому були використані два нових поняття:

1) параметричної модульності (parametric modularity);

2) рівнозначності подій (event equivalence). Інші дослідники досить довго не

використовували одночасно цих понять.

Але одночасне застосування цих понять дозволяє об'єднувати статистичну інформацію та експертних знань для побудові БМ.

Вонг і Ксіанг (Wong and Xiang) запропонували в 1994 році алгоритм для побудови Марковських мереж з використанням значення ентропії та I-map. Граф G ймовірнісної моделі M називають незалежною картою (independency map, скорочено I-map), якщо з незалежності вершин графа G випливає незалежність моделі M. Цей алгоритм дозволяє представити процес, який моделюється, у вигляді I-map і у випадку, коли мережа являється однозв'язною гарантовано будується БМ. Разом із Чу (Chu) Ксіанг розробив у 1997 році більш швидкий варіант запропонованого алгоритму.

Алгоритм Лема-Бахуса (Lam-Bacchus), запропонований в 1996 році, виконує евристичну побудову структури мережі, використовуючи значення взаємної інформації між вершинами, а в якості функції оцінки використається функція опису мінімальною довжиною (minimum description length).

Алгоритм Бенедикта (Benedict), запропонований в 1996 році, виконує евристичний пошук на основі ВМВ, аналізуючи умовні незалежності в структурі мережі на основі *d*-розділення, а в якості функції оцінки використається ентропія.

СВ алгоритм запропонований в 1995 році. Він використає ТУН між вершинами мережі, для побудови ВМВ. Для побудови структури мережі використається функція КГ.

Алгоритм Фрідмана-Голдшмідта (Friedman-Goldszmidt) запропонований в 1996 році. Для побудови мережі використається аналіз її локальних підструктур, а в якості функції оцінки використається функція опису мінімальною довжиною (ОМД) та оцінка Байєса.

В алгоритмі WKD, запропонованому в 1996 році, в якості функції оцінки при побудові мережі використається функція повідомлення мінімальної довжини (minimum message length), яка схожа на ОМД.

Алгоритм Сузукі (Suzuki), запропонований в 1999 році, заснований на методі гілок та границь (branch and bound method) для завдання послідовності побудови

структури мережі, а в якості функції оцінки використається ОМД.

Також існує ціла низка різноманітних поглинаючих алгоритмів (greedy algorithm) в яких для оцінювання можуть використовуватися різноманітні функції, наприклад максимальної правдоподібності або байєсівський інформаційний критерій [10].

Методи з використанням тестів на умовну незалежність (dependency analysis)

В 1983 році Вермут і Лоуренс (Wermuth and Lauritzen) запропонували алгоритм для побудови структури БМ застосовуючи ТУН. Цей алгоритм виконує послідовний перебір ВМВ. Для кожної пари вершин X_k та X_t, таких що $X_t < X_k$ (тобто X_k це предок відносно X_t), виконується обчислення значення умовної незалежності. Цей алгоритм гарантує побудову БМ по навчальним даним, але при цьому буде потрібно виконати велику кількість ТУН між вершинами, що можливо лише у випадку, коли мережа складається з невеликої кількості вершин.

В 1988 році Перл (Pearl) запропонував алгоритм побудови скінченого спрямованого ациклічного графа (boundary DAG algorithm). Цей алгоритм будує БМ маючи ВМВ та функцію спільного розподілу (або достатньо велику навчальну вибірку даних). Разом із будь-яким, не досить складним методом пошуку, цей алгоритм позбавлений проблеми, яка полягає в необхідності розрахунку великої кількості ТУН застосовуючи алгоритм Вермута і Лоуренса. Однак проблема в необхідності обчислення великої кількості ТУН з'являється при використанні цього алгоритму для побудови Марковських мереж, тобто мереж зі скритими вузлами (hidden node).

В 1990 році був запропонований SRA алгоритм, який являється модифікацією алгоритму скінченого спрямованого ациклічного графа. Цей алгоритм має менш жорсткі вимоги до ВМВ, для побудови БМ достатньо мати частково впорядковану множину вершин та ще деякі обмеження. Побудова БМ виконується послідовним додаванням дуг між вершинами, для цього використовується евристичний пошук. Але алгоритм виконує експоненціальну кількість розрахунків ТУН.

Алгоритм "Конструктор" (constructor algorithm) запропонований в 1990 році,

дуже схожий на алгоритм побудови скінченого спрямованого ациклічного графа. Він намагається замість БМ побудувати Марковську мережу. Відмінність цього метода від інших методів які використовують ТУН в тому що він не виконую надлишкові ТУН та йому не потрібна ВМВ.

Алгоритму SGS, запропонованому в 1990 році, для побудови структури не потрібна наявність ВМВ, але замість неї йому доводиться виконувати експоненціальну кількість тестів на умовну незалежність між вершинами.

PC алгоритм розроблений в 1991 році представляє собою вдосконалений варіант SGS алгоритму. Цей алгоритм спеціально розроблявся для побудови розріджених БМ (sparse BN), тобто для мереж з невеликою кількістю дуг між вершинами.

Алгоритм KDB, запропонований в 1996 році, для визначення напряму побудови мережі використовує ЗВІ. Як оціночна функція використовується функціонал, що мінімізує значення мережі.

Алгоритм FBC (full Bayesian network), запропонований в 2006 році, представляє собою удосконалений алгоритм KDB, який в якості функції оцінки при побудові мережі використовує функцію сумарних значень ЗВІ вершин.

Розглянемо інші методи.

Не завжди побудована структура БМ відповідає процесу який моделюється, інколи це пов’язано з неповнотою даних спостережень або недостатньою визначеністю предметної області. Замість побудови однієї найкращої структури БМ деякі алгоритми [11] в якості результату видають кілька мережевих структур.

Іноді дослідник може не мати всієї інформації про процес, який моделюється. Тобто деякі змінні, які впливають на процес, відсутні, їх називають прихованими змінними (hidden variables) або латентним змінними (latent variables). Існують алгоритми евристичного пошуку [12] які намагаються враховувати такі скриті змінні при моделюванні.

Для випадку, коли навчальні дані неповні, або частина з них невірна (missing data) було запропоновано декілька алгоритмів стиснення границь (bound and collapse) та група алгоритмів, які використовують значення максимального

математичного очікування (expectation maximization, скорочено EM).

Метод стиснення границь моделює відсутність даних, припускаючи що ймовірність даних, які відсутні, приймає значення в інтервалі від 0 до 1. Тобто виконується обчислення цього інтервалу на відсутність даних, за тією інформацією яка мається. Після цього робиться стиснення границь інтервалу в точку за допомогою використання опуклої комбінації з точок екстремумів, використовуючи інформацію про неповні дані.

Алгоритм максимізації математичного очікування був запропонований в 1977 році в роботі [13 Алгоритм намагається знайти локальні оптимальні оцінки максимальної правдоподібності параметрів. Головна ідея алгоритму полягає в тому що, якби ми знали значення всіх вузлів, то навчання (на кроці M) було б простим, оскільки ми мали би всю необхідну інформацію. Тому на кроці E робиться обчислення значення очікуваної правдоподібності (expectation of the likelihood) включаючи латентні змінні, так ніби ми мали можливість їх спостерігати. На кроці M робиться обчислення значення максимальної правдоподібності (maximum likelihood estimates) параметрів використовуючи максимізацію значень очікуваної правдоподібності отриманих на кроці E. Далі алгоритм знову виконує крок E з використанням параметрів отриманих на кроці M і так далі.

На основі алгоритму максимізації математичного очікування розроблено цілу серію подібних алгоритмів. Так, наприклад, структурний алгоритм максимізації математичного очікування (structural EM algorithm) поєднує в собі стандартний алгоритм максимізації математичного очікування, що оптимізує параметри, та алгоритм структурного пошуку моделі відбору. Цей алгоритм будує мережі, ґрунтуючись на штрафних ймовірнісних значеннях, які включають значення, отримані за допомогою байєсівського інформаційного критерію, принципу мінімальної довжини опису, а також значеннях інших критеріїв.

2.1.2 Нелінійна поліноміальна складність задачі побудови байєсівської мережі

Побудову БМ можна виконати "у чоло", простим перебором (exhaustive

search) множини всіх можливих нециклічних моделей, з яких обирається модель найбільш адекватна навчальним даним. Але тоді ця задача набуває поліноміальної складності, тому що при повному переборі кількість всіх моделей дорівнює $3^{\frac{n\cdot(n-1)}{2}} - k_{cycle}$, де n – кількість вершин, k_{cycle} – кількість моделей з циклами. Кількість всіх можливих нециклічних моделей можна порахувати за допомогою рекурентної формули Робінсона, запропонованої у 1976 році [14] (таблиця 2.1):

$$f(n) = \sum_{i=1}^{n} (-1)^{i+1} \cdot C_n^i \cdot 2^{i\cdot(n-i)} \cdot f(n-i)$$

де n – кількість вершин, а $f(0) = 1.$

Таблиця 2.1 – Таблиця залежності кількості моделей без циклів від кількості вершин, які потрібно проаналізувати при повному переборі моделей

Кількість вершин	Кількість моделей без циклів		Кількість вершин	Кількість моделей без циклів
1	1		8	783.702.329.343
2	3		9	1.213.442.454.842.881
3	25		10	4.175.098.976.430.598.100
4	543		…	
5	29.281		15	2,38*10^41
6	3.781.503		…	…
7	1.138.779.265		20	2,34*10^72

Однак на практиці виконати повний перебір моделей можна тільки для мереж не більш ніж з 7 вершинами. При кількості вершин більше 7 виконати простий перебір неможливо, тому що не вистачить ніяких обчислювальних ресурсів.

2.1.3 Методи зменшення розрахункової складності при побудові байєсівських мереж

2.1.3.1 Методи з використанням впорядкованої множини вершин

Купер і Герскович, Дехтер та багато інших дослідників для зменшення

простору структур мережі пропонують вважати, що множина змінних (вершин), описуючих процес, впорядкована. Тобто задається ВМВ вигляду $\{X^{(1)} < X^{(2)} < ... < X^{(n)}\}$, де вершина $X^{(1)}$ – це головна коренева вершина в якої немає предків, а запис $X^{(i)} < X^{(j)}$ означає, що вершина $X^{(i)}$ передує вершині $X^{(j)}$, тобто вершина $X^{(j)}$ не може бути предком вершини $X^{(i)}$, дуги можуть іти тільки з $X^{(i)}$ в $X^{(j)}$.

Використовуючи попередню впорядкованість вершин, кількість структур необхідних для розгляду зменшується до $2^{\frac{n(n-1)}{2}}$, де n – кількість вершин. Це дуже важливе припущення, що дозволяє заощаджувати обчислювальні потужності, при $n=10$ кількість структур зменшується в 10^7 разів. Але при цьому потрібно застосовувати експертів, які повинні докладно знати та розуміти предметну область і будуть виконувати попередню упорядкованість вершин. Також можна впорядковувати вершини на основі аналізу характеристик навчальних даних, наприклад можна використати кореляцію атрибутів вершин або табу пошук (taboo search). Але навіть при введенні попередньої упорядкованості вершин обсяг обчислень залишається величезним, наприклад при $n=8$ вершин, буде потрібно виконати аналіз $2^{\frac{8*(8-1)}{2}} = 268.435.456$ структур.

В статті [63] пропонується метод, відомий у літературі як К2. До вершини $X^{(N)}$ по черзі, послідовно перебираючи ВМВ, додають предків від $X^{(1)}$ до $X^{(N-1)}$ та обчислюють значення функції Купера-Герсковича (КГ) для кожної побудованої таким чином мережі. В якості батьківської вершини для дитячої вершини $X^{(N)}$ залишають вершину $X^{(i)}$ при якій функція КГ приймає максимальне значення. Після цього.до вершин $X^{(N)}$ і $X^{(N-1)}$ по черзі додають батьків від $X^{(1)}$ до $X^{(N-2)}$, обчислюючи значення функції КГ відповідних мереж. В результаті в якості рішення метод видає таку структуру мережі, для якої функція КГ приймає максимальне значення.

Припущення про наявність ВМВ суттєво скорочує простір всіх можливих ациклічних структур. Але з'являється нова нетривіальна проблема – як маючи

множину навчальних даних отримати ВМВ мережі. Самий очевидний спосіб – це залучення експертів. Однак не завжди можлива наявність кваліфікованих експертів, тому що може виникнути потреба моделювання даних у такій предметній області, де досвідчених експертів просто немає.

2.1.3.2 Інші евристичні методи.

Під евристичними розуміють методи які дозволяють виконати побудову БМ без здійснення повного перебору простору всіх можливих структур мережі. Серед евристичних методів нерідко зустрічаються методи із використанням ВМВ.

Замість ВМВ можна застосовувати інші характеристики навчальних даних, наприклад кореляцію між змінними мережі. Але як показують результати обчислювальних експериментів методи із використанням кореляції не ефективні. В якості ТУН можна використовувати χ^2-критерій Пірсона або χ^2- тест відношення правдоподібності, але найбільш ефективними виявилися методи із використанням значення взаємної інформації (mutual information).

В таблиці 2.2 представлені значення частоти взаємного сполучення змінних $X^{(1)}$ та $X^{(2)}$.

Таблиця 2.2 – Таблиця значень частоти взаємного сполучення змінних $X^{(1)}$ та $X^{(2)}$

Змінні	$X^{(1)} = x_1^{(1)}$	$X^{(1)} = x_2^{(1)}$	*Сума*
$X^{(2)} = x_1^{(2)}$	a	b	$a+b$
$X^{(2)} = x_2^{(2)}$	c	d	$c+d$
Сума	$a+c$	$b+d$	n

Статистичні показники зв'язку змінних. В якості ТУН для виміру тісноти залежності застосовують статистичні показники. Наприклад коефіцієнти асоціації, контингенції (contingency) або взаємної спряженості [15].

В таблиці 2.2 a,b,c,d – частоти взаємного сполучення двох змінних, а n – сума частот.

Коефіцієнт асоціації Юла змінних $X^{(1)}$ та $X^{(2)}$ з таблиці 2.2 обчислюється за формулою

$$K_{ac} = \frac{a \cdot d - b \cdot c}{a \cdot d + b \cdot c}.$$

Коефіцієнт контингенції (спряженості) Бравайса (він же – показник подоби Чупрова) між змінними $X^{(1)}$ та $X^{(2)}$ в таблиці 2.2 розраховується за формулою:

$$K_{kon} = \frac{a \cdot d - b \cdot c}{\sqrt{(a+b) \cdot (b+d) \cdot (a+c) \cdot (c+d)}}.$$

Коефіцієнти асоціації та контингенції приймають значення від мінус одиниці до одиниці, треба зауважити, що для одних і тих самих даних коефіцієнт контингенції завжди менше коефіцієнта асоціації. Одиниця означає повний позитивний зв'язок, від'ємна одиниця означає повний негативний зв'язок, а нуль означає незалежність змінних.

У випадку коли змінні мережі мають більше ніж два стани, як наприклад змінні $X^{(1)}$ та $X^{(2)}$ (таблиця 2.3), застосовуються коефіцієнт взаємної спряженості.

Таблиця 2.3 – Таблиця значень частоти взаємного сполучення змінних $X^{(1)}$ та $X^{(2)}$

Змінні	$X^{(1)} = x_1^{(1)}$	…	$X^{(1)} = x_n^{(1)}$	Сума
$X^{(2)} = x_1^{(2)}$	k_{11}	…	k_{n1}	$\sum_{j=1}^{n} k_{j1}$
…	…	…	…	…
$X^{(2)} = x_m^{(2)}$	k_{1m}	…	k_{nm}	$\sum_{j=1}^{n} k_{jm}$
Сума	$\sum_{i=1}^{m} k_{1i}$	…	$\sum_{i=1}^{m} k_{ni}$	$\sum_{i=1}^{m}\sum_{j=1}^{n} k_{ji}$

Коефіцієнт взаємної спряженості Пірсона розраховується за формулою

$K = \sqrt{\frac{\varphi^2}{1+\varphi^2}}$, де φ^2 – показник середньо квадратичної спряженості, який розраховується за формулою:

$$\varphi^2 = \left(\sum_{j=1}^{n} \left(\sum_{i=1}^{m} \left(\frac{k_{ji}}{\left(\sum_{j=1}^{n} k_{ji} \right) \cdot \left(\sum_{i=1}^{m} k_{ji} \right)} \right) \right) \right).$$

Коефіцієнт взаємної спряженості Пірсона приймає значення від нуля до одиниці. Нуль говорить про те що змінні незалежні, а одиниця означає що значення однієї змінної можна точно спрогнозувати по іншій.

Значення взаємної інформації. В роботі Шоу і Лью в 1968 році, для оцінки ступеня залежності двох довільних змінних x^i та x^j при побудові БМ, вперше було запропоновано використання значення взаємної інформації (ЗВІ) $MI(x^i, x^j)$. Для розрахунку запропонована формула:

$$MI(x^i, x^j) = \sum_{x^i, x^j} P(x^i, x^j) \cdot \log\left(\frac{P(x^i, x^j)}{P(x^i) \cdot P(x^j)} \right). \quad (2.1)$$

За своєю суттю ЗВІ являється аналогом кореляції, але за своїм змістом – це оцінка кількості інформації яка присутня в змінній x^i про змінну x^j. ЗВІ приймає невід'ємні значення, тобто $MI(x^i, x^j) \geq 0$, а у випадку якщо вершини x^i та x^j повністю незалежні одна від одної, то $MI(x^i, x^j) = 0$, тому що $P(x^i, x^j) = P(x^i) \cdot P(x^j)$, так як:

$$\log\left(\frac{P(x^i, x^j)}{P(x^i) \cdot P(x^j)} \right) = \log\left(\frac{P(x^i) \cdot P(x^j)}{P(x^i) \cdot P(x^j)} \right) = \log(1) = 0$$

У випадку, коли БМ складається з N вершини, для обчислення $MI(x^i, x^j)$ всіх пар x^i та x^j потрібно виконати $\frac{N \cdot (N-1)}{2}$ обчислення, при цьому треба враховувати що $MI(x^i, x^j) = MI(x^j, x^i)$.

В якості прикладу обчислимо ЗВІ між змінними X^1 та X^2 для яких задана множина навчальних даних з 10 записів у таблиці 2.4, яке обчислюється із використанням проміжних значень ймовірностей, наведених у таблицях 2.5–2.7, значення цих таблиць обчислені на основі даних з таблиці 2.4.

Таблиця 2.4 – Множина навчальних даних з 10 записів

n	X^1	X^2		n	X^1	X^2
1	0	1		6	0	1
2	1	0		7	1	0
3	0	1		8	1	0
4	1	0		9	0	1
5	0	1		10	1	1

$$MI(X^{(1)}, X^{(2)}) = \sum_{x^{(1)}, x^{(2)}} P\left(X^{(i)}, X^{(j)}\right) \cdot \log\left(\frac{P(X^{(i)}, X^{(j)})}{P(X^{(i)}) \cdot P(X^{(j)})}\right) =$$

$$= P\left(X^{(1)} = 0, X^{(2)} = 0\right) \cdot \log\left(\frac{P(X^{(1)} = 0, X^{(2)} = 0)}{P(X^{(1)} = 0) \cdot P(X^{(2)} = 0)}\right) +$$

$$+ P\left(X^{(1)} = 0, X^{(2)} = 1\right) \cdot \log\left(\frac{P(X^{(1)} = 0, X^{(2)} = 1)}{P(X^{(1)} = 0) \cdot P(X^{(2)} = 1)}\right) +$$

$$+ P\left(X^{(1)} = 1, X^{(2)} = 0\right) \cdot \log\left(\frac{P(X^{(1)} = 1, X^{(2)} = 0)}{P(X^{(1)} = 1) \cdot P(X^{(2)} = 0)}\right) +$$

$$+ P\left(X^{(1)} = 1, X^{(2)} = 1\right) \cdot \log\left(\frac{P(X^{(1)} = 1, X^{(2)} = 1)}{P(X^{(1)} = 1) \cdot P(X^{(2)} = 1)}\right) =$$

$$= 0 + 0{,}5 \cdot \log\left(\frac{0{,}5}{0{,}5 \cdot 0{,}6}\right) + 0{,}4 \cdot \log\left(\frac{0{,}4}{0{,}5 \cdot 0{,}4}\right) + 0{,}1 \cdot \log\left(\frac{0{,}1}{0{,}5 \cdot 0{,}6}\right) =$$

$$= 0 + 0{,}2554 + 0{,}2773 - 0{,}1099 = 0{,}4228 \ .$$

Таблиця 2.5 – Таблиця значень ймовірності змінної X^1

$P(X^1)$	
$P(X^1 = 0)$	$P(X^1 = 1)$
0,5	0,5

Таблиця 2.6 – Таблиця значень ймовірності змінної X^2

$P(X^2)$	
$P(X^2=0)$	$P(X^2=1)$
0,4	0,6

Ідея використання ЗВІ для побудови БМ пізніше була застосована в роботі [16]. Так, евристичний метод спочатку робить обчислення ЗВІ між всіма вершинами, після чого виконується цілеспрямований пошук, який використовує принцип ОМД в якості оціночної функції, що застосовується на кожній ітерації алгоритму побудови БМ.

Таблиця 2.7 – Таблиця значень спільної ймовірності змінних X^1 та X^2

$P(X^1, X^2)$	
$P(X^1=0, X^2=0)$	0
$P(X^1=0, X^2=1)$	0,5
$P(X^1=1, X^2=0)$	0,4
$P(X^1=1, X^2=1)$	0,1

2.1.4 Функція Купера-Герсковича

У роботі Купер і Герскович запропонували метод із використанням функції Купера-Герсковича (КГ) для навчання БМ, що заснований на пошуку структури БМ із максимальним значенням функції КГ. Функція КГ структури $g \in G$ при заданій послідовності зі n спостережень $x^n = d_1 d_2 ... d_n$ записується як рівняння:

$$P(g, x^n) = P(g) \cdot \prod_{j \in J} \left(\prod_{s \in S(j,g)} \frac{(\alpha^{(j)} - 1)! \prod_{q \in A^{(j)}} n[q, s, j, g]!}{(n[s, j, g] + \alpha^{(j)} - 1)!} \right),$$

де $P(g)$ – апріорна ймовірність структури $g \in G$ (її часто опускають при обчисленнях, вважаючи що всі структури рівноймовірні);

$j \in J = \{1, ..., N\}$ – перебір всіх вершин структури мережі g;

$s \in S(j,g)$ – перебір множини всіх наборів значень які приймають батьківські вершини j-ї вершини

$$n(s,j,g) = \sum_{i=1}^{n} I(\pi_i^{(j)} = s); \qquad n[q,s,j,g] = \sum_{i=1}^{n} I(x_i = q, \pi_i^{(j)} = s),$$

де $\pi^{(j)} = \Pi^{(j)}$ означає $X^{(k)} = x^{(k)}, \forall k \in \phi^{(j)}$, а функція $I(E) = 1$, коли предикат $E = true$, в противному випадку $I(E) = 0$.

Алгоритм навчання БМ із використанням функції КГ заснований на циклічному переборі всіх можливих ациклічних мережевих структур. В g^* зберігається оптимальна мережна структура, тобто структура яка максимально відповідає навчальним даним. Оптимальною структурою буде та, котра має найбільше значення функції $P(g, x^n)$.

Алгоритм побудови БМ із використанням метода КГ:

1 – $g^* \leftarrow g_0 (\in G)$;

2 – для $\forall g \in G - \{g_0\}$ якщо $P(g, x^n) > P(g^*, x^n)$ то $g^* \leftarrow g$;

3 – в якості рішення на вихід подається g^*.

В якості прикладу обчислимо значення функції КГ для структури зображеної на рисунку 2.1, на основі 10 навчальних даних наведених у таблиці 2.8.

$$P(1,g,x^n) = \frac{(2-1)!\cdot 5!\cdot 5!}{(10+2-1)!} = 0.00036;$$

$$P(2,g,x^n) = \frac{(2-1)!\cdot 0!\cdot 5!}{(5+2-1)!} \cdot \frac{(2-1)!\cdot 4!\cdot 1!}{(5+2-1)!} = 0.0056;$$

$$P(3,g,x^n) = \frac{(2-1)!\cdot 3!\cdot 1!}{(4+2-1)!} \cdot \frac{(2-1)!\cdot 0!\cdot 6!}{(6+2-1)!} = 0.0071;$$

$$P(g,x^n) = \frac{1}{25} \cdot \prod_{j \in J} P(j,g,x^n) = 5.7254 \cdot 10^{-10}.$$

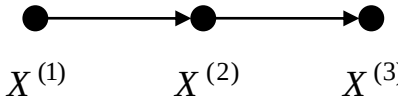


Рисунок 2.1 – Оптимальна структура БМ,
що відповідає даним з табл. 2.8

Таблиця 2.8 – Множина навчальних даних з 10 записів

n	$X^{(1)}$	$X^{(2)}$	$X^{(3)}$		n	$X^{(1)}$	$X^{(2)}$	$X^{(3)}$
1	0	1	1		6	0	1	1
2	1	0	0		7	1	0	1
3	0	1	1		8	1	0	0
4	1	0	0		9	0	1	1
5	0	1	1		10	1	1	1

Однак при використанні функції КГ необхідно враховувати обчислювальні обмеження моделюючих систем, пов'язані з кінцевою довжиною розрядної сітки. Приведемо тривіальний приклад, коли в структурі є дві вершини $X^{(1)}$ і $X^{(2)}$, а множина навчальних даних складається з мільйона записів $D=\{d^{(1)},...,d^{(1.000.000)}\}$. У такому випадку при обчисленні $P(g,x^n)$ буде потрібно обчислити факторіал виду $(n[s,j,g]+\alpha^{(j)}-1)!=(1.000.000+\alpha^{(j)}-1)!$, у той час як такі солідні 32-х розрядні програми як MatLab й MathCAD здатні обчислити факторіали не більше 170!

2.1.5 Модифікована логарифмічна функція Купера-Герсковича

Для ширшого застосування функції КГ потрібно позбутися від факторіала. Для цього візьмемо логарифм рівняння за допомогою якого обчислюється функція КГ:

$$\log P(g,x^n) = \log\left(P(g)\cdot\prod_{j\in J}\left(\prod_{s\in S(j,g)}\frac{(\alpha^{(j)}-1)!\prod_{q\in A^{(j)}}n[q,s,j,g]!}{(n[s,j,g]+\alpha^{(j)}-1)!}\right)\right)=$$

$$=\log P(g)+\sum_{j\in J}\left(\sum_{s\in S(j,g)}\left(\sum_{i=1}^{\alpha^{(j)}-1}i+\sum_{q\in A^{(j)}}\left(\sum_{i=1}^{n[q,s,j,g]}i\right)-\sum_{i=1}^{n[s,j,g]+\alpha^{(j)}-1}i\right)\right)=$$

$$=\log P(g)+\sum_{j\in J}\left(\sum_{s\in S(j,g)}\left(\sum_{q\in A^{(j)}}\left(\sum_{i=1}^{n[q,s,j,g]}i\right)-\sum_{i=\alpha^{(j)}}^{n[s,j,g]+\alpha^{(j)}-1}i\right)\right).$$

Отриманий вираз помножимо на мінус одиницю та для економії обчислювальних ресурсів заберемо з нього $\log P(g)$, вважаємо, що апріорні ймовірності $P(g)$ всіх структур рівні. Тепер замість пошуку структури з максимальним значенням функції КГ потрібно буде здійснювати пошук структури з мінімальним значенням модифікація функції Купера-Герсковича (МФКГ):

$$F(g,x^n)=\sum_{j\in J}\left(\sum_{s\in S(j,g)}\left(\sum_{i=\alpha^{(j)}}^{n[s,j,g]+\alpha^{(j)}-1}i\right)\right)-\sum_{j\in J}\left(\sum_{s\in S(j,g)}\left(\sum_{q\in A^{(j)}}\left(\sum_{i=1}^{n[q,s,j,g]}i\right)\right)\right).$$

В ході обчислювальних експериментів виявлено, що функції КГ і МФКГ видають на одних і тих самих навчальних даних абсолютно однакові рішення. Однак на маленьких мережах до 10 вершин функція КГ працює швидше МФКГ, а на мережах з більшою кількістю вершин ситуація протилежна.

2.1.6 Функція опису мінімальною довжиною

Згідно з теорією кодування Шеннона, при відомому розподілі $P(X)$ випадкової величини X, довжина оптимального коду для передачі конкретного значення x по каналу зв'язку прямує до $L(x)=-\log P(x)$. Ентропія джерела $S(P)=-\sum_x P(x)\cdot\log P(x)$ є мінімально очікуваною довжиною закодованого повідомлення. Будь-який інший код, заснований на невірному уявленні про джерело повідомлень, приведе до більшої довжини повідомлення. Іншими словами, чим краще побудована модель джерела, тим компактніше кодуються дані.

При навчанні джерелом даних виступає деяка невідома істинна функція

розподілу $P(D|h_0)$, де $D=\{d_1,...,d_N\}$ – набір даних, а h – гіпотеза про ймовірнісне походження даних, $L(D|h)=-\log P(D|h)$ – емпіричний ризик адитивний по кількості спостережень і пропорційний емпіричній помилці. Відмінність між $P(D|h_0)$ і розподілом моделі $P(D|h)$ за мірою Кулбака-Леблера визначається як:

$$|P(D|h)-P(D|h_0)|=\sum_D P(D|h_0)\cdot\log\frac{P(D|h_0)}{P(D|h)}= \quad (2.2)$$
$$=\sum_D P(D|h_0)\cdot|L(D|h)-L(D|h_0)|\geq 0 \ ,$$

тобто, вона представляє собою різницю очікуваної довжини кодування даних на основі гіпотези про мінімально можливу довжину. Ця різниця завжди позитивна додатна і дорівнює нулю лише при повному збігу двох розподілів. Іншими словами, гіпотеза краща у тому випадку, коли середня довжина кодування даних як можна менша [17]. Принцип ОМД у своїй спрощеній і найбільш загальній формі стверджує: серед безлічі моделей варто обрати ту, котра дозволяє описати дані найбільш коротко без втрати інформації [18].

У загальному випадку задача ОМД виглядає наступним чином. Спочатку задається множина навчальних даних $D=\{d_1,...,d_n\}$, $d_i=\{x_i^{(1)}x_i^{(2)}...x_i^{(N)}\}$ (нижній індекс – номер спостереження, а верхній – номер змінної), n – кількість спостережень, кожне спостереження складається з N $(N\geq 2)$ змінних $X^{(1)},X^{(2)},...,X^{(N)}$, кожна j-та змінна $(j=1,...,N)$ має $A^{(j)}=\{0,1,...,\alpha^{(j)}-1\}$ $(\alpha^{(j)}\geq 2)$ станів, кожна структура БМ представляється N множинами предків $(\Pi^{(1)},...,\Pi^{(N)})$, тобто для кожної вершини $j=1,...,N$, $\Pi^{(j)}$ – це множина батьківських вершин, така що $\Pi^{(j)}\subseteq\{X^{(1)},...,X^{(N)}\}\setminus\{X^{(j)}\}$ (вершина не може бути батьком самої себе , тобто петлі в графі відсутні). Тоді ОМД структури $g\in G$ при заданій послідовності з n спостережень $x^n=d_1d_2...d_n$ обчислюється за формулою:

$$L(g,x^n)=H(g,x^n)+\frac{k(g)}{2}\cdot\log(n), \tag{2.3}$$

де $k(g)$ – кількість незалежних умовних ймовірностей у структурі БМ g;

$H(g,x^n)$ – це емпірична ентропія.

$$H(g,x^n)=\sum_{j\in J}H(j,g,x^n), \tag{2.4}$$

$$k(g)=\sum_{j\in J}k(j,g), \tag{2.5}$$

де ОМД j-ї вершини обчислюється за формулою:

$$L(j,g,x^n)=H(j,g,x^n)+\frac{k(j,g)}{2}\cdot\log(n), \tag{2.6}$$

де $k(j,g)$ – кількість незалежних умовних імовірностей j-ї вершини:

$$k(j,g)=(\alpha^{(j)}-1)\cdot\prod_{k\in\phi(j)}\alpha^k, \tag{2.7}$$

де $\phi(j)\subseteq\{1,..,j-1,j+1,...,N\}$, – це така множина для якої $\Pi^{(j)}=\{X^{(k)}:k\in\phi^{(j)}\}$.

Емпірична ентропія j-ї вершини обчислюється за формулою:

$$H(j,g,x^n)=\sum_{s\in S(j,g)}\sum_{q\in A^{(j)}}-n[q,s,j,g]\cdot\log\frac{n[q,s,j,g]}{n[s,j,g]}, \tag{2.8}$$

де

$$n(s,j,g)=\sum_{i=1}^{n}I(\pi_i^{(j)}=s), \tag{2.9}$$

$$n[q,s,j,g]=\sum_{i=1}^{n}I(x_i=q,\pi_i^{(j)}=s), \tag{2.10}$$

де $\pi^{(j)}=\Pi^{(j)}$, а $X^{(k)}=x^{(k)},\forall k\in\phi^{(j)}$, функція $I(E)=1$, якщо предикат $E=true$, інакше $I(E)=0$.

Простий алгоритм навчання БМ із використанням ОМД виглядає наступним чином: циклічно робиться перебір усіх можливих не циклічних структур мереж. В g^* зберігається оптимальна структура мережі. Оптимальною структурою буде та,

у якої буде найменше значення функції $L(g, x^n)$.

Алгоритм побудови БМ із використанням ОМД:

1. $g^* \leftarrow g_0 (\in G)$;

2. для $\forall g \in G - \{g_0\}$ якщо $L(g, x^n) < L(g^*, x^n)$ то $g^* \leftarrow g$;

3. в якості рішення на вихід подається g^*.

Приклад застосування функції опису мінімальною довжиною

Нехай задано набір навчальних даних з 10 спостережень для навчання БМ, що наведені у таблиці 2.8. У випадку повного перебору всіх можливих структур мережі потрібно буде розглянути 25 структур. Після того як всі структури будуть розглянуті, у якості оптимальної буде видана структура, зображена на рисунку 2.1.

Обчислення значення ОМД цієї структури робиться наступним чином. У вершини $X^{(1)}$ відсутні батьки, тобто $\Pi^{(1)} = \{\}$. Емпірична ентропія обчислюється як $H(j=1, g) = -5 \cdot \log\left(\frac{5}{10}\right) - 5 \cdot \log\left(\frac{5}{10}\right) = 6{,}9315$, а кількість незалежних умовних ймовірностей $k(j=1, g) = 2 - 1 = 1$. Отже, довжина опису вершини $X^{(1)}$ дорівнює $L(1, g) = 6{,}9315 + \frac{1}{2} \cdot \log(10) = 8{,}0828$. При обчисленні можна застосовувати логарифм із будь-якою базою, в даному прикладі використається з базою $e = 2{,}7183$, тобто , натуральний логарифм.

Значення параметрів вершини $X^{(1)}$ представленні в таблиці 2.9.

Таблиця 2.9 – Таблиця значень параметрів вершини $X^{(1)}$

$X^{(1)}$	$n[q, s, j, g]$	$n[s, j, g]$
0	5	10
1	5	

Вершина $X^{(2)}$ має одну батьківську вершину $X^{(1)}$, тобто $\Pi^{(2)} = \{X^{(1)}\}$. Емпірична ентропія обчислюється як:

$$H(j=2,g)=\left(-0\cdot\log\left(\frac{0}{5}\right)-5\cdot\log\left(\frac{5}{5}\right)\right)+\left(-4\cdot\log\left(\frac{4}{5}\right)-1\cdot\log\left(\frac{1}{5}\right)\right)=2{,}502,$$

а кількість незалежних умовних ймовірностей $k(j=2,g)=(2-1)\cdot 2=2$. Отже довжина опису вершини дорівнює: $L(2,g)=2{,}502+\frac{2}{2}\cdot\log(10)=4{,}8046.$

Значення параметрів вершини $X^{(2)}$ і $X^{(3)}$ представленні в таблиці 2.10.

Таблиця 2.10 – Таблиця значень параметрів вершин $X^{(2)}$ та $X^{(3)}$

$X^{(1)}$	$X^{(2)}$	$n[q,s,j,g]$	$n[s,j,g]$		$X^{(2)}$	$X^{(3)}$	$n[q,s,j,g]$	$n[s,j,g]$
0	0	0	5		0	0	3	4
0	1	5			0	1	1	
1	0	4	5		1	0	0	6
1	1	1			1	1	6	

Вершина $X^{(3)}$ має одну батьківську вершину $X^{(2)}$, тобто $\Pi^{(3)}=\{X^{(2)}\}$. Емпірична ентропія обчислюється як:

$$H(j=3,g)=\left(-3\cdot\log\left(\frac{3}{4}\right)-1\cdot\log\left(\frac{1}{4}\right)\right)+\left(-0\cdot\log\left(\frac{0}{6}\right)-6\cdot\log\left(\frac{6}{6}\right)\right)=2.2493,$$

а кількість незалежних умовних ймовірностей $k(j=3,g)=(2-1)\cdot 2=2$. Отже довжина опису вершини $X^{(3)}$ дорівнює

$$H(j=3,g)=\left(-3\cdot\log\left(\frac{3}{4}\right)-1\cdot\log\left(\frac{1}{4}\right)\right)+\left(-0\cdot\log\left(\frac{0}{6}\right)-6\cdot\log\left(\frac{6}{6}\right)\right)=2.2493$$

Тобто довжина опису структури представленої на рисунку 2.1 дорівнює:

$$H(g,x^n)=\sum_{j=1}^{3}H(j,g,x^n)=17{,}4393.$$

2.1.7 Порівняння ефективності оціночних функцій

Для побудови структури БМ найчастіше застосовують методи із використанням функцій КГ або ОМД, а також різні модифікації цих функцій, наприклад МФКГ. Результати обчислювальних експериментів показали, що на коротких навчальних вибірках, до 170 записів, і мережах, що складаються не більш

ніж з 10 вершин, функція КГ працює швидше в порівнянні з функцією МФКГ й ОМД. Але функції МФКГ й ОМД, на відміну від КГ, працюють із навчальними вибірками будь-якого розміру. Також з'ясовано, що методи побудови БМ із використанням функцій КГ та її модифікацій, виконують перенавчання БМ, тобто часто такі мережі містять зайві дуги.

На рисунках 2.2 та 2.3 показано графіки витраченого часу на побудову БМ методами, що використовують МФКГ та ОМД. Для задавання порядку додавання дуг у БМ ці методи використовують ЗВІ [16]. При побудові БМ в якості вибірки навчальних даних використано генетичні дані, які складаються з 600 навчальних записів.

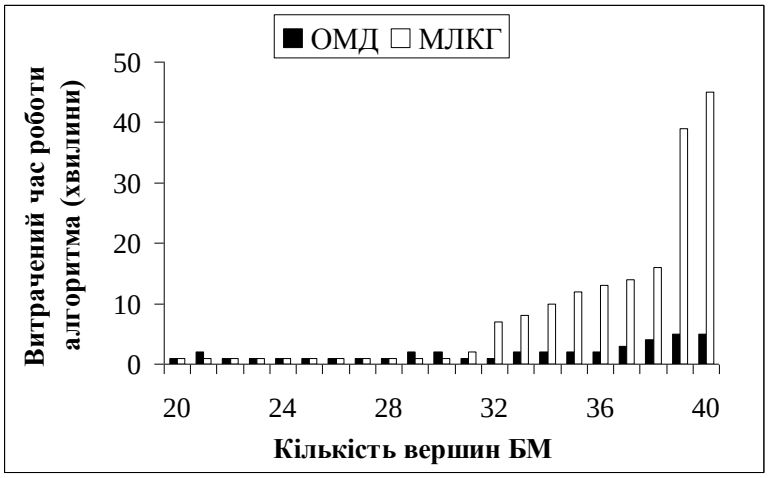


Рисунок 2.2 – Діаграма витрат часу на алгоритми з використанням функцій МФКГ та ОМД

Рисунок 2.3 – Графік витрат часу при використанні алгоритму із застосуванням функції ОМД

Як можна побачити з рисунків 2.2 та 2.3, для побудови БМ, що складаються з більш ніж з 30 вершин, алгоритм із використанням ОМД працює швидше у порівнянні із МФКГ.

2.2 Еврістичний алгоритм побудови байєсівських мереж по навчальним даним

Вхідні дані. Множина навчальних даних $D=\{d_1,...,d_n\}$, $d_i=\{x_i^{(1)}x_i^{(2)}...x_i^{(N)}\}$ (нижній індекс – номер спостереження, а верхній – номер змінної), n – кількість спостережень; N – кількість вершин (змінних).

Перший етап. Для всіх пар вершин обчислюються значення взаємної інформації $Set\_MI=\left\{MI(x^i,x^j);\ \forall i,j\right\}$. Після цього елементи множини Set_MI впорядковують за зменшенням

$$Set\_MI=\{MI(x^{m_1},x^{m_2}),MI(x^{m_3},x^{m_4}),MI(x^{m_5},x^{m_6}),...\}.$$

Другий етап

Крок 1. З множини значень взаємної інформації Set_MI обирають перші два максимальних значення $MI(x^{m_1},x^{m_2})$ та $MI(x^{m_3},x^{m_4})$. За цими значеннями $MI(x^{m_1},x^{m_2})$ та $MI(x^{m_3},x^{m_4})$ будується множина моделей G:

{ $(m_1\to m_2;\ m_3\to m_4)$, $(m_1\to m_2;m_3\leftarrow m_4)$, $(m_1\leftarrow m_2;m_3\leftarrow m_4)$, $(m_1\leftarrow m_2;m_3\to m_4)$, $(m_1\leftarrow m_2;$ m_3 не залежить від $m_4)$, $(m_1\to m_2;$ m_3 не залежить від $m_4)$, (m_1 не залежить від $m_2;$ $m_3\to m_4)$, (m_1 не залежить від $m_2;$ $m_3\leftarrow m_4)$, (m_1 не залежить від $m_2;$ m_3 не залежить від $m_4)$}.

Запис виду $m_i\to m_j$ означає, що вершина x^{m_i} є предком вершини x^{m_j}.

Крок 2. Здійснюється пошук, серед всіх моделей множини G. В параметрі g^* зберігається оптимальна структура БМ. Оптимальною структурою буде та, у якої буде найменше значення функції $L(g,x^n)$. Де $L(g,x^n)$ – ОМД структури

моделі при заданій послідовності з n спостережень $x^n = d_1 d_2 ... d_n$.

1. $g^* \leftarrow g_0 (\in G)$;

2. для $\forall g \in G - \{g_0\}$ якщо $L(g, x^n) < L(g^*, x^n)$ то $g^* \leftarrow g$;

3. на вихід в якості рішення подається g^*.

Крок 3. Після того як знайдена оптимальна структура (структури) g^* з G, з множини значень взаємної інформації Set_MI обирають наступне максимальне значення $MI(x^{i\_next_i}, x^{j\_next})$. За отриманим значенням $MI(x^{i\_next_i}, x^{j\_next})$ і структурою (структурам) g^* будується множина моделей G у вигляді: { (g^*; $i\_next \rightarrow j\_next$), ($g^*$; $i\_next \leftarrow j\_next$), (g^*; i_next не залежить від j_next)}. Після цього виконується крок 2.

Умова завершення алгоритму. Евристичний метод буде виконуватися до тих пір, поки не буде проаналізовано певне число елементів множини або всі $\frac{N \cdot (N-1)}{2}$ елементи множини Set_MI. Як показує практика, у більшості випадків нема рації виконувати аналіз більш ніж половини (тобто $\frac{N \cdot (N-1)}{4}$) елементи множини Set_MI.

Вихідні дані алгоритму.

Оптимальна структура (структури) g^*.

В якості приклада розглянемо загальновідому мережу "Азія" яка складається з восьми вершин.

У таблиці 2.11 наведені ЗВІ між всіма вершинами мережі розташовані за зменшенням (перший етап алгоритму).

Побудова виконувалося вибіркою з 7000 навчальних спостережень. На рисунку 2.4 наведена структура оригінальної БМ, на основі якої генерувалися значення.

Таблиця 2.11 – ЗВІ між всіма вершинами БМ "Азія"

№	*MI*	*i*	*j*		№	*MI*	*i*	*j*
1	0,251	7	8		15	0,001227	3	5
2	0,136	2	4		16	0,000851	1	6
3	0,125	4	6		17	0,000508	2	7
4	0,096	2	6		18	0,000381	3	7
5	0,048	1	7		19	0,000266	4	5
6	0,036	3	4		20	0,000197	1	5
7	0,025	3	6		21	0,000128	4	7
8	0,0245	1	8		22	0,00012271	2	5
9	0,0132	4	8		23	0,00006475	5	6
10	0,0101	2	8		24	0,00003950	2	3
11	0,0051	6	8		25	0,00003249	5	7
12	0,0031	1	2		26	0,00001725	5	8
13	0,0028	3	8		27	0,00000303	1	3
14	0,0022	1	4		28	0,00000074	6	7

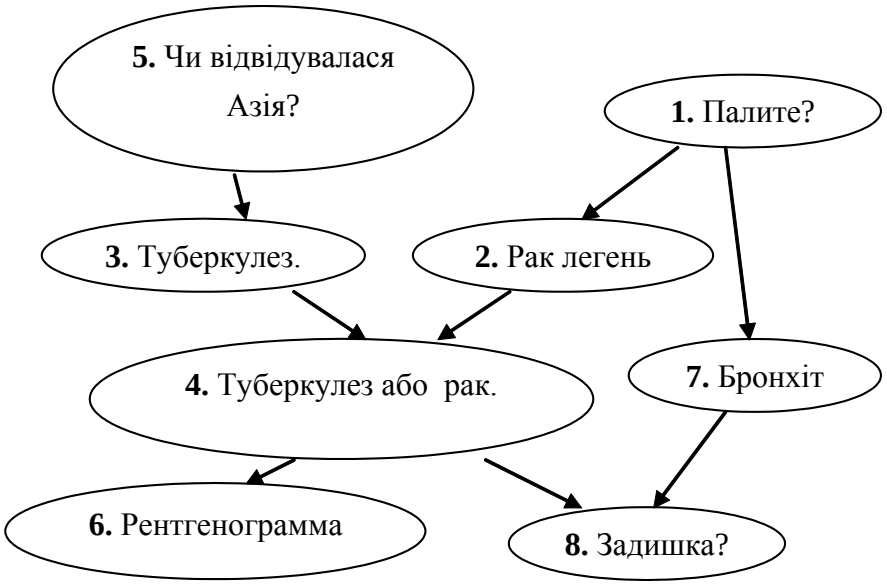


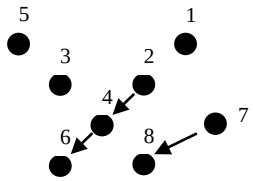
Рисунок 2.4 – Оригінальна мережа "Азія"

На 1-й ітерації по першим двом рядкам *MI(7,8)* та *MI(2,4)*, відсортованої матриці *MI*, будується множина моделей з 9 структур, після перебору цієї множини з'ясовується що мінімальне значення функції ОМД відповідає двом структурам рисунку 2.5.

а *L=17820,9586* б

Рисунок 2.5 – Структури БМ отримані на 1-й ітерації

На 2-й ітерації по отриманим на 1-й ітерації структурам (див. рисунок 2.5) і *MI(4,6)* будується множина моделей з 6 структур, з яких мінімальне значення функції ОМД приймає мережа зображена на рисунку 2.6.



L=16918,4958

Рисунок 2.6 – Структура БМ отримана на 2-й та 3-й ітераціях

На 3-й ітерації по отриманій на 2-й ітерації структурі (див. рисунок 2.6) та *MI(2,6)* будується множина моделей з 3 структур. З цих трьох структур мінімальне значення функції ОМД приймає та ж сама структура зображена на рисунку 2.6.

На 4-й ітерації по отриманій на 3-й ітерації структурі (див. рисунок 2.6) і *MI(1,7)* будується множина моделей з 3 структур, з яких мінімальне значення функції ОМД приймають мережі зображені на рисунку 2.7.

а *L=16570,5342* б

Рисунок 2.7 – Структури БМ отримані на 4-й ітерації

На 5-й ітерації по структурам отриманим на 4-й ітерації (див. рисунок 2.7) і *MI(3,4)* будується множина моделей з 6 структур, з яких мінімальне значення функції ОМД приймають мережі зображені на рисунку 2.8.

а *L=16312,9052* б

Рисунок 2.8 – Структури БМ отримані на 5-й, 6-й та 7-й ітераціях

На 6-й ітерації по структурам отриманим на 5-й ітерації і *MI(3,6)* будується множина моделей з 6 структур. В результаті одержані ті ж самі структури що і на 5-й ітерації (див. рисунок 2.8).

На 7-й ітерації по структурам отриманим на 6-й ітерації і *MI(1,8)* будується множина моделей з 6 структур. Результат збігається з результатом попередніх двох ітерацій (див. рисунок 2.8).

На 8-й ітерації по структурам отриманим на 7-й ітерації (див. рисунок 2.8) і *MI(4,8)* будується множина моделей з 6 структур, з яких мінімальне значення функції ОМД приймають мережі зображені на рисунку 2.9.

На 9-й ітерації по структурам отриманим на 8-й ітерації і *MI(2,8)* будується множина моделей з 6 структур. В результаті одержані ті ж самі структури, що і на 8-й ітерації (див. рисунок 2.9).

а *L*= 16153,4686 б

Рисунок 2.9 – Структури БМ отримані на 8-й, 9-й та 10-й ітераціях

На 10-й ітерації по структурам отриманим на 9-й ітерації і *MI(6,8)* будується множина моделей з 6 структур. Результат збігається з результатом попередніх двох ітерацій (див. рисунок 2.9).

На 11-й ітерації по структурам отриманим на 10-й ітерації (див. рисунок 2.9) і *MI(1,2)* будується множина моделей з 6 структур, з яких мінімальне значення функції ОМД приймають мережі зображені на рисунку 2.10.

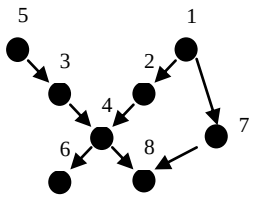
а *L=16143,5518* б

Рисунок 2.10 – Структури БМ отримані на 11-й, 12-й та 13-й ітераціях

На 12-й ітерації по структурам отриманим на 11-й ітерації і *MI(3,8)* будується множина моделей з 6 структур. В результаті одержані ті ж самі структури що і на 11-й ітерації (див. рисунок 2.10).

На 13-й ітерації по структурам отриманим на 12-й ітерації і *MI(1,4)* будується множина моделей з 6 структур. Результат збігається з результатом попередніх двох ітерацій (див. рисунок 2.10).

На 14-й ітерації по отриманим на 13-й ітерації структурам (див. рисунок 2.10) і *MI(3,5)* будується множина моделей з 6 структур, з яких мінімальне значення функції ОМД приймає мережа зображена на рисунку 2.11.



L=16139,1545

Рисунок 2.11 – Структура БМ отримана з 14 по 27 ітерації

З 15-й по 27-й ітерації ніяких змін в структурі не відбувається тобто результат збігається з результатом 14-й ітерації (див. рисунок 2.11).

Для побудови БМ "Азія" при простому аналізі всіх можливих нециклічних структур буде потрібно виконати оцінку 783 702 329 343 моделей. Тоді як евристичний метод на 27-ми ітераціях алгоритму виконує аналіз усього лише 120 структур, причому вже на 14-й ітерації, після аналізу 81 структури, метод видає структуру, що повністю збігається з оригінальною мережею "Азія". Тобто наступні 13 ітерацій методу не роблять ніяких змін, тому що оптимальна структура вже знайдена на 14 ітерації.

Проналізуємо експериментальні результати.

Виконано шість обчислювальних експериментів. У кожному експерименті евристичним методом проводилося навчання мережі з 10 вершин вибіркою з 2000 навчальних спостережень. Для оцінювання якості навчання використовується структурна різниця між побудованою і оригінальною БМ. У таблиці 2.12 наведені результати шести обчислювальних експериментів. Для кожного експерименту було виконано 44 ітерації навчання.

Таблиця 2.12 – Результати шести обчислювальних експериментів

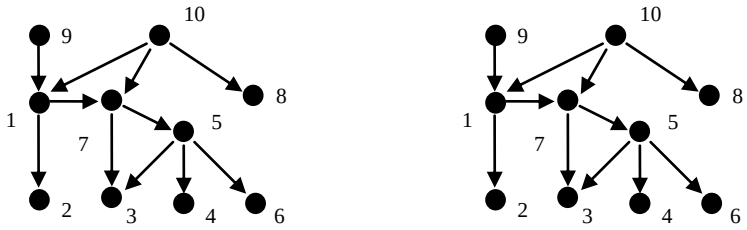
Номер обчислювального експерименту	№1	№2	№3	№4	№5	№6
Номер рисунка який відповідає обчислювальному експерименту	Рис. 3.12	Рис. 3.13	Рис. 3.14	Рис. 3.15	Рис. 3.16	Рис. 3.17
Загальна кількість моделей, проаналізованих евристичним методом на всіх ітераціях	513	178	415	282	550	329
Зайві дуги	1	0	1	2	4	0
Відсутні дуги	0	0	0	0	1	0
Реверсовані дуги	3	0	1	1	1	0
Структурна різниця між побудованою та оригінальною БМ	8	0	3	4	7	0

Як можна бачити з таблиці 2.12, у двох із шести обчислювальних

експериментах №2 та №6 (рисунках 2.13 та 2.17) побудована мережа повністю збіглася з оригінальною БМ. У двох із шести експериментах №3 та №4 (рисунки 2.14 та 2.15) помилка навчання, тобто структурна різниця між побудованою та оригінальною моделями дорівнює 3 та 4, для мережі з 10 вершин така помилка являється прийнятною. Значні помилки навчання отримані в експериментах №1 та №5 (рисунки 2.12 та 2.16). Однак для побудови мережі був виконаний аналіз усього лише 513 та 550 моделей відповідно, на всіх 44 ітерація, у той час як при простому переборі, тобто аналіз “в чоло”, всіх можливих нециклічних моделей потрібно було б проаналізувати 4 175 098 976 430 598 100 моделей.

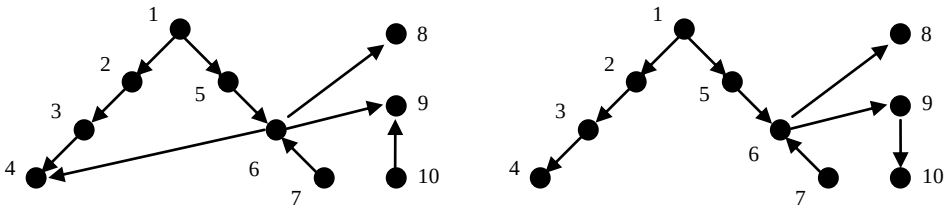
а б

Рисунок 2.12 – Експеримент №1 (а) побудована (б) оригінальна БМ



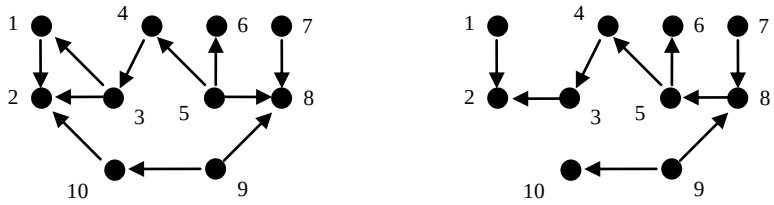
а б

Рисунок 2.13 – Експеримент №2 (а) побудована (б) оригінальна БМ



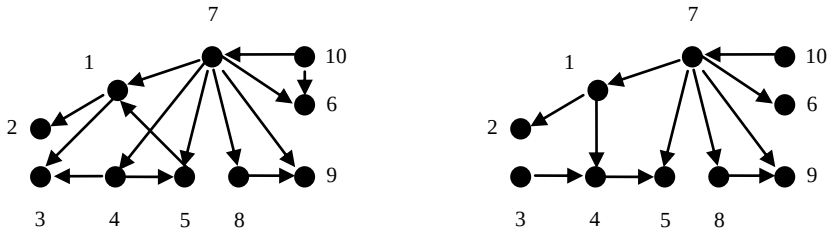
а б

Рисунок 2.14 – Експеримент №3 (а) побудована (б) оригінальна БМ



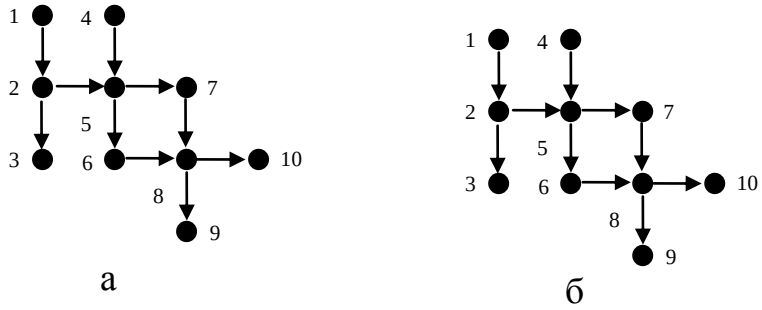
а б

Рисунок 2.15 – Експеримент №4 (а) побудована (б) оригінальна БМ



а б

Рисунок 2.16 – Експеримент №5 (а) побудована (б) оригінальна структура БМ



а б

Рисунок 2.17 – Експеримент №6 (а) побудована (б) оригінальна структура БМ

2.3 Розробка алгоритму формування ймовірнісного висновку в байєсівській мережі на основі навчальних даних

Вхідні дані, необхідні для алгоритму формування висновку:

1. Множина навчальних даних $D=\{d_1,...,d_n\}$, де $d_i=\{x_i^{(1)}x_i^{(2)}...x_i^{(N)}\}$ (нижній індекс – номер спостереження, а верхній – номер змінної), n – кількість

спостережень. Кожне спостереження складається з N ($N \geq 2$) змінних $X^{(1)}, X^{(2)},..., X^{(N)}$, кожна j-та змінна ($j = 1,..., N$) має $A^{(j)} = \{0,1,..., \alpha^{(j)} - 1\}$ ($\alpha^{(j)} \geq 2$) станів.

2. Структура БС g представлена множиною з N предків $(\Pi^{(1)},...,\Pi^{(N)})$, тобто для кожної вершини $j = 1,..., N$; $\Pi^{(j)}$ – множина батьківських вершин, при цьому $\Pi^{(j)} \subseteq \{X^{(1)},..., X^{(N)}\} \setminus \{X^{(j)}\}$ (вершина не може бути батьком для самої себе, тобто петлі в графі відсутні).

3. Множина інстанційованих вершин $\{X^{(P_1)} = x^{(P_1)},..., X^{(P_v)} = x^{(P_v)}\}$, тобто вершин, що перебувають у деякому певному стані з одиничною ймовірністю. Якщо множина інстанційованих вершин порожня, то потрібно використати ймовірнісний висновок, який грунтується на класичній теоремі Байєса.

Послідовність формування висновку за алгоритмом ймовірнісного висновку на основі навчальних даних

Крок 1. За множиною навчальних даних обчислюється матриця емпіричних значень спільного розподілу ймовірностей всієї мережі $P(X^{(1)},..., X^{(N)})$, за формулою:

$$P_{matrix}(X^{(1)} = x^{(1)},..., X^{(N)} = x^{(N)}) = \frac{n[X^{(1)} = x^{(1)},..., X^{(N)} = x^{(N)}]}{n}$$

де n – кількість навчальних спостережень, $x^{(j)} \in A^{(j)}$, а

$$n[X^{(1)} = x^{(1)},..., X^{(N)} = x^{(N)}] = \sum_{j=1}^{n} I(X^{(1)} = x_j^{(1)},..., X^{(N)} = x_j^{(N)})$$

де функція $I(E) = 1$, якщо предикат $E = true$, інакше $I(E) = 0$.

Далі обчислюються значення ймовірностей всіх можливих станів, неінстанційованих за алгоритмом, наведеним на рисунку 2.18.

$for \quad j=1 \quad to \quad N \quad if \quad X^j \notin \{X^{(P_1)},...,X^{(P_v)}\} \quad then$

$begin$

$sum = 0;$

$\forall x^{(j)} \in A^{(j)} \quad do$

$begin$

$for \quad k=1 \quad to \quad last\_string\_matrix \quad do$

$begin$

$if \quad (X^{(P_1)}_{matrix} = x^{(P_1)}) \quad and \quad ... \quad and \quad (X^{(P_v)}_{matrix} = x^{(P_v)})$

$and \quad (X^{(j)}_{matrix} = x^{(j)}) \quad then$

$begin$

$P(X^{(j)} = x^{(j)}) = P(X^{(j)} = x^{(j)}) + P_{matrix}(X^{(1)}_{matrix},...,X^{(N)}_{matrix});$

$end;$

$end;$

$sum = sum + P(X^{(j)} = x^{(j)});$

$end;$

$\forall x^{(j)} \in A^{(j)} \quad do$

$begin$

$$P(X^{(j)} = x^{(j)}) = \frac{P(X^{(j)} = x^{(j)})}{sum};$$

$end;$

$end;$

Рисунок 2.18 – Алгоритм обчислення значень ймовірностей всіх можливих станів неінстанційованих вершин

Крок 2. Перебираємо послідовно всі вершини БМ. Якщо вершина не інстанційована, то потрібно обчислити значення ймовірностей всіх можливих станів цієї вершини. Для цього виконується послідовний перебір всіх рядків матриці емпіричних значень спільного розподілу ймовірностей всієї мережі. Якщо значення вершин рядка збігаються зі значеннями інстанційованих вершин і станом аналізованої вершини, то відповідне значення $P_{matrix}(X^{(1)},...,X^{(N)})$ додається до значення ймовірності відповідного стану аналізованої вершини. Після цього

нормуються значення ймовірностей станів аналізованої вершини:

В другому розділі для розв’язання проблеми нелінійної поліноміальної складності задачі побудові БМ запропоновано новий евристичний алгоритм побудови БМ [16], який грунтується на використанні оцінки взаємної інформації між вершинами та функції ОМД. Даний евристичний алгоритм ітераційний і дозволяє значно зменшити обчислювальну складність навчання БМ. На першому етапі виконується обчислення значення взаємної інформації між всіма вершинами. На другому виконується цілеспрямований пошук, що використовує в якості оціночної функції мережевих структур функцію ОМД, яка грунтується на принципі опису, що застосовується на кожній ітерації алгоритму побудови.

Використання евристичного алгоритму побудови мереж істотно розширює можливості використання БМ при виконанні аналізу в різних областях людської діяльності, особливо там, де доводиться працювати з великими обсягами інформації.

Також запропоновано алгоритм формування ймовірнісного висновку в БМ на основі навчальних даних. Цей алгоритм відноситься до точних, але на відміну від інших точних методів та алгоритмів його швидкодлія залежить не від кількості дуг мережі, а від розміру навчальної вибірки. Але головною перевагою запропонованого алгоритму є його простота застосування та можливість швидкої реалізації на будь-якій мові програмування.

3 СИСТЕМА ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ ОЦІНКИ КРЕДИТНИХ РИЗИКІВ НА ОСНОВІ БАЙЄСІВСЬКИХ МЕРЕЖ

3.1 Структура системи підтримки прийняття рішень

Даний розділ дипломного проекту присвячено проектуванню та реалізації СППР, створеної на основі запропонованих в попередньому розділі роботи алгоритмів побудови топології та ймовірнісного висновку в БМ.

СППР представляє собою комп'ютерну систему, що спрямована на розв'язання таких задач ІАД, як аналіз зв'язків, класифікація, кластеризація, прогнозування та візуалізація. Роз'язання цих задач здійснюється без залучення експертів – за навчальними даними; головна мета цієї СППР полягає у наданні допомоги особам, що приймають рішення (ОПР).

СППР має гнучку архітектуру, що передбачає введення нових алгоритмічних модулів для реалізації будь-яких алгоритмів побудови БМ та ймовірнісного висновку, або навіть інших методів ІАД.

За рівнем користувача розроблена СППР належить до активної СППР, тобто ця система допомагає ОПР обрати рішення, користуючись значеннями ймовірностей виникнення тієї чи іншої події, а не тільки допомагає прийняття рішення шляхом виконання складних обчислювальних операцій [19].

За технічним рівнем розроблена СППР належить до настільної СППР, тобто ця система обслуговує лише один комп'ютер користувача [20], автором дисертаційної роботи не ставилася мета створення мережевої СППР.

За інструментальним підходом класифікації СППР, запропонована СППР відноситься до СППР-генератора, тому що кожна БМ, яка побудована на основі навчальних даних, представляє собою причинно-наслідкову мережу, що може розглядатись як окрема СППР [21].

СППР дозволяє без залучення експертів будувати моделі у вигляді БМ, за навчальними даними та вирішувати задачі класифікації, кластеризації та прогнозування шляхом формування відповідного ймовірнісного висновку.

Архітектура будь-якої СППР у спрощеному вигляді має структуру, наведену

в додатку А, або близьку до цієї структури.

В додатку Б наведена структурна схема розробленої СППР, що призначена для (1) накопичення даних; (2) побудови ймовірнісних моделей у вигляді БМ та (3) формування ймовірнісного висновку відносно заданого того чи іншого стану процесу. Система складається з трьох основних підсистем і передбачає модульно-блочну побудову.

Пристрої вводу-виводу надають користувачу можливість завантажувати дані в СППР. Для цього підсистема вводу-виводу функціонально зв'язана з підсистемою інтерфейсу користувача.

Підсистема інтерфейсу користувача призначена для здійснення зв'язку між користувачами СППР та внутрішніми елементами системи і забезпечує ввід та вивід інформації для ОПР і експертів, а також надає доступ до зовнішніх запам'ятовуючих пристроїв ПЕОМ. Інтерфейс дозволяє операторові вводити інформацію, дані, команди, параметри і запити в систему та одержувати вихідну інформацію в зручному для сприйняття вигляді.

В зв'язку з тим, що система повинна обробляти досить велике число запитів різного типу, виникає проблема проектування та реалізації зручного інтерфейсу користувача. Для розв'язання цієї проблеми використано підхід на основі багатовіконного діалогу з використанням контекстної допомоги на кожному етапі прийняття рішення.

Для організації діалогу користувач–система в СППР реалізовані наступні запити мовної системи:

– запити на формування конкретних процедур обробки даних та прогнозування (формулювання вимог);

– запити на вибір та формулювання критеріїв розв'язку задачі;

– запити на виконання задач моделювання і прогнозування;

– запити на форму представлення результатів.

Головною з точки зору обробки даних СППР є система обробки даних та генерації результатів. Вона сприймає коректні запити користувача і виконує наступні задачі:

– читання необхідних даних у вигляді таблиць навчальних даних для побудови моделей;

– вибір методів побудови БМ;

– запуск на виконання модулів обробки даних та прогнозування;

– формування результатів обробки даних та їх зберігання в короткостроковій та довгостроковій пам’яті;

– генерування діагностичних повідомлень.

Підсистема зберігання інформації. Підсистема зберігання інформації складається з бази даних (БД), бази моделей (БМод) та бази знань (БЗ), які призначені для накопичення даних, моделей у вигляді структур БМ, таблиць умовних ймовірностей та метаданих.

База даних. В рамках розробленої СППР реалізовано спрощений варіант БД – текстова БД, що представляє собою сукупність навчальних даних у текстовому вигляді. Кожний такий файл включає всі навчальні дані для всіх змінних (атрибутів) процесу, тобто перший стовбець – дані для першої змінної, другий стовбець – дані для другої змінної і т. ін. Файл даних повинен містити не менше трьох стовпців даних, тобто процес повинен описуватися не менш як трьома змінними. Дані в одному рядку (між стовпцями) повинні бути відділені один від одного пробілом, знаком табуляції або крапкою з комою. В дійсних числах для відокремлення десяткової частини числа від цілої використовується кома, наприклад 10,2.

Змінні можуть містити не тільки кількісні (числові), але і якісні (строкові) значення (наприклад, вербальні оцінки вигляду false/true або yes/no). Але в цьому випадку не допускаються пробіли усередині строкового значення, тому що значення виду "Married more 10" при завантаженні даних призведе до некоректної роботи програми. Тому такі строкові значення, які складаються з декількох слів, повинні бути замінені на такі ж строкові значення, але без пробілів, знаків табуляції та крапок з комою. Наприклад, строкове значення "Married more 10" можна замінити на "Married_more_10" (замість пробілу використається нижній знак підкреслення) або "MarriedMore10" (кожне нове слово починається з великої

букви). Редагувати довгі строкові значення можна, наприклад, за допомогою програми Microsoft Excel або текстового редактора.

Головна підсистема аналізу (ГПА) складається з трьох блоків: (1) обробки даних; (2) побудови структури БМ; (3) побудови ймовірнісного висновку. Ця система призначена для аналізу даних від користувача системою, побудови БМ та ймовірнісного висновку за цими даними. ГПА отримує дані по шині даних від підсистеми зберігання інформації.

Блок обробки даних призначений для перевірки даних, що поступають від ГПА в блок обробки даних на наявність пропусків та викидів (екстремальних значень), а також для здійснення операції агрегування над цими даними. Блок обробки даних складається з чотирьох модулів: (1) аналізу неповних даних; (2) видалення записів з неповними даними; (3) заповнення неповних даних; (4) агрегування даних.

Модуль аналізу повноти даних призначений для перевірки вхідних навчальних даних на наявність пропусків та викидів. Модуль аналізу повноти даних зв'язаний з модулями: (1) видалення записів з неповними даними, (2) заповнення неповних даних та (3) агрегування даних. Якщо запис навчального набору даних, що аналізується, містить невелику кількість пропусків (відсутніх даних), то цей запис передається для заповнення найбільш ймовірним значенням для даного запису в модуль заповнення неповних даних, інакше такий запис видаляється з множини навчальних даних в модулі видалення записів з неповними даними.

Модуль видалення записів з неповними даними призначений для видалення записів з навчальних даних, що мають велику кількість пропусків.

Модуль заповнення неповних даних призначений для заповнення записів з навчальних даних найбільш ймовірним значенням, для цього пропонується використовувати спрощену БМ.

Модуль агрегування даних призначений для виконання операції агрегування даних відносно множини навчальних даних. Операція агрегування здійснюється наступним чином: якщо якийсь вузол з множини навчальних даних приймає

числові значення, то ці числові значення замінюються інтервальними оцінками.

Блок побудови структури БМ складається з таких модулів: (1) обчислення ТУН та (2) пошуку оптимальної структури БМ. Цей блок призначений для побудови структури БМ за даними, що передаються з блока обробки даних.

Модуль обчислення ТУН призначений для обчислення матриці взаємної залежності між вершинами (факторами процесу). В СППР в якості ТУН використовується значення взаємної інформації.

Модуль пошуку оптимальної структури БМ реалізує цілеспрямований пошук оптимальної структури БМ; в якості критерію оптимальності застосовується функція опису мінімальною довжиною. Докладніше опис наведено в третьому розділі дисертаційної роботи – евристичний метод побудови БМ за навчальними даними.

Блок побудови ймовірнісного висновку призначений для побудови ймовірнісного висновку по структурі БМ, що надходить з блоку побудови структури БМ. Даний блок складається з чотирьох модулів: (1) перевірки наявності інстанційованих вершин; (2) побудови таблиці умовних ймовірностей; (3) побудови матриці емпіричних значень спільного розподілу ймовірностей всієї мережі; (4) побудови ймовірнісного висновку.

Модуль перевірки наявності інстанційованих вершин. Якщо множина інстанційованних значень порожня, то СППР передає керування в модуль побудови ТУЙ, інакше модулю побудови матриці емпіричних значень спільного розподілу ймовірностей всієї мережі.

Модуль побудови таблиць умовних ймовірностей призначений для побудови ТУЙ кожного вузла БМ за множиною навчальних даних.

Модуль побудови матриці емпіричних значень спільного розподілу ймовірностей всієї мережі призначений для побудови матриці емпіричних значень сумісного розподілу ймовірностей всієї мережі Байєса за навчальними даними.

Модуль побудови ймовірнісного висновку. В залежності від того, від якого модуля прийшов сигнал, виконуються відповідні обчислення. Якщо сигнал прийшов від модуля побудови ТУЙ, то застосовується класичний метод прямого

розповсюдження ймовірностей по БМ. Спочатку обчислюються ймовірності значень кореневих вершин, тобто вершин, у яких відсутні батьківські вершини. Після цього визначаються ймовірності усіх інших вузлів за формулою Байєса і ТУЙ. Якщо сигнал прийшов від модуля побудови матриці емпіричних значень спільного розподілу ймовірностей всієї мережі, то розрахунки виконуються у такий спосіб.

На основі інформації про структуру моделі БМ та інстанційовані значення виконується послідовний перебір всіх вузлів БМ. Якщо вузол не інстанційований, то виконується розрахунок усіх можливих станів цього вузла. Після цього виконується послідовний перебір усіх рядків матриці емпіричних значень спільного розподілу ймовірностей всієї БМ і, якщо значення вузла рядка співпадає із значенням інстанційованних вузлів і значеннями аналізованого вузла, то відповідне значення додається до значення ймовірності відповідного стану аналізованого вузла. Після цього виконується нормування значень ймовірностей станів аналізованого вузла.

3.2 Програмна реалізація системи підтримки прийняття рішень

На основі запропонованої оригінальної архітектури СППР (див. додаток Б) із використанням мови програмування Delphi реалізована комп’ютерна програма для побудови мереж та ймовірнісного висновку за навчальними даними. Мова програмування Delphi є об’єктно-орієнтованим засобом програмування, що дозволяє представити розроблені алгоритми у вигляді ієрархії класів з наслідуваними властивостями та методами. Це, в свою чергу, дає можливість одержати найбільш ефективну структуру алгоритмів, забезпечує підвищену надійність розроблювального програмного продукту, та зручність його подальшого супроводу.

Для нормальної експлуатації програми необхідна наявність комп’ютера з наступними характеристиками:

– тактова частота процесора не менше 300 Мгц;

– 4 Мбайт вільного місця на жорсткому диску для програмних модулів системи та 4 Мбайт для навчальних даних і формування баз даних (останнє значення не фіксоване і може змінюватися в залежності від умов експлуатації);

– оперативна пам'ять 32 Мбайт і більше;

– операційна система Windows 98/2000/NT/XP;

– пристрій для забезпечення безперебійної роботи комп’ютера для можливості автономної роботи програми;

– клавіатура та комп’ютерна мишка;

– пристрій для запису даних і результатів на оптичні носії для резервного копіювання файлів баз даних.

Програма надає такі можливості: будувати структуру та формувати ймовірнісний висновок в байєсівській мережі.

3.2.1 Побудова структури байєсівської мережі

3.2.1.1 Завантаження даних в програму

Програма призначена для роботи з текстовими файлами. Кожен такий файл повинен мати не менше трьох стовпців даних. Тобто програма працює не менше ніж з трьома вершинами. Для того щоб програмі вказати ім’я файлу з даними, необхідно вибрати опцію “Побудова структури БМ за даними” в головному меню та натиснути “Завантажити дані з файлу” (рисунок 3.1); після цього в діалоговому вікні необхідно вказати шлях до файлу на жорсткому диску (рисунок 3.2).

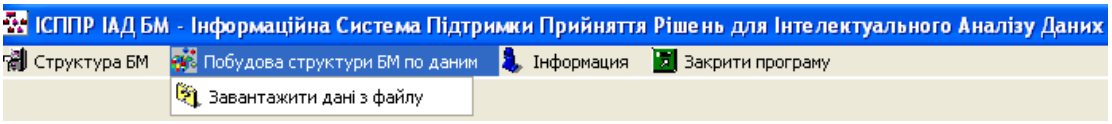


Рисунок 3.1 – Завантаження даних з файлу

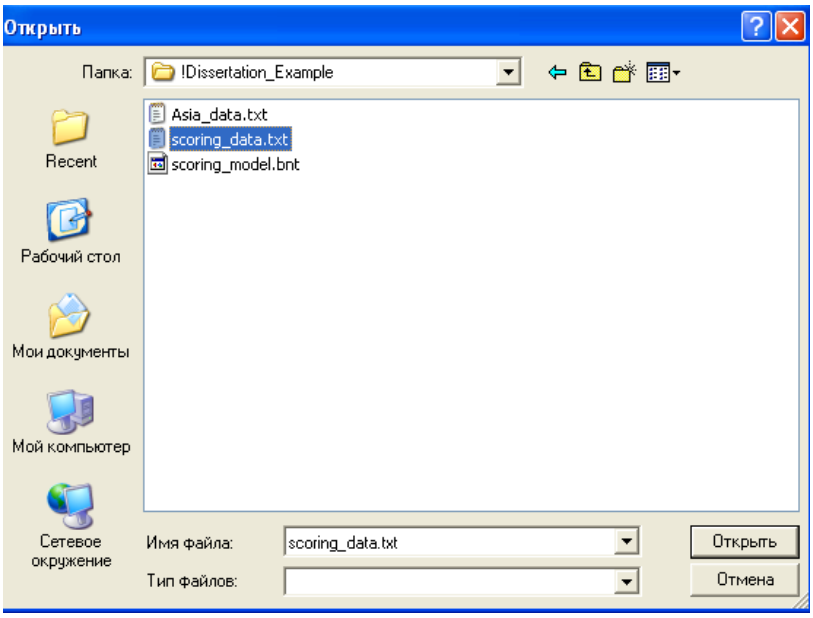


Рисунок 3.2 – Діалогове вікно завантаження даних з файлу

Після того як файл вказано, виконується завантаження даних в таблицю, рисунок 3.3. Стовпці таблиці – фактори (параметри, змінні), за якими буде будуватися топологія БМ. Надалі ці фактори будуть представлені у вигляді вершин байєсівської мережі. Рядки таблиці – це навчальні записи.

good_normal_bad_	Age	TotalCreditAmount	ContactPerson	JobPosition	Children	MaritalStatus	Education	Gender	SpouseWorks
good	29	370	absent	MM	one	MARRIED	HIGH	Male	True
good	32	480	absent	SP	zero	CIVILMARRIAGE	HIGH	Female	True
good	24	1360	absent	SP	zero	SINGLE	HIGH	Female	False
good	22	740	absent	DV	zero	SINGLE	HIGHNF	Female	False
good	21	438	absent	SP	zero	MARRIED	HIGH	Male	True
good	26	1390	absent	SP	one	MARRIED	HIGH	Male	False
good	22	2374	absent	SP	zero	SINGLE	SECSP	Male	False
bad	26	2110	absent	AS	zero	CIVILMARRIAGE	SECSP	Male	True
good	54	2395	absent	AS	zero	MARRIED	SEC	Female	True
good	35	2582	absent	TM	two	MARRIED	SECSP	Male	True
good	22	1000	absent	DV	zero	SINGLE	SECSP	Male	False
good	49	4055	absent	TM	one	WIDOWED	HIGH	Female	False
bad	28	2400	absent	MM	one	MARRIED	HIGHNF	Male	False
good	26	1650	absent	AS	zero	CIVILMARRIAGE	SECSP	Female	True

Рисунок 3.3 – Таблиця з даними

3.2.1.2 Вибір функції оцінки моделі

Для оцінювання моделей в програмі реалізовані наступні функції:

– опису мінімальною довжиною;

– Купер-Герсковича;

– модифікація функції Купера-Герсковича.

Для вибору бажаної функції треба поставити крапку навпроти відповідної функції, за замовчуванням використовується ОМД (рисунок 3.4).

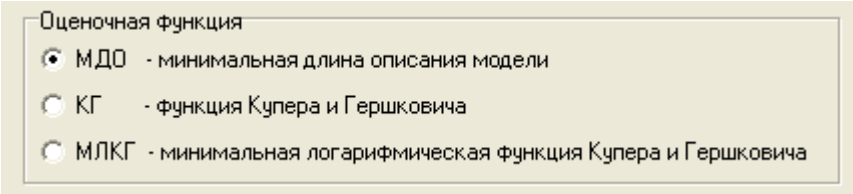


Рисунок 3.4 – Типи функцій оцінювання моделі

3.2.1.3 Агрегування даних

Натисканням лівої кнопки миші на стовпець таблиці даних (див. рисунок 3.3) можна дізнатися про тип фактора – ця інформація відображається зверху вікна. Програма підтримує роботу з числовими (рисунок 3.5) та рядковими (рисунок 3.6) значеннями факторів.

Рисунок 3.5 – Тип змінної – число

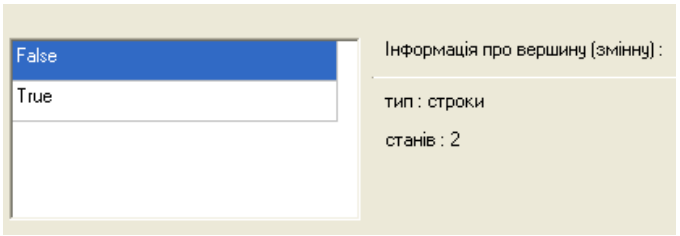


Рисунок 3.6 – Тип змінної – строкова

Для факторів типу „число” повинна бути виконана операція агрегування даних. В полі “Кількість інтервалів розбиття” вказується, на яку кількість інтервалів розбивається множина значень що приймаються фактором, рисунок 3.7.

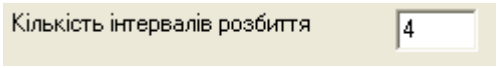


Рисунок 3.7 – Кількість інтервалів розбиття

Кількість інтервалів для числових даних можна не вказувати (рисунок 3.8), в цьому випадку програма за замовчуванням автоматично виконає агрегування всіх числових даних до двох інтервалів.

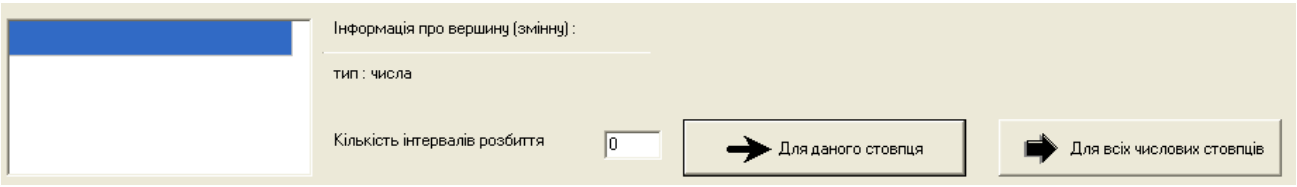


Рисунок 3.8 – Розбиття на інтервали

Вказавши кількість інтервалів розбиття і натиснувши кнопку “Для всіх числових інтервалів” буде виконано агрегування всіх числових даних до заданої кількості інтервалів для всіх числових стовпців даних з таблиці даних (див. рисунок 3.3).

Після того як для числових факторів задана кількість інтервалів розбиття, потрібно натиснути на робочій формі “Завантаження даних” кнопку “Продовжити” (рисунок 3.9).

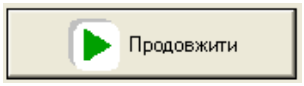


Рисунок 3.9 – Кнопка “Продовжити”

Після завершення операції агрегування даних числові фактори вже складатимуться не з чисел, а з значень станів, які представляють собою інтервальні оцінки. Після цього знову потрібно натиснути кнопку “Продовжити” (див. рисунок 3.9). З'явиться інформаційне вікно, яке повідомляє яка кількість ітерацій буде виконана, рисунок 3.10.

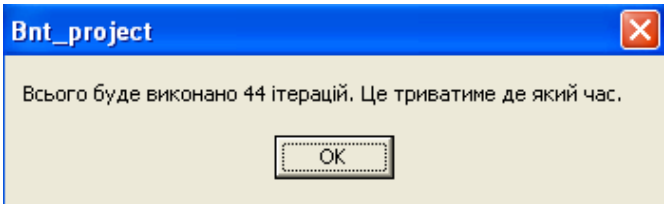
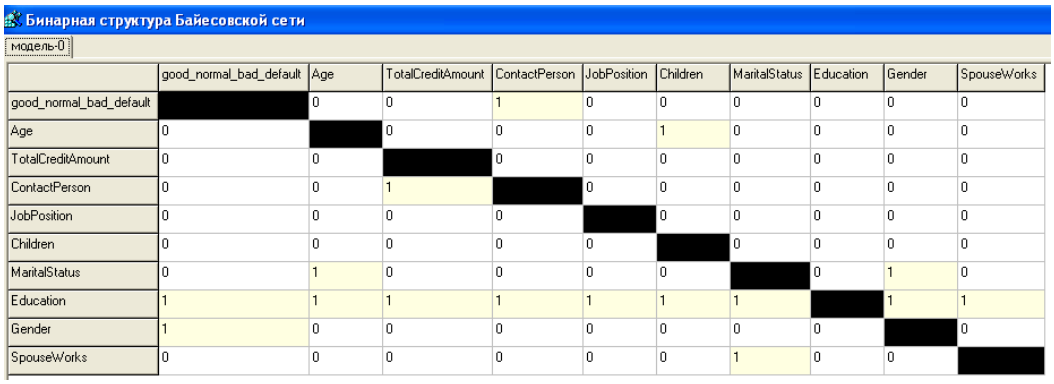


Рисунок 3.10 – Інформаційне вікно програми

Натисніть кнопку “OK”. Програма виконає побудову матриці взаємних значень факторів, а потім за алгоритмом евристичного методу побудує структуру БМ.

3.2.1.4 Відображення результатів

Побудована структура БМ відображається на робочій формі програми “Бінарна структура байєсівської мережі” у вигляді бінарної таблиці зв’язків, рисунок 3.11.



	good_normal_bad_default	Age	TotalCreditAmount	ContactPerson	JobPosition	Children	MaritalStatus	Education	Gender	SpouseWorks
good_normal_bad_default		0	0	1	0	0	0	0	0	0
Age	0		0	0	0	1	0	0	0	0
TotalCreditAmount	0	0		0	0	0	0	0	0	0
ContactPerson	0	0	1		0	0	0	0	0	0
JobPosition	0	0	0	0		0	0	0	0	0
Children	0	0	0	0	0		0	0	0	0
MaritalStatus	0	1	0	0	0	0		0	1	0
Education	1	1	1	1	1	1	1		1	1
Gender	1	0	0	0	0	0	0	0		0
SpouseWorks	0	0	0	0	0	0	1	0	0	

Рисунок 3.11 – Бінарна таблиця зв’язків мережі

Для виведення чисельних значень функції ОМД, отриманих на ітераціях алгоритму побудови структури, треба натиснути на кнопку “Відкрити звіт”, рисунок 3.12.

Рисунок 3.12 – Виведення значень функції ОМД

В результаті цієї дії з’явиться форма з проміжними значеннями ОМД, рисунок 3.13.

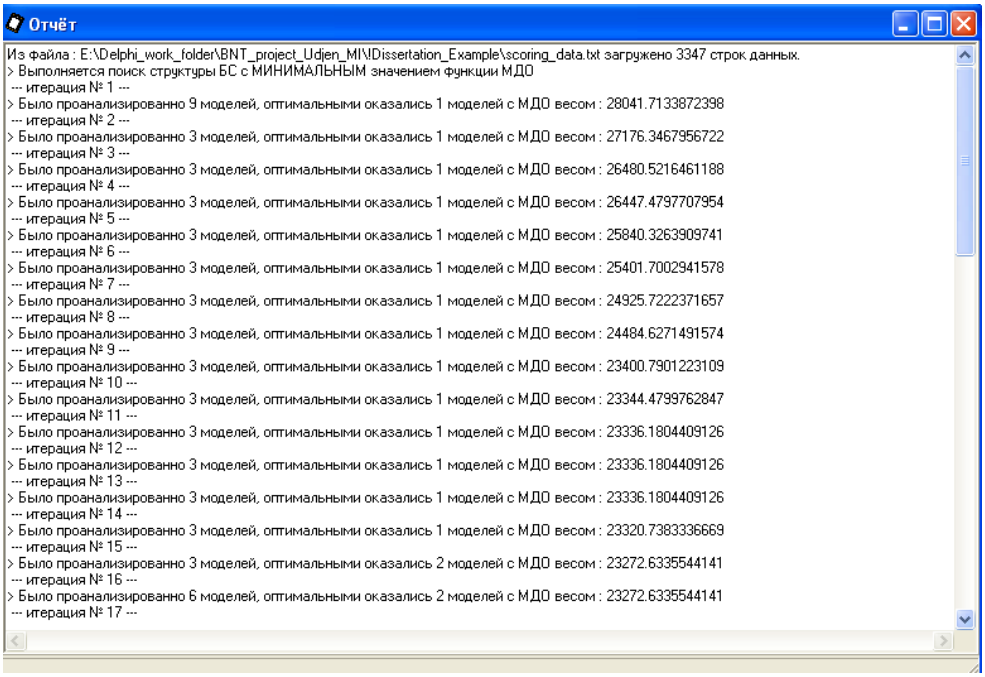


Рисунок 3.13 – Вікно звіту проміжних значень ОМД

Для виводу побудованої БМ у вигляді причинно-наслідкової мережі (додаток В) треба натиснути кнопку “Відобразити байєсівську мережу” (рисунок 3.14).

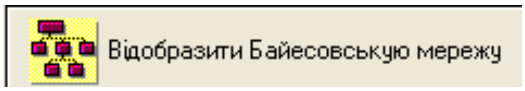


Рисунок 3.14 – Кнопка виводу БМ

3.2.1.5 Робота з мережею

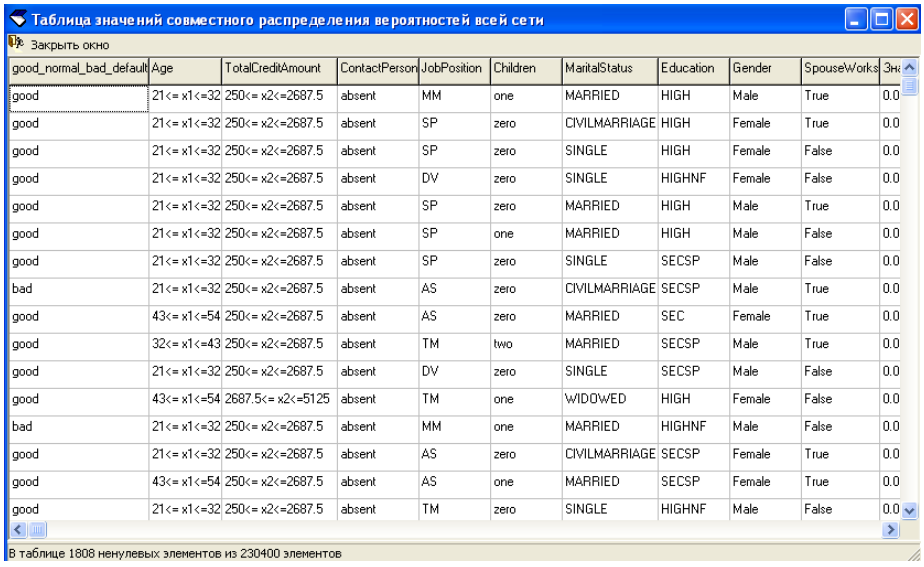
Натиснувши ліву кнопку миші і утримуючи її, можна переміщати вершини мережі по формі програми. Подвійним натисненням лівої кнопки миші на вершині БМ виводиться форма з ТУЙ мережі (рисунок 3.15).

	JobPosition					
Education	AS	DV	MM	PE	SP	TM
HIGH	0.0305	0.0776	0.1734	0.011	0.4781	0.2295
HIGHNF	0.0693	0.0601	0.1109	0.0416	0.5701	0.1479
PHD	0	0	0.2	0	0.4	0.4
SEC	0.2171	0.0448	0.0651	0.0041	0.5556	0.1133
SECSP	0.1483	0.0586	0.1284	0.0173	0.5336	0.1137

Рисунок 3.15 – Приклад ТУЙ вершини БМ

З таблиці умовних ймовірностей, натиснувши F5 або кнопку “Таблиця значень всієї мережі” (рисунок 3.16), можна вивести форму з таблицею значень спільного розподілу ймовірностей всієї БМ (рисунок 3.17).

Рисунок 3.16 – Кнопка “Таблиця значень всієї мережі”



good_normal_bad_default	Age	TotalCreditAmount	ContactPerson	JobPosition	Children	MaritalStatus	Education	Gender	SpouseWorks	Зн
good	21<= x1<=32	250<= x2<=2687.5	absent	MM	one	MARRIED	HIGH	Male	True	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	SP	zero	CIVILMARRIAGE	HIGH	Female	True	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	SP	zero	SINGLE	HIGH	Female	False	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	DV	zero	SINGLE	HIGHNF	Female	False	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	SP	zero	MARRIED	HIGH	Male	True	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	SP	one	MARRIED	HIGH	Male	False	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	SP	zero	SINGLE	SECSP	Male	False	0.0
bad	21<= x1<=32	250<= x2<=2687.5	absent	AS	zero	CIVILMARRIAGE	SECSP	Male	True	0.0
good	43<= x1<=54	250<= x2<=2687.5	absent	AS	zero	MARRIED	SEC	Female	True	0.0
good	32<= x1<=43	250<= x2<=2687.5	absent	TM	two	MARRIED	SECSP	Male	True	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	DV	zero	SINGLE	SECSP	Male	False	0.0
good	43<= x1<=54	2687.5<= x2<=5125	absent	TM	one	WIDOWED	HIGH	Female	False	0.0
bad	21<= x1<=32	250<= x2<=2687.5	absent	MM	one	MARRIED	HIGHNF	Male	False	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	AS	zero	CIVILMARRIAGE	SECSP	Female	True	0.0
good	43<= x1<=54	250<= x2<=2687.5	absent	AS	one	MARRIED	SECSP	Female	True	0.0
good	21<= x1<=32	250<= x2<=2687.5	absent	TM	zero	SINGLE	HIGHNF	Male	False	0.0

Рисунок 3.17 – Приклад таблиці значень спільного розподілу ймовірностей всієї БМ

Збереження структури БМ в файл на диск. Треба натиснути кнопку “Зберегти структуру мережі” (рисунок3.18). Файл зберігається з розширенням **.bnt*.

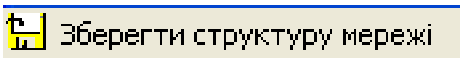


Рисунок 3.18 – Збереження структури БМ в файл

Для переходу в режим побудови ймовірнісного висновку потрібно або закрити програму і запустити її наново, або закрити всі вікна до самої початкової форми.

3.2.2 Побудова ймовірнісного висновку

3.2.2.1 Завантаження структури мережі

Для того щоб програмі вказати ім’я файлу із збереженою структурою БМ, необхідно в головному меню програми вибрати закладку “Структура БМ” та натиснути кнопку “Завантажити файл структури БМ” (рисунок 3.19); після цього в діалоговому вікні вказати шлях до файлу (див. рисунок 3.2).

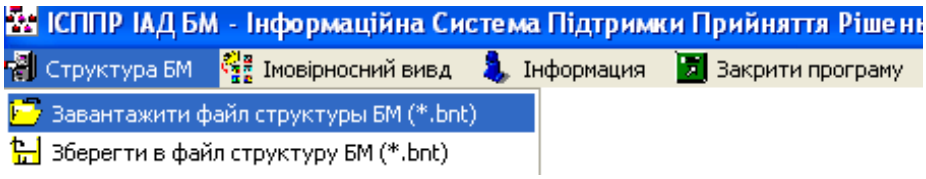


Рисунок 3.19 – Завантаження файлу з структурою БМ

3.2.2.2 Відображення значень ймовірнісного висновку

Після того як структура БМ з’явилася на формі програми, можна будувати ймовірнісний висновок, для цього в головному меню програми треба обрати закладку “Ймовірнісний висновок” та обрати тип представлення у вигляді окремих таблиць або однієї таблиці значень ймовірностей станів вершин (рисунок 3.20).

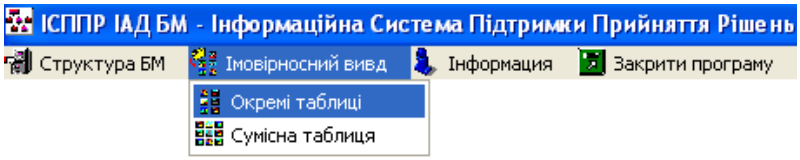


Рисунок 3.20 – Вибір типу представлення ймовірнісного висновку

3.2.2.3 Представлення у вигляді окремих таблиць

Для побудови ймовірнісного виводу в БМ за навчальними даними та представлення результатів виводу у вигляді окремих таблиць, треба натиснути на пункт меню “Окремі таблиці” (рисунок 3.21). Після цього з’явиться форма з таблицями, кожна таблиця – це вершина БМ.

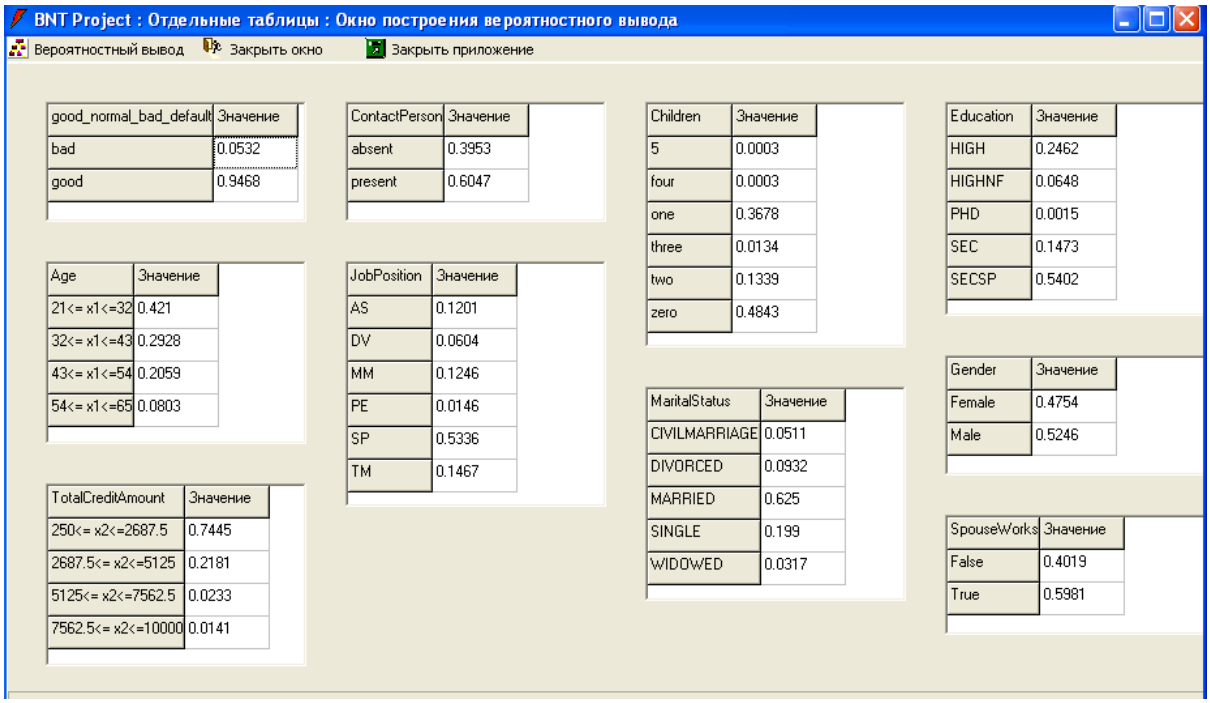


Рисунок 3.21 – Приклад представлення ймовірнісного висновку у вигляді окремих таблиць

Натиснувши ліву кнопку миші і утримуючи її, можна переміщати таблиці по екрану. Подвійним натисненням лівої кнопки миші в таблиці виконується виділення інстанційованого значення вершини (фактора). Після натискування F5 або кнопки “Точний метод” формується ймовірнісний висновок (див. рисунок 3.21). В програмі реалізовано алгоритм точного методу побудови ймовірнісного висновку за навчальними даними (рисунок 3.22).

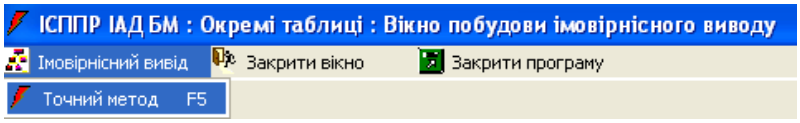


Рисунок 3.22 – Закладка “Точний метод”

3.2.2.4 Представлення у вигляді загальної таблиці

Для побудови ймовірнісного висновку в БМ за навчальними даними та представлення результатів виводу у вигляді однієї загальної таблиці, треба натиснути на пункт меню “Спільна таблиця” (рисунок 3.23).

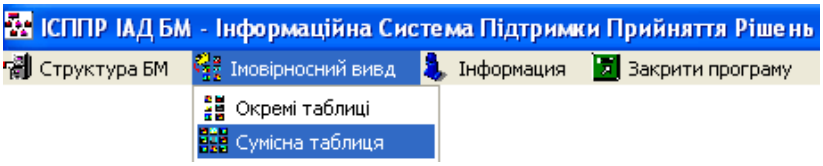


Рисунок 3.23 – Приклад представлення ймовірнісного висновку у вигляді загальної таблиці

Після цього з’явиться форма із загальною таблицею (рисунок 3.24). Перший стовпчик таблиці – назви вершин, другий – назви станів вершин, а третій – чисельні значення ймовірностей станів вершин мережі. Подвійним натисненням лівої кнопки миші в таблиці виконується виділення інстанційованого значення вершини (фактора). Після натискування F5 або кнопки “Точний метод” здійснюється побудова ймовірнісного висновку (див. рисунок 3.22).

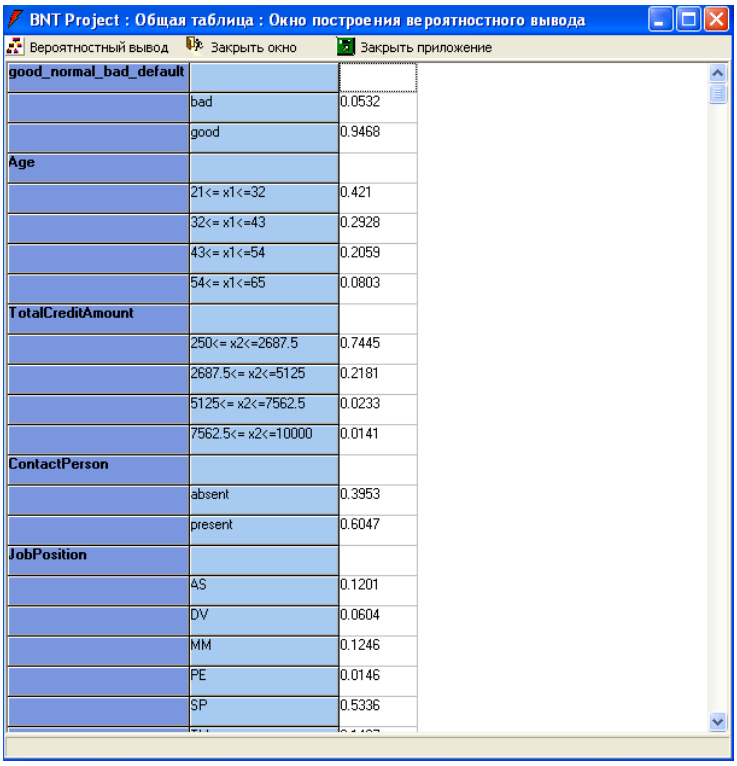


Рисунок 3.24 – Загальна таблиця станів та чисельних значень ймовірностей всієї побудованої мережі Байєса

Код програми системи підтримки прийняття рішень для інтелектуального аналізу даних на основі байєсівських мереж представлено в додатку Г.

3.3 Практичне використання програми

Розроблена СППР використана для розв’язання задачі побудови скорингової моделі оцінювання кредитоспроможності фізичних осіб при отриманні споживчого кредиту в банку. Для побудови скорингової моделі використана база даних клієнтів одного з українських комерційних банків. Усього в базі 3347 записів про клієнтів. Як змінні процесу обрані такі десять атрибутів: 1. Стать; 2. Вік; 3. Сімейний стан; 4. Кількість дітей; 5. Чоловік (дружина) працює; 6. Освіта; 7. Тип трудозайнятості; 8. Поручитель; 9. Сума кредиту; 10. Результат (кредитоспроможний?).

В додатку Д наведена система кредитного скорингу у вигляді БМ, побудованої за допомогою реалізованої програми. В таблиці 3.1 наведено десять ситуацій, для яких побудовано ймовірнісний висновок.

Таблиця 3.1 – Результати моделювання по БМ

Номер ситуації	Інстанційовані вершини (апріорна інформація, щодо клієнта)	Вірогідність того, що клієнт поверне кредит
1	Стать = “Чоловік”	92,08%
2	Стать = “Жінка”	97,55%
3	Поручитель = “Так”	99,06%
4	Поручитель = “Ні”	87,98%
5	Вік < 32 років та Сімейний стан = “Самотній” та Сума кредиту > 5000	76,92%
6	Тип трудозайнятості =“Працівник банку” та Сімейний стан = “Одружений”	94,66%
7	Освіта =”Вища” та Кількість дітей = “один” та Чоловік (дружина) працює = “Так”	97,39%
8	Освіта =”Середня” та Кількість дітей = “немає” та Чоловік (дружина) працює = “Ні” та Поручитель = “Ні” та Сума кредиту > 2500	69,78%
9	Стать = “Чоловік” та Сімейний стан = “Удівець” та Освіта =”Середня спеціальна”	78,95%
10	Стать = “ Жінка” та Сімейний стан = “Удівець” та Освіта =”Середня спеціальна”	98,81%

Побудованій скоринговій моделі у вигляді БМ (див. додаток Г) відповідають наступні статистичні характеристики:

1 – похибка першого роду: 115;

2 – похибка другого роду: 157;

3 – загальна похибка: 272;

4 – загальна точність моделі: 0,918;

5 – похибка класифікації: 15%.

Тестова (перевірочна вибірка), що використовувалася для визначення якості прогнозу, складається із 100 записів. Процент похибок при класифікації дорівнює 15; це означає, що із 100 виданих кредитів 15 були невірно класифіковані.

Класифікація клієнтів за вибіркою, використаною для навчання, дала наступні результати:

1 – похибки 1-го роду дорівнюють 3,5% – прямі втрати банку, тобто використовуючи дану модель банк класифікував цих клієнтів як надійних, але вони виявилися некредитоспроможними;

2 – похибки 2-го роду дорівнюють 4,8 % – нереалізований дохід банку, тобто, використовуючи дану модель банк класифікував цих клієнтів як ненадійних, але вони виявилися кредитоспроможними.

3 – загальна похибка складає 8,3%.

У третьому розділі дипломного проекту розроблені алгоритми інтегровано у складі оригінальної архітектури СППР для ІАД на основі байєсівських мереж, яка відрізняється гнучкою побудовою і передбачає функціонально-блочну архітектуру. В запропонованій СППР експерти не приймають участь у побудові моделей, оскільки розроблені методи побудови моделей та формування ймовірнісного висновку призначені саме для автоматичного аналізу процесів за даними, що їх описують.

Запропонована СППР задовольняє основним характеристикам СППР: (1) використовує дані і моделі у вигляді БМ; (2) призначена для надання допомоги ОПР при прийнятті рішень для структурованих та неструктурованих задач; (3) підтримує, а не замінює рішення, що приймаються ОПР; (4) мета застосування

створеної системи – підвищення якості та ефективності рішень.

На основі запропонованої архітектури СППР розроблено комп’ютерне програмне забезпечення, яке за навчальними даними будує модель процесу у вигляді причино-наслідкової БМ. Виконано опис структури, основних можливостей та порядку використання розробленої програми для побудови моделей у вигляді БМ.

В даному розділі наведений практичний приклад використання програми для побудови моделей різних процесів, а саме скорингова система для оцінювання кредитоспроможності фізичних осіб у вигляді мережі Байєса для Тернопільського відділення ПАТ «Укрінбанк» (довідка про використання результатів дипломного проекту представлена в додатку Е).

4 ОХОРОНА ПРАЦІ

Метою даного дипломного проекту є розробка системи підтримки прийняття рішень оцінки кредитних ризиків. Тому потенційною аудиторією користувачів розробленого програмного забезпечення являються працівники фінансових та кредитних установ, які перебувають на своїх робочих місцях. Отже, особливу увагу варто приділити питанням охорони праці користувачів персональних комп'ютерів. Оскільки оператор ПК переважно оперує із візуальною інформацією, то окрім основних вимог до робочого місця та загальних вимог до навколишнього середовища, необхідно розглянути питання сприйняття саме зорової інформації.

4.1 Аналіз небезпечних і шкідливих виробничих факторів на робочому місці оператора

Система дистанційного контролю акустичного оточення керується та контролюються оператором ЕОМ, що розташований у звичайній кімнаті для працівників, або у спеціально відведеному для цього приміщенні.

На робочому місці оператора персонального комп'ютера присутні наступні шкідливі виробничі фактори:

Фізичні:

- недостатня освітленість робочої зони;
- пряма й відбита блискавичність;
- підвищений рівень статичної електрики;
- підвищений рівень електромагнітних випромінювань;
- підвищений рівень шуму;
- несприятливі параметри мікроклімату.

Психофізіологічні:

- статичні фізичні перевантаження;
- нервовопсихічні перевантаження (розумова перенапруга, перенапруга аналізаторів).

4.2 Фізично небезпечні й шкідливі фактори

Недостатня освітленість робочої зони природним світлом виникає внаслідок недостатньої площі світлових прорізів, їхнього забруднення, а також нераціонального розташування робочого стола щодо джерел природного світла.

Недостатнє освітлення негативно впливає на зір людини, стан його центральної нервової системи, знижує продуктивність праці, збільшує стомлення працівника.

Для виконання зорової роботи, пов'язаної зі сприйняттям інформації з екрана, зовсім непридатний спосіб освітлення всього приміщення. Наявний досвід створення світлової обстановки при сприйнятті інформації з екрана свідчить про те, що найбільший обсяг інформації може бути сприйнятий в темряві. Однак при необхідності реєстрації цієї інформації, яскравість робочого місця, де відбувається ця реєстрація, створювана місцевим освітленням, повинна відповідати яскравості екрана (75-100 кд/м^2). При цьому варто передбачити, щоб екран ЕПТ був захищений від прямого влучення на нього світла спеціальним щитом. При відсутності такого захисту й, отже, зменшенні контрасту зображення обсяг і точність сприйманої інформації може скоротитися на 30%.

Виконання зорової роботи при недостатньому освітленні може привести до розвитку деяких дефектів ока. Дефекти ока ділять на два основних види:

- короткозорість;
- далекозорість.

При організації раціонального виробничого освітлення варто уникати наявності в поле зору працюючих блисткостей. Порушення зорових функцій блисткістю називається сліпимістю. Чим вище яскравість поля адаптації, тим менше ймовірність явища сліпимості.

В умовах даного проекту природне освітлення є неможливим, тому що це закрите приміщення, а використовується штучне. Недостатня освітленість знижує швидкість розрізнення деталей (іноді робить це взагалі неможливим), що позначається на продуктивності праці, збільшує стомлюваність працівника й т.д.

Внаслідок цього необхідна розробка штучного освітлення робочої зони оператора.

Пряма блискість - влучення в поле зору яскравих джерел світла.

Відбита блискість виникає через високий коефіцієнт відбиття екрана.

Блискість викликає сліпучий ефект. Від сліпучої дії світла спочатку найбільше всього погіршується контрастна чутливість ока, потім гострота зору.

Границею дискомфортності зорового відчуття є М=40, а при М=60 виникають хворобливі відчуття.

Підвищений рівень статичної електрики. Джерелами електростатичного поля на робочому місці оператора є дисплей і периферійні пристрої. Вплив статичної електрики на людину може проявлятися у вигляді слабкого довгостроково струму, що протікає, або у формі короткочасного розряду через його тіло. Такий розряд викликає в людини рефлекторний рух, що може привести до травм. Електростатичне поле підвищеної напруженості негативно впливає на організм людини, викликаючи функціональні зміни з боку центральної нервової, серцево-судинної й іншої систем організму. Для обмеження шкідливого впливу електростатичного поля проводиться його нормування.

Напруженість електромагнітного поля на відстані 50 см. навколо ВДТ по електричній складовій повинна бути не більше:

- у діапазоні частот 5 Гц - 2 кГц; 25 В/м
- у діапазоні частот 2 - 400 кГц 2,5 В/м

Щільність магнітного потоку повинна бути не більше:

- у діапазоні частот 5 Гц - 2 кГц; 250 нТл
- у діапазоні частот 2 - 400 кГц. 25 нТл

Поверхневий електростатичний потенціал не повинен перевищувати 500 В.

Підвищений рівень електромагнітних випромінювань. Основним джерелом електромагнітних полів на робочому місці оператора персонального комп'ютера є електронно-променева трубка дисплея.

Електромагнітні поля впливають на тканині людини як на біологічні об'єкти. Вони змінюють орієнтацію кліток або ланцюгів молекул відповідно до напрямку

силових ліній електричного поля, послабляють біохімічну активність чистових молекул, порушують функції серцево-судинної системи, органів подиху, травлення й деяких біохімічних показників крові (змінюється співвідношення еритроцитів і лейкоцитів крові, виникає лейкоцитоз).

Електромагнітні поля несприятливо впливають на зір, викликають головний біль, порушення сну, зниження апетиту.

ЕПТ дисплея є джерелом електромагнітних випромінювань із частотами 10-16кГц.

Оператор персонального комп'ютера перебуває в ближній зоні (зона індукції), де ще не сформована електромагнітна хвиля, що біжить, тому цю зону можна характеризувати як електричною, так і магнітного складового електромагнітного поля.

Контроль рівнів електричного поля здійснюється за значенням напруженості електричного поля, вираженої в В/м.

Контроль рівнів магнітного поля здійснюється за значенням напруженості магнітного поля, вираженої в А/м або за значенням магнітної індукції, вираженої в Тл.

Підвищений рівень шуму. Джерелами шуму на робочому місці програміста є як самі ЕОМ, так і периферійне встаткування.

Шум - несприятливо діючі на людину звуки. Він є хаотичним сполученням звуків різної частоти й інтенсивності. Джерелом звуку в ЕОМ і периферійному встаткуванні є коливні тверді частини, до яких можна віднести системи вентиляції встаткування, дисководи, каретки й приводи принтерів. Так само джерелом високочастотних шумів може бути електронна частина ЕОМ і периферійного встаткування.

Тривалий вплив інтенсивного шуму може привести до патологічного стану слухового органа, до його стомлення й виникнення професійного захворювання - приглухуватості, тобто до втрати слуху. Шум викликає зміни у серцево-судинній системі, супроводжувані порушенням тонусу й ритму серцевих скорочень, змінюється артеріальний тиск, приводить до порушення нормальної функції

шлунка. Особливо піддається впливу центральна нервова система. Відзначається зміна органів зору, вестибулярного апарата, збільшення внутрічерепного тиску, порушення обмінних процесів організму.

4.3 Психофізіологічні небезпечні й шкідливі фактори

Статичні фізичні перевантаження. При роботі з ЕОМ застосовують клавішний ввід. Робочі цикли при роботі на клавішних апаратах, як правило, багаторазово повторюються. Велике їхнє число за робочу зміну приводить до нервово-м'язового стомлення, що може бути основним етиологічниим фактором м'язової перенапруги й виникнення професійних захворювань рук.

При статичних фізичних перевантаженнях ніг, плечей, шиї й рук довгостроково прибувають у стані скорочення. У них погіршується кровообіг. Живильні речовини надходять у м'язи недостатньо швидко, у тканинах накопичуються продукти розпаду, у результаті чого можуть виникнути хворобливі відчуття.

Оскільки кожне натискання на клавішу сполучене зі скороченням м'язів, сухожилля безупинно сковзають уздовж костей і стикаються із тканинами.

Внаслідок чого можуть виникнути запальні процеси. Розпухлі внаслідок повторюваних рухів, оболонки сухожиль можуть здавити нерв. Виникає зап'ястний синдром.

Нервовопсихічні перевантаження. Нервова перенапруга обумовлена напругою уваги. Часта й тривала перенапруга може служити джерелом ряду захворювань серцево-судинної, нервової, зорової й іншої систем організму.

Розумова перенапруга. Розумова діяльність - це діяльність, насамперед, центральної нервової системи, її вищого відділу кори головного мозку.

При розумовій роботі відбувається звуження судин кінцівок і розширення судин внутрішніх органів.

Низький рівень загального обміну при розумовій діяльності не є показником малої інтенсивності обмінних процесів, навпаки, споживання кисню збільшується

в 15-20 разів у порівнянні з фізичною роботою. Можливі значні зміни кров'яного тиску, пульсу, підвищення цукру в крові. Тривала розумова робота може привести до серцево-судинних захворювань.

Перенапруга аналізаторів. Центральна нервова система одержує інформацію від зовнішнього середовища за допомогою чутливих апаратів, що сприймають сигнали. Ці апарати академіком И.П.Павловим названі аналізаторами.

Основна характеристика аналізаторів – висока чутливість. При роботі з дисплеєм, яскравість сигналів значно перевищує мінімальний рівень світлового впливу. Верхня межа інтенсивності світлового сигналу, при якому ще не порушується робота аналізаторів, становить 10.000 кд/м^2. Але крайні значення стомлюючі для очей. Так ознакою ненормально великої яскравості зображення на сітківці є виникнення послідовних образів. Зорова система має властивість оцінювати сприйману яскравість сигналу.

Зорова робота вимагає частого перемикання з однієї поверхні на іншу, що відбувається на тлі нерівномірних яскравостей. Результати досліджень показують, що робота в умовах постійної переадаптації до яскравостей, що розрізняються приблизно в 10 разів, викликає почуття дискомфорту вже в перші години роботи, а потім й явну перевтому. Особливо несприятливі такі перепади яскравості, які викликають сліпучий ефект. Від сліпучої дії світла спочатку найбільше всього погіршується контрастна чутливість ока, а потім гострота зору. При незадовільному розподілі яскравості в освітленому просторі виникає відчуття зорового дискомфорту.

Дискомфортні умови для роботи ока можуть виникнути не тільки в результаті більших яскравостей у полі зору, але й внаслідок недостатньої освітленості поля зору. Психофізіологічні досвіди показали, що різна чутливість ока досягає максимуму при освітленості білої поверхні більше 200 лк і зберігає його аж до 3000 лк. Сталість гостроти зору протягом роботи (стійкість ясного бачення) досягає максимуму приблизно при освітленості білої поверхні більше 200 лк.

4.4 Заходи захисту від небезпечних і шкідливих виробничих факторів

Основним способом захисту від статичної електрики є заземлення периферійного встаткування, а також зволоження навколишнього повітря.

При відсутності природного освітлення використається штучне. Для загального освітлення використаються лампи денного світла, тому що їхній спектр близький до природного.

Зниження рівня шуму можна домогтися, застосовуючи демпфірування, звукоізоляцію, поглинання. Демпфірування - покриття поверхні демпферними матеріалами, що мають велике внутрішнє тертя (мастики, спеціальні види повсті, лінолеум). Звукоізоляція - зниження виробничого шуму на шляху його поширення. За допомогою звукових перегородок легко знизити рівень шуму на 30-40 дБ. Звукопоглинання - застосування звуковбирних матеріалів при обладнанні кімнати.

Для запобігання виникнення шкідливих наслідків від статичних фізичних перевантажень, необхідно обладнати місце так, щоб виключити незручні пози, тривалі напруги. Дисплей повинен бути встановлений на такій висоті й під таким кутом, щоб шия працівника не була зігнута й утримувалася в такому стані напруженими м'язами. Клавіатура повинна розташовуватися так, щоб до її не потрібно було тягтися, руки не повинні бути у висячому положенні або перенапружені. Не можна довго перебувати в одній позі. Щогодини протягом 15 хв. необхідно займатися якою-небудь справою, зробити розминку.

Для запобігання перенапруги аналізаторів необхідно визначити режим яскравості. Для цього потрібно встановити рівень яскравості, співвідношення рівнів яскравості в полі зору, рівень контрасту. Оптимальною вважається така яскравість, при якій проявляється контрастна чутливість ока, гострота зору й швидкість розрізнення сигналів. Нижньою комфортною границею рівня яскравості світних сигналів можна вважати 30 кд/м^2, верхня комфортна границя визначається значенням сліпучої яскравості. Яскравість об'єктів на екрані повинна бути погоджена з яскравістю фону екрана й навколишнім освітленням. При зворотному контрасті контраст яскравості повинен перебувати в межах 85-90% з можливістю

регулювання яскравості об'єктів, а при прямому контрасті 75-80% з можливістю регулювання яскравості фону. Прямий контраст переважніше зворотного. Оптимальне співвідношення яскравостей між екраном, його найближчим оточенням і далеким оточенням становить 5:2:1. Відстань зчитування інформації з екрана може бути від 400мм і більше.

4.5 Розрахунок виробничого освітлення

Недостатня освітленість робочої зони оператора, усувається введенням комбінованого штучного освітлення. Для розрахунку освітлення необхідно знати розміри приміщення. Для даного виду діяльності (дистанційний контроль виробничого процесу на екрані комп'ютера) може використовуватися, будь-яка доступна площа, як правило, це закрите приміщення без доступу світла. Для розрахунку візьмемо приміщення площею 10×3×3 (довжина А = 10 м, ширина В = 3 м, висота С = 3 м) з повною відсутністю природного освітлення.

Розрахуємо освітлення. Для організації освітлення робочих місць скористаємося світильниками з люмінесцентними лампами. Знайдемо висоту підвісу над робочою площиною *Н*. З рисунка 4.1, представленого нижче, видно, що висота робочої площини (стола) над підлогою дорівнює 0,85 м. Отже, з огляду на відстань від стелі до світильника (0,15 м), відстань від робочої площини до світильника буде приблизно дорівнювати 2 м.

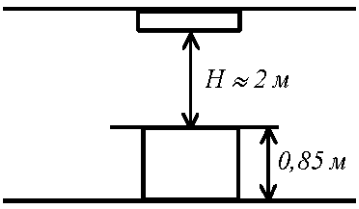


Рисунок 4.1 – Габарити приміщення

Знайдемо відстань між центрами світильників *l.* Для люмінесцентних світильників

$$l = 1{,}4H = 1{,}4 \cdot 2 = 2{,}8 \ м$$

При такому значенні l можливо реалізувати тільки один ряд світильників уздовж довгої стіни.

Відстань світильників від стін дорівнює

$$l_1 = 0{,}25l = 0{,}25 \cdot 2{,}8 = 0{,}7 \ м$$

Кількість світильників визначається за формулою

$$N = \frac{A - 2l_1}{l} + 1 = \frac{10 - 2 \cdot 0{,}7}{2{,}8} \approx 4$$

Розташування світильників умовно показане на рисунку 4.2. Крапками позначені центри світильників.

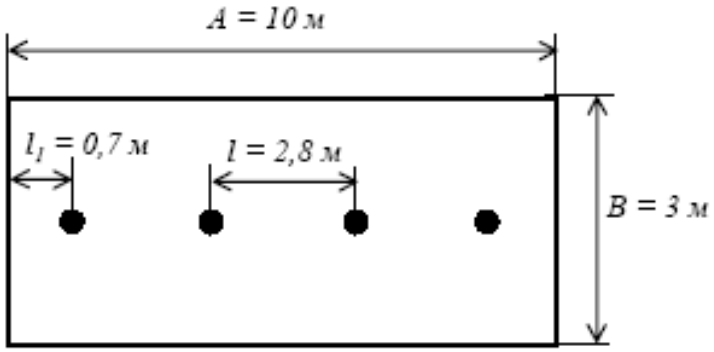


Рисунок 4.2 – Розташування світильників

Для знаходження необхідного світлового потоку одного джерела необхідно попередньо знайти деякі параметри.

Коефіцієнт запасу, що враховує старіння ламп і забруднення світильників, дорівнює

$$K_з = 1{,}5,$$

тому що в даному приміщенні виділення пилу низьке.

Площа освітлюваного приміщення

$$S = AB = 10 \cdot 3 = 30 \ м^2$$

Коефіцієнт мінімальної освітленості для люмінесцентних ламп

$$Z = 1{,}1$$

Визначимо індекс приміщення

$$i = \frac{S}{H(A+B)} = \frac{30}{2(10+3)} \approx 1{,}15$$

Коефіцієнт відбиття стелі (свіжа побілка)

$$\rho_{пт} = 0{,}7$$

Коефіцієнт відбиття стін (голубий колір)

$$\rho_{ст} = 0{,}53$$

Коефіцієнт відбиття підлоги (коричневий колір)

$$\rho_{пл} = 0{,}23 .$$

З огляду на те, що застосовується світильник ЛСПО2, даним коефіцієнтам відбиття й індексу приміщення відповідає коефіцієнт використання світлового потоку

$$\eta = 0{,}47$$

Знаючи тепер всі необхідні значення параметрів, обчислимо необхідний світловий потік одного джерела світла

$$F = \frac{E \cdot K_з \cdot S \cdot Z}{N \cdot \eta} = \frac{250 \cdot 1{,}5 \cdot 30 \cdot 1{,}1}{4 \cdot 0{,}47} \approx 6582 \ \ лм$$

Найближча по світловому потоці лампа ЛБ80-4 (6220 лм), Відхилення світлового потоку цієї лампи від розрахованого становить -5,5 %. Відхилення повинне лежати в межах (-10% - +20%), по цьому критерію лампа підходить.

4.6 Електробезпека

Для захисту від електротравм у приміщенні слід використовувати сховану, добре ізольовану електропроводку. Розподіл енергії здійснюється за допомогою розподільного щита з ізольованими кабелями і розетками, що виключають можливість короткого замикання. Розподільний щит має запобіжники, що спрацьовують при критичному режимі роботи. Персонал, що обслуговує ЕОМ, зобов'язаний пройти навчання безпечним методам роботи на робочому місці і перевірку знань правил техніки безпеки.

У мережі нейтраль джерела струму слід приєднати до заземлення за допомогою заземлюючого провідника . Цей заземлювач розташовується поблизу джерела живлення (в окремих випадках) біля стіни будинку, у якому він знаходиться. Ефективним заходом захисту в даному випадку є захисне занулення.

Захисне занулення - це навмисне електричне з'єднання з нульовим захисним провідником металевих неструмоведучих частин, що можуть виявитися під напругою (ГОСТ 12.1.009-76). Захисна дія занулення здійснюється тим, що при замиканні однієї з фаз на занулений корпус у колі цієї фази виникає струм короткого замикання, що впливає на струмовий захист (плавкий запобіжник, автомат), у результаті чого відбувається відключення аварійної ділянки від кола. Таким чином, занулення зменшує напругу дотику й обмежує час, протягом якого людина, торкнувшись до корпуса, може потрапити під дію напруги.

ГОСТ 12.1.038-82 установлює гранично припустимі рівні напруг дотику (В), і струмів (мА), що протікають через тіло людини, призначені для проектування способів і засобів захисту людей при взаємодії з електроустановками виробничого і побутового призначення постійного і змінного струму частотою 50 і 400 Гц.

На підставі ПУЭ-85 дане приміщення по ступені небезпеки поразки електричним струмом відноситься до класу приміщень без підвищеної небезпеки поразки електричним струмом, так як умови, що створюють підвищену небезпеку поразки електричним струмом (вологість, струмоведучий пил, висока температура, можливість одночасного торкання до струмоведучих частин і заземлення) відсутні .

Електропроводка в приміщенні схованого типу, тому випадкове торкання проводів з напругою 220 В виключено, за умови дотримання правил техніки безпеки. Вимикачі штучного освітлення ізольовані струмонепровідним облицюванням.

Поразка електричним струмом може відбутися в результаті несправних розеток і вилок ЕОМ, а також пристроїв місцевого освітлення, короткого замикання.

4.7 Пожежна безпека

При гасінні пожеж найбільше поширення одержали наступні принципи припинення горіння:

- ізоляція вогнища горіння від повітря чи зниження шляхом розведення повітря непальними газами концентрації кисню до значення, при якому не може відбуватися горіння;
- охолодження вогнища горіння нижче визначених температур;
- інтенсивне гальмування швидкості хімічної реакції в полум'ї;
- механічний зрив полум'я в результаті впливу на нього сильного струменя газу чи води;
- створення умов вогнеперешкодження, тобто таких умов, при яких полум'я поширюється через вузькі канали

Приміщення, де виробляється монтаж друкованих плат відноситься по пожежній безпеці до категорії В по ОНТП 24-86 і зоні П-І по ПУЭ.

Категорія В – пожежнонебезпечні; до цієї категорії відносяться приміщення, у яких застосовуються рідини з температурою спалаху вище 61^{O}С і горючі пили чи волокна, нижня межа запалення яких більш 65 г/м^{3}, тверді спаленні речовини і матеріали, здатні тільки горіти, але не вибухати при контакті з повітрям, чи водою один з одним.

Пожежа при монтажі може виникнути в результаті короткого замикання. Джерелами запалювання можуть стати джерела місцевого освітлення, а також розігрітий паяльник у результаті наявності пальних речовин, таких як спиртобензинова суміш, ацетон. Причини виникнення пожежі наступні:

- порушення режимних вимог;
- несправність і неправильна експлуатація електропаяльників і пристроїв місцевого освітлення;
- порушення працюючими технологічних інструкцій.

Весь обслуговуючий персонал проходить періодичний інструктаж з техніки

безпеки. На випадок пожежі в приміщенні обов'язкова наявність запасних евакуаційних виходів. У таких приміщеннях не можна застосовувати для гасіння пожежі воду, так як вода має значну електропровідність. У цьому випадку застосовують вуглекислі вогнегасники ОУ-8. У якості пожежних оповіщувачів використовуються теплові (ДТЛ, ДПС-ОЗ8 і ін.). для оповіщення про пожежу в приміщенні мається телефон загального користування і табличка з номерами телефонів.

У коридорі установлюється внутрішній пожежний кран для гасіння пожежі за допомогою води.

У даному розділі дипломного проекту проведений аналіз необхідних умов для роботи оператора, і фактори, що діють на нього в процесі роботи при максимально невідповідних умовах праці, а також рекомендації до усунення або зменшення небезпечних і шкідливих виробничих факторів. Приводяться рекомендації зі зменшення пожежонебезпеки приміщення.

ВИСНОВКИ

В результаті написання дипломного проекту виконано ряд завдань:

1. Виконано аналіз поточного стану розвитку методів інтелектуального аналізу даних та обґрунтована ефективність застосування апарату байєсівських мереж (БМ).

2. Для побудови структури БМ запропоновано евристичний алгоритм навчання лінійної складності за статистичними даними. Для визначення міри зв'язку між вершинами алгоритм використовує значення взаємної інформації, а в якості функції оцінювання структури моделі – опис мінімальною довжиною (ОМД).

3. Розроблено алгоритм побудови точного ймовірнісного висновку в БМ за навчальними даними. Для обчислення значень ймовірностей станів вершин замість таблиць умовних ймовірностей використана матриця емпіричних значень спільного розподілу ймовірностей всієї мережі. Головними перевагами алгоритму є залежність швидкості роботи тільки від розміру навчальної вибірки, відсутність потреби в попередньому перетворенні структури БМ та простота реалізації.

4. Розроблена і програмно реалізована оригінальна система підтримки прийняття рішень для інтелектуального аналізу даних на основі БМ, яка грунтується на запропонованих алгоритмах побудови структури та ймовірнісного висновку. СППР впроваджена у Тернопільському відділенні ПАТ «Укрінбанк», що дало можливість побудувати ефективні прогнозуючі моделі для підтримки прийняття рішень з метою оцінювання ризиків при кредитуванні фізичних осіб.

5. Проведений аналіз необхідних умов для роботи оператора, і фактори, що діють на нього в процесі роботи при максимально невідповідних умовах праці, а також рекомендації до усунення або зменшення небезпечних і шкідливих виробничих факторів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бидюк П. И. Модели оценки рисков кредитования физических лиц / П. И. Бидюк, Е. А. Матрос // Кибернетика и вычислительная техника. – К.: Академпериодика, 2007. – 153. – С. 87-95.
2. Hunt E. B., Marin J., Stone P. Experiments in Induction by E. B. // The American Journal of Psychology. – N.Y.: Academic Press, 1966. – Vol. 80, № 4. – P. 651-653.
3. Sneath P.H.A., Sokal R.R. Numerical taxonomy – the principles and practice of numerical classification. – SF.: Freeman, 1973. – 573 p.
4. Zhang T., Ramakrishnan R., Livny M. BIRCH: an efficient data clustering method for very large databases / Proceedings of the 25-th International Conference on Management of Data, Montreal, Canada, 4-6 June, 1996. – NY.: ACM, 1996. – Vol. 25, №2. – P. 103-114.
5. Хайкин С. Нейронные сети: полный курс / С. Чайкин. – М.: Издательский дом Вильямс, 2008. – 1103 с.
6. Терехов С. А. Вероятностное моделирование в байесовых сетях / С. А. Терехов // Лекция для школы-семинара “Современные проблемы нейроинформатики”, 29-31 января 2003 г. – М.: МИФИ, 2003 г. – 63 с.
7. Bayes T. (1763). An Essay towards solving a Problem in the Doctrine of Chances. By the late Rev. Mr. Bayes, F.R.S., communicated by Mr. Price, in a letter to John Canton, A.M. and F.R.S. // Philosophical Transactions of the Royal Society of London. – 1763. – 53. – P. 370–418.
8. Cheng J., Greiner R., Kelly J., Bell D.A. and Liu W. Learning Bayesian networks from data: an information-theory based approach // The artificial intelligence journal (AIJ). – 2002. – 137. – P. 43-90.
9. Chow C.K., Liu C.N. Approximating discrete probability distributions with dependence trees // IEE Transactions on information theory, May 1968. – Vol. IT-14, №3. – P. 462 – 467.
10. Liu R. F. and Soetjipto R. Learning on Bayesian networks / Report for class

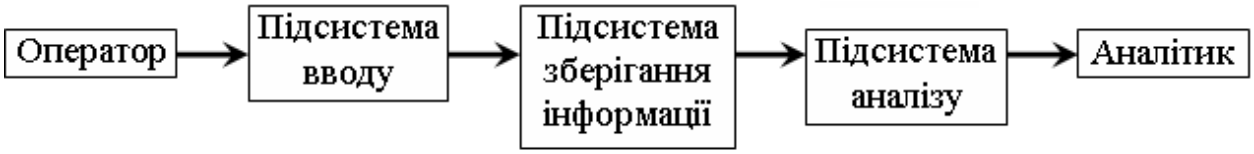
project in course MIT 6.825: techniques in artificial intelligence, the MIT computer science and artificial intelligence laboratory, Massachusetts Institute of Technology, Cambridge, December 2004. – 39 p.

11. Buntine W. Operations for learning with graphical models // Journal of artificial intelligence research (JAIR). – Menlo Park: AAAI Press, 1994. – 2. – P. 159-225.
12. Verma T. and Pearl J. Equivalence and synthesis of causal models / Proceedings of the sixth international conference on Uncertainty in Artificial Intelligence (UAI'90), Cambridge, Massachusetts, USA, 27–29 July, 1990. – NY.: Elsevier science, 1991. – P. 255-270.
13. Dempster A.P., Laird N.M. and Rubin D. B. Maximum likelihood from incomplete data via the EM algorithm // Journal of the Royal Statistical Society. – 1977. – Vol. 39, №1. – P. 1-38.
14. Dechter R. Bucket elimination: a unifying framework for reasoning // ACM Press, December 1996. –Vol. 28, № 61. – P. 1-51.
15. Васнев С.А. Статистика: учебное пособие / С.А. Васнев – М.: МГУП. – 2001. – 170 с.
16. Терентьев А. Н. Эвристический метод построения байесовских сетей / А.Н. Терентьев, П.И. Бидюк // Математические машины и системы. – К.: ТОВ "РВПК ЕКСЛИБРИС", 2006. – № 3. – С. 12-23.
17. Шумский С.А. Байесова регуляризация обучения / С.А. Шумский // Лекции по нейроинформатике, часть 2. – М.: МИФИ, 2002. – 172 с.
18. Grunwald P. A Tutorial Introduction to the Minimum Description Length Principle. // Advances in minimum description length: theory and applications. MIT Press, Cambridge, MA, USA. – 2005. – 80 p.
19. Haettenschwiler P. Neues anwenderfreundliches Konzept der Entscheidungs-unterstutzung. Gutes Entscheiden in Wirtschaft, Politik und Gesellschaft. Zurich: Hochschulverlag AG, 1999. — S. 189—208.
20. Power D. J. «What is a DSS?» // The On-Line Executive Journal for Data-Intensive Decision Support, 1997. — Vol. 1. — N3.
21. Бідюк П.І. Проектування систем підтримки прийняття рішень (навчальний

посібник) / П. І. Бідюк, Є. О. Демківський. – К.: КНУТД, 2005. – 156 с.
22. Даценко ГЛ., Габович Р.Д., Йонда М.Є. Умови праці, з комп'ютером і їх оптимізація.- Львів: ЛДМУ, 1998. - 46с.
23. Жидецький В.Ц., Джигирей В.С, Мельников О.В. Основи охорони праці.- Львів: Афіша, 1999.-348с.
24. Методичні рекомендації до виконання дипломного проекту з освітньо-кваліфікаційного рівня “Спеціаліст”. Спеціальність «Комп’ютерні системи та мережі» / О.М. Березький, Р.Б.Трембач, Г.М. Мельник / Під ред. О.М. Березького. – Тернопіль: ТНЕУ, 2012. – 40 с.

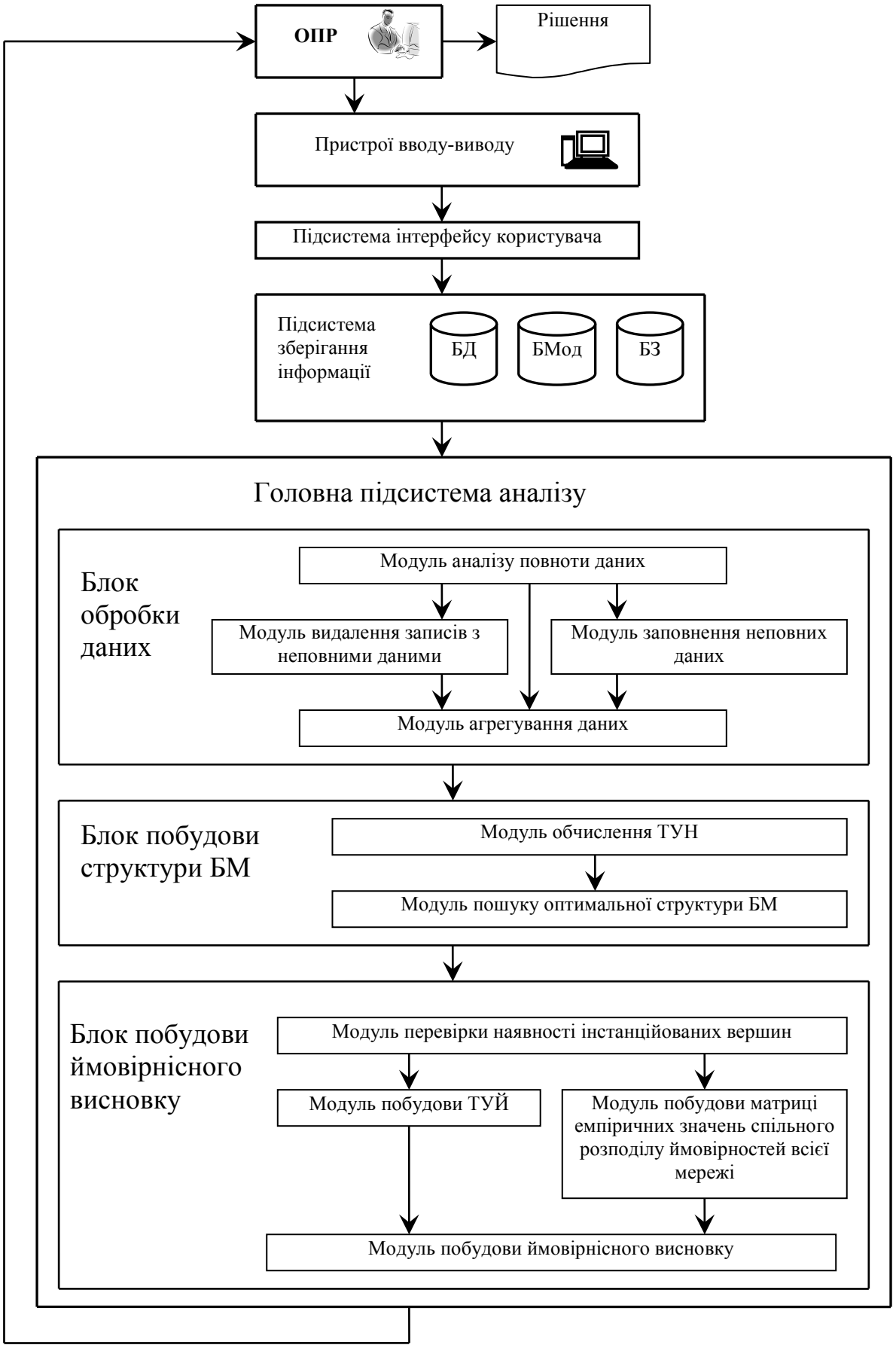
Додаток А

СППР. Схема структурна



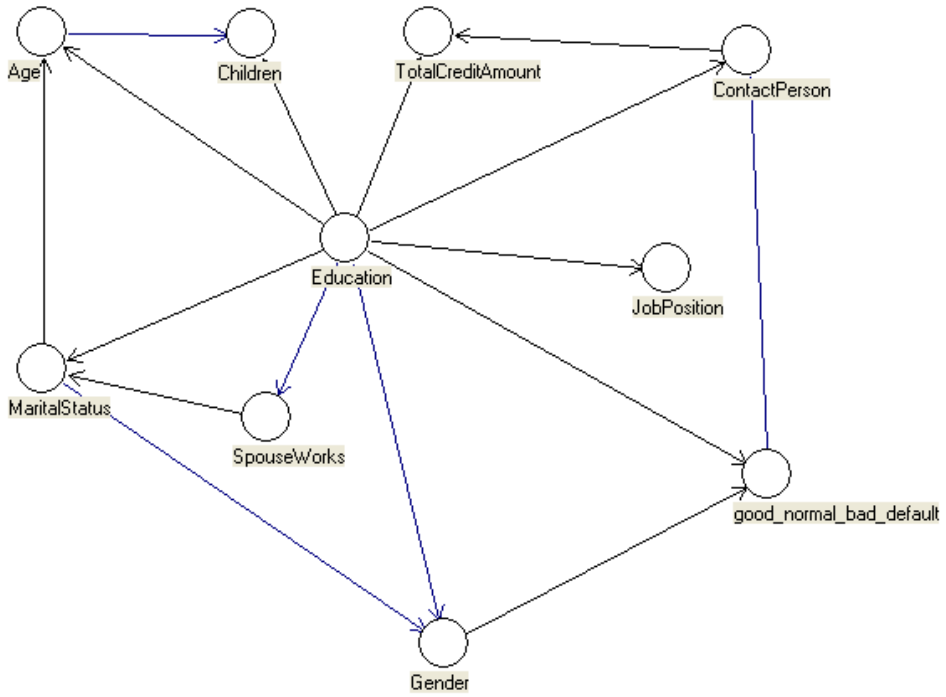
Додаток Б

Архітектура СППР. Схема структурна



Додаток В

БМ. Схема взяємозв’язків функціональних елементів



Додаток Г

Код програми системи підтримки прийняття рішень для інтелектуального аналізу даних на основі байєсівських мереж

В.1 Процедура обчислення оцінки опису мінімальної довжини

```
Procedure heuristic_part_svModern_m2;
Var
   CPT : array of array of Integer;
  // в матриці CPT зберігаються залежності вершин одна від одної.
   List_parent : array of Integer;
   AFull_List_parent : array of array of String;
   i, j, i1, j1, i2, j2, k, k2, r : LongInt;
   L_optimal, model_entropy, sum, denominator, numerator, L : Real;
   List_of_Model : array of Real;
   List_model_optimal : array of Integer;
   // List_model_optimal – список оптимальних моделей, тобто моделей які описуються мінімальною
   // довжиною.
   count_model_optimal : Integer;
   // count_model_optimal – кількість оптимальних моделей
   count_parent, count, number_parent, number_child : Integer;
   rc1rcn, coef, kg, model_kg   : Integer;
   string_value : LongInt;
   // VFullC_column    - кількість стовбців у файлі, який зберігає нециклічні моделі
   // VFullC_row       - кількість строк у файлі, який зберігає нециклічні моделі.

Begin
  count_model_optimal:=0;
  L_optimal:=1.7*power(10,308);
  SetLength(List_of_Model,High(VFullC)+1);
  for i2:=Low(VFullC) to High(VFullC) do
    Begin
      SetLength(CPT,number_variable,number_variable);
      for j2:=Low(VFullC[i2]) to High(VFullC[i2]) do
        Begin
          i1:=V[1,j2];
          j1:=V[2,j2];
          if VFullC[i2,j2]=-1 then  CPT[j1,i1]:=1;
          if VFullC[i2,j2]=1  then  CPT[i1,j1]:=1;
        End;
      model_entropy:=0;
      model_kg:=0;
      for j2:=Low(CPT) to High(CPT) do
        Begin
          count_parent:=0;
          try
          except
          finalize(List_parent);
          end;
          for k2:=Low(CPT[j2]) to High(CPT[j2]) do
            Begin
              if j2<>k2 then // ця умова дозволяє нам не враховувати діагональні елементи
                Begin
                  if CPT[k2,j2]=1 then
                    Begin
                      count_parent:=count_parent+1;
                      SetLength(List_parent,count_parent);
```

```
              List_parent[count_parent-1]:=k2;
             End;
           End;
         End;
        count_parent:=count_parent+1;
        SetLength(List_parent,count_parent);
        List_parent[count_parent-1]:=j2; // в кінці списку вказуємо дочерний вузол

        // Тепер треба розрахувати емпіричну ентропію j-го елемента i-ї моделі
        // В List_parent зберігаються пращури, а останнім елементом зберігається дочерний вузлел
        // List_parent=[1-й пращур, 2-й пращур, ... , к-й пращур, вузол-спадкоємець]
        // к = count_parent, тобто в List_parent зберігається count_parent+1 елементів.
        // В матриці AFull_List_parent  зберігаються всі можливі конфігурації мережі.
        // Вона буде зберігати rc1rcn=rc1*rc2*...*rcn строк. rc1rcn розраховується в наступному циклі.
        rc1rcn:=1;
         for i:=Low(List_parent) to High(List_parent) do
          Begin
           j:=List_parent[i];
           rc1rcn:=rc1rcn*(High(General_Unit.General_Form.Matrix_Description[j])-1);
          End;
        coef:=rc1rcn;
        try
        except
        finalize(AFull_List_parent);
        end;
        SetLength(AFull_List_parent, rc1rcn, High(List_parent)+1);
        for k:=Low(List_parent) to High(List_parent) do
         Begin
          i:=List_parent[k];
          coef:=floor(coef/(High(General_Unit.General_Form.Matrix_Description[i])-1));
          r:=0;
          for j:=Low(AFull_List_parent) to High(AFull_List_parent) do
           Begin
            if (j mod coef) =0 then r:=r+1;
            if r>(High(General_Unit.General_Form.Matrix_Description[i])-1)  then r:=1;
            AFull_List_parent[j,k]:=General_Unit.General_Form.Matrix_Description[i,r];
           End;
         End;
        // Всі можливі конфігурації зберігаються в матриці AFull_List_parent
        // В наступному циклі застосовуючи AFull_List_parent розраховується порядок змінних, який
        // зберігається в AFull_List_parent в стовпці count_parent+2.
        SetLength(AFull_List_parent, rc1rcn, High(List_parent)+2);
        for i:=Low(AFull_List_parent) to High(AFull_List_parent) do
         Begin
          string_value:=0;
          for j:=Low(AFullC)+1 to High(AFullC) do
           Begin
            count:=0;
            for k:=Low(List_parent) to High(List_parent) do
             Begin
              number_parent:=List_parent[k];
              if AFull_List_parent[i,k]=AFullC[j,number_parent] then count:=count+1;
             End;
            if count=High(List_parent)+1 then
string_value:=string_value+Round(AFullC_Record_Count[j]*(Loader_Unit.Loader_Form.SG_Data.RowCount-1));
           End;
          AFull_List_parent[i,High(List_parent)+1]:=IntToStr(string_value);
         end;
        i:=0;
        sum:=0;
        number_child:=List_parent[High(List_parent)];
        while i<High(AFull_List_parent)+1 do
         Begin
```

```
          denominator:=0;
          for j:=1 to (High(General_Unit.General_Form.Matrix_Description[number_child])-1) do
            Begin
              i:=i+1;
              denominator:=denominator+StrToFloat(AFull_List_parent[i-1,High(AFull_List_parent[i-1])]);
            End;
          i:=i-(High(General_Unit.General_Form.Matrix_Description[number_child])-1);
          for j:=1 to (High(General_Unit.General_Form.Matrix_Description[number_child])-1) do
            Begin
              i:=i+1;
              numerator:=StrToFloat(AFull_List_parent[i-1,High(AFull_List_parent[i-1])]);
              if (numerator<>0) And (denominator<>0) then
                Begin
                  sum:=sum-numerator*logN(exponenta,(numerator/denominator));
                  // LogN – розрахунок логарифма X по базі N
                End;
            End;
        End;
      model_entropy:=model_entropy+sum;
      kg:=1;
      for i:=Low(List_parent) to (High(List_parent)-1) do
        Begin
          j:=List_parent[i];
          kg:=kg*(High(General_Unit.General_Form.Matrix_Description[j])-1);
        End;
      j:=List_parent[High(List_parent)];
      kg:=kg*((High(General_Unit.General_Form.Matrix_Description[j])-1)-1);
      // kg – кількість станів вузла № j2 в моделі № i2
      model_kg:=model_kg+kg;
      Finalize(AFull_List_parent);
      Finalize(List_parent);
      End;
    // model_entropy – імперічна ентропія вузла моделі № i2
    // model_kg, – кількість умовних імовірностей моделі № i2
    L:=model_entropy+((model_kg/2)*logN(exponenta,MatrixOfData_row));
    List_of_Model[i2]:=L;
    // List_model_optimal - список оптимальних моделей, тобто моделей які описуються мінімальною
    // довжиною
    // count_model_optimal – кількість оптимальних моделей
    if abs(L_optimal-L)<min_norma then
      Begin
        count_model_optimal:=count_model_optimal+1;
        SetLength(List_model_optimal, count_model_optimal);
        List_model_optimal[count_model_optimal-1]:=i2;
      End
    Else
      Begin
        if L_optimal>L then
          Begin
            L_optimal:=L;
            count_model_optimal:=1;
            SetLength(List_model_optimal, count_model_optimal);
            List_model_optimal[count_model_optimal-1]:=i2;
          End;
      End;
    Finalize(CPT);
  End;
  try
    Finalize(List_OM);
  except
  end;
  if High(List_model_optimal)<MathUnitBNT1.max_size_list_om then
    Begin
```

```
      SetLength(List_OM,High(List_model_optimal)+1,VFullC_column);
      // Формируем List_MO
      for k2:=Low(List_model_optimal) to High(List_model_optimal) do
        Begin
          for j2:=0 to VFullC_column-1 do
            Begin
              List_OM[k2,j2]:=VFullC[List_model_optimal[k2],j2];
            End;
        End;
    End
  Else
    Begin
      SetLength(List_OM,max_size_list_om,VFullC_column);
      // Формуємо матрицю List_MO
      for k2:=Low(List_model_optimal) to (max_size_list_om-1) do
        Begin
          for j2:=0 to VFullC_column-1 do
            Begin
              List_OM[k2,j2]:=VFullC[List_model_optimal[k2],j2];
            End;
        End;
    End;
  if MathUnitBNT1.flag_finish_estimation = true then
    Begin
      for k2:=Low(List_OM) to High(List_OM) do
        Begin
          if k2<1 then
            Begin
              SetLength(CPT,number_variable,number_variable);
              i2:=List_model_optimal[k2];
              for j2:=0 to VFullC_column do
                Begin
                  i1:=V[1,j2];
                  j1:=V[2,j2];
                  if List_OM[k2,j2]=-1 then CPT[j1,i1]:=1;
                  if List_OM[k2,j2]=1  then CPT[i1,j1]:=1;
                End;
              try
                Finalize(CPT);
              except
              end;
            End;
        End;
  End;
  Finalize(List_model_optimal);
  Finalize(List_of_Model);
End;
```

В.2 Процедура побудови імовірнісного висновку в мережі Байєса

```
procedure TInference_Form.N4Click(Sender: TObject);
  Var i,j, i2, j2, i3, j3, i4, j4, i5, wid, table_num, i_pos : integer;
    Temp, Temp_A : TComponent;
    Array_RowSelected : array of array of string;
    Array_RowSelected_Count : integer;
    Array_ListParentNode : array of string;
    Array_Node_Index_Count : integer;
    Array_Node_Index : array of array of string;
    NameNode, Name_C : String;
    f_exit, fj_exit, f_present_selected : boolean;
    pS : PString;
```

```
    pR : PReal;
    Node_State_sum : Real;
    st_col_node, one_to_one_correspondence, one_to_one, count_flag_computation, count_parent_equal : integer;
    buf_stt, Node_State : string;
    test_i : integer;
    test_s, state_value_parent_node : string;
    i6, j6, i7, j7, displacement, i3_table_index, i_state_table : integer;
    probability_ij, probability_state_value_parent_node, Sum_probability_ij, test_f : real;

begin
  // аналіз інстанціірованних значень вузлів
  Array_RowSelected_Count:=0;
  f_present_selected:=false;
  one_to_one:=0;
  for i := 0 to Inference_Form.Panel.ComponentCount - 1 do
    Begin
      if Inference_Form.Panel.Components[i].ClassNameIs('TMyNodeTable') then
        Begin
          Temp := Inference_Form.Panel.Components[i];
          inc(Array_RowSelected_Count);
          SetLength(Array_RowSelected,Array_RowSelected_Count,3);
          Array_RowSelected[Array_RowSelected_Count-1,0]:=(Temp as TMyNodeTable).Cells[0,0];
          if (Temp as TMyNodeTable).RowSelected<>0
            then
              Begin
                Array_RowSelected[Array_RowSelected_Count-1,2]:=(Temp as TMyNodeTable).Cells[0,(Temp as
                                                                                    TMyNodeTable).RowSelected];
                f_present_selected:=true;
                // рахується кількість RowSelection
                one_to_one:=one_to_one+1;
              End
            else
              Begin
                Array_RowSelected[Array_RowSelected_Count-1,2]:='';
              End;
          // знаходиться номер стовпчика зв'язаний з вузлом, який
          buf_stt:=(Temp as TMyNodeTable).Name;
          delete(buf_stt,1,Length('Node'));
          Array_RowSelected[Array_RowSelected_Count-1,1]:=buf_stt;
        End;
    End;
        if not(f_present_selected) // випадок коли інстанційовані значення відсутні
          then
            Begin
              // формування матриці індексов Array_Node_Index
              Array_Node_Index_Count:=0;
              for i := 0 to Inference_Form.Panel.ComponentCount - 1 do
              // аналіз всіх стовпчиків
                Begin
                  if Inference_Form.Panel.Components[i].ClassNameIs('TMyNodeTable')
                   then
                    Begin
                      Temp := Inference_Form.Panel.Components[i];
                      inc(Array_Node_Index_Count);
                      SetLength(Array_Node_Index, Array_Node_Index_Count, 4);
                      // ім'я таблиці
                      Array_Node_Index[Array_Node_Index_Count-1,0]:=(Temp as TMyNodeTable).Name;
                      // індекс таблиці серед Inference_Form.Panel.Components
                      Array_Node_Index[Array_Node_Index_Count-1,1]:=IntToStr(i);
                      // індекс вузла в General_Form зв'язаний з таблицею
                      j:=0;
                      fj_exit:=false;
                      while (j<General_Form.Panel.ComponentCount) and (fj_exit=false) do
```

```
          Begin
            if General_Form.Panel.Components[j].ClassNameIs('TMyShape') then
              Begin
                if ((General_Form.Panel.Components[j] as TMyShape).Name) = ((Temp as
                                                                    TMyNodeTable).Name) then
                  Begin
                    fj_exit:=true;
                    Array_Node_Index[Array_Node_Index_Count-1,2]:=IntToStr(j);
                  End;
              End;
            inc(j);
          End;
        Array_Node_Index[Array_Node_Index_Count-1,3]:='0';
      End;
  End;
// матриця індексів Array_Node_Index сформована
count_flag_computation:=0;
while count_flag_computation < High(Array_Node_Index) do
  Begin
  for i:=Low(Array_Node_Index) to High(Array_Node_Index) do
  // аналіз всіх елементів Array_Node_Index
    Begin
      if Array_Node_Index[i,3] = '0' then
      // = 0, тобто розрахунку не було
        Begin
          i2:=StrToInt(Array_Node_Index[i,2]);
          if (General_Form.Panel.Components[i2] as TMyShape).ListParentNode.Count = 0 then
          // тобто у вершини відсутні пращури
            Begin
              Array_Node_Index[i,3] := '1';
              inc(count_flag_computation);
              // записуються маргінальні імовірності
              // Array_Node_Index[i,1] - індекс таблиці
              // маргінальні значення
              for i5:=0 to (General_Form.Panel.Components[i2] as TMyShape).ListStatesValue.Count-1 do
                Begin
                  New(pR);
                  pR:=(General_Form.Panel.Components[i2] as TMyShape).ListStatesValue[i5];
                  (Inference_Form.Panel.Components[StrToInt(Array_Node_Index[i,1])] as
                                                          TMyNodeTable).Cells[1,i5+1]:=FloatToStr(pR^);
                End;
            End;
        End;
      if Array_Node_Index[i,3] = '0' then
      //  = 0, тобто розрахунку не було
        Begin
          count_parent_equal:=0;
          // випадок коли вузол має пращурів, але вони ще не розраховані
          i2:=StrToInt(Array_Node_Index[i,2]);
          if (General_Form.Panel.Components[i2] as TMyShape).ListParentNode.Count > 0 then
          // тобто пращурі маються, але їх треба
            Begin
              for j2:=0 to (General_Form.Panel.Components[i2] as TMyShape).ListParentNode.Count-1 do
              // аналіз пращурів вузла
                Begin
                  pS:=(General_Form.Panel.Components[i2] as TMyShape).ListParentNode[j2];
                  test_i := pos('Node',pS^);
                  test_s:=copy(pS^, test_i+Length('Node'),Length(pS^));
                  i3:=StrToInt(test_s);
                  if ( pS^ = Array_Node_Index[i3,0]) And (Array_Node_Index[i3,3] = '1') then
                    Begin
                      inc(count_parent_equal);
                    End;
```

```
                    End;
                  // умова того що всі пращури розраховані
                  // і можна розраховувати значення ймовірностей аналізуємого вузла
                  if count_parent_equal = (General_Form.Panel.Components[i2] as
                                                          TMyShape).ListParentNode.Count then
                    Begin
                     displacement:=(General_Form.Panel.Components[i2] as TMyShape).ListParentNode.Count;
                     Array_Node_Index[i,3] := '1';
                     inc(count_flag_computation);
                     for i6:=0 to (General_Form.Panel.Components[i2] as TMyShape).ListStates.Count-1 do
                      Begin
                       // розрахунок значень ймовірностей i6 - го стану вузла
                       Sum_probability_ij:=0;
                       for j6:=2 to (General_Form.Panel.Components[i2] as TMyShape).JPT_row-1 do
                        Begin
                          probability_ij:=StrToFloat((General_Form.Panel.Components[i2] as
                                                          TMyShape).JPT_Table[j6,displacement+i6]);
                         for j2:=0 to (General_Form.Panel.Components[i2] as TMyShape).ListParentNode.Count-1 do
                            Begin
                              // i6 - номер строки в JPT_Table, а j2 - номер стовпчика
                              state_value_parent_node:=(General_Form.Panel.Components[i2] as
                                                          TMyShape).JPT_Table[j6,j2];
                              pS:=(General_Form.Panel.Components[i2] as TMyShape).ListParentNode[j2];
                              test_i := pos('Node',pS^);
                              test_s:=copy(pS^, test_i+Length('Node'),Length(pS^));
                              i3:=StrToInt(test_s);
                              i3_table_index:=StrToInt(Array_Node_Index[i3,1]);
                              i_state_table:=1;
                              while  i_state_table<(Inference_Form.Panel.Components[i3_table_index] as
                                                          TMyNodeTable).RowCount do
                                Begin
                                  if (Inference_Form.Panel.Components[i3_table_index] as
                                      TMyNodeTable).Cells[0,i_state_table]=state_value_parent_node then
probability_state_value_parent_node:=StrToFloat((Inference_Form.Panel.Components[i3_table_index] as
                                                          TMyNodeTable).Cells[1,i_state_table]);
                                    End;
                                  inc(i_state_table);
                                End;
                              probability_ij:=probability_ij*probability_state_value_parent_node;
                            End;
                          Sum_probability_ij:=Sum_probability_ij+probability_ij;
                        End;
                       // додавання розрахованого значення імовірності в таблицю
                       Sum_probability_ij:=Round(Sum_probability_ij*10000)/10000;
                       if (Sum_probability_ij>=0) And (Sum_probability_ij<=1) then
                        Begin
                          (Inference_Form.Panel.Components[StrToInt(Array_Node_Index[i,1])] as
                                  TMyNodeTable).Cells[1,i6+1]:=FloatToStr(Sum_probability_ij);
                        End
                       Else
                        Begin
                          (Inference_Form.Panel.Components[StrToInt(Array_Node_Index[i,1])] as
                                  TMyNodeTable).Cells[1,i6+1]:='NaN';
                        End;
                      End;
                    End;
                  End;
                End;
              End;
            End;
           Finalize(Array_Node_Index);
         End
    else // тобто f_present_selected = true  випадок коли присутні інстанційовані значення
```

```
      Begin
        for i := 0 to Inference_Form.Panel.ComponentCount - 1 do
        Begin
          if Inference_Form.Panel.Components[i].ClassNameIs('TMyNodeTable') then
            Begin
              Temp := Inference_Form.Panel.Components[i];
              if (Temp as TMyNodeTable).RowSelected = 0 then
                Begin
                  for i2:=1 to (Temp as TMyNodeTable).RowCount-1 do
                    Begin
                      Node_State:=(Temp as TMyNodeTable).Cells[0,i2];
                      Node_State_sum:=0;
                      buf_stt:=(Temp as TMyNodeTable).Name;
                      delete(buf_stt,1,Length('Node'));
                      j3:=StrToInt(buf_stt);
                      for i3:=1 to Temp_Form.SG.RowCount do
                        Begin
                          if Temp_Form.SG.Cells[j3,i3]=Node_State then
                            Begin
                              one_to_one_correspondence:=0;
                              for i4:=Low(Array_RowSelected) to High(Array_RowSelected) do
                                Begin
                                  if Array_RowSelected[i4,2]<>'' then
                                    Begin
                                      j4:=StrToInt(Array_RowSelected[i4,1]); // номер столбца в PT
                                      if Array_RowSelected[i4,2] = Temp_Form.SG.Cells[j4,i3] then
                                        Begin
                                          inc(one_to_one_correspondence);
                                        End;
                                    End;
                                End;
                              if one_to_one_correspondence = one_to_one then
                                Begin
Node_State_sum:=Node_State_sum+StrToFloat(Temp_Form.SG.Cells[Temp_Form.SG.ColCount-1,i3]);
                                End;
                            End;
                        End;
                      (Temp as TMyNodeTable).Cells[1,i2]:=FloatToStr(Node_State_sum);
                    End;
                  Node_State_sum:=0;
                  for i2:=1 to (Temp as TMyNodeTable).RowCount-1 do
                    Begin
                      Node_State_sum:=Node_State_sum+StrToFloat((Temp as TMyNodeTable).Cells[1,i2]);
                    End;
                  for i2:=1 to (Temp as TMyNodeTable).RowCount-1 do
                    Begin
                      test_f:=StrToFloat((Temp as TMyNodeTable).Cells[1,i2])/Node_State_sum;
                      test_f:=Round(test_f*10000)/10000;
                      if (test_f>=0) And (test_f<=1) then
                        Begin
                          (Temp as TMyNodeTable).Cells[1,i2]:=FloatToStr(test_f);
                        End
                      else
                        Begin
                          (Temp as TMyNodeTable).Cells[1,i2]:='NaN';
                        End;
                    End;
                End;
            End;
        End;
    try Finalize(Array_RowSelected) except; end;
  End;
```

Додаток Д

Структура системи кредитного скорингу у вигляді БМ

Додаток Е

Довідка про використання