МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Державний вищий навчальний заклад
“Національний гірничий університет”

КОНСПЕКТ ЛЕКЦІЙ

з дисципліни «Інтелектуальний аналіз даних»

для студентів спеціальності 8.04030301
«Системний аналіз і управління»

Дніпропетровськ, 2014 р.

ВСТУП

Розвиток методів запису і зберігання даних привів до бурхливого зростання об'ємів збираної і аналізованої інформації. Об'єми даних настільки значні, що людині просто не під силу проаналізувати їх самостійно, хоча необхідність проведення такого аналізу цілком очевидна, адже в цих "сирих даних" укладені знання, які можуть бути використаний при ухваленні рішень.

Для того, щоб провести автоматичний аналіз даних, використовується Data Mining (здобич (витягання) знань). Це нова технологія інтелектуального аналізу даних з метою виявлення прихованих закономірностей у вигляді значущих особливостей, кореляцій, тенденцій і шаблонів. Сучасні системи здобичі даних використовують засновані на методах штучного інтелекту засоби уявлення і інтерпретації, що і дозволяє знаходити розчинену в терабайтних сховищах не очевидну, але вельми цінну інформацію. Фактично, ми говоримо про те, що в процесі Data mining система не відштовхується від наперед висунутих гіпотез, а пропонує їх сама на основі аналізу.

Існує безліч визначень Data Mining, але в цілому вони співпадають у виділенні чотирьох основних ознак. Згідно визначенню, Г. Пиатецкого-Шаниро (G. Pia-tetsky Shapiro,GTE Labs), одного з ведучих світових експертів в даній області, Data Mining — дослідження і виявлення алгоритмами, засобами штучного інтелекту в "сирих даних" прихованих структур, шаблонів або залежності, яка:

- раніше не були відомі;
- нетривіальні;
- практично корисні;
- доступні для інтерпретації людиною і необхідні для ухвалення рішень в різних сферах діяльності.

Специфіка сучасних вимог до продуктивної переробки інформації наступна:

- дані мають необмежений обсяг;
- дані є різнорідними (кількісними, якісними, текстовими);
- результати повинні бути конкретний і зрозумілий;
- інструменти для обробки "сирих даних" повинні бути прості у використовуванні.

Традиційна математична статистика, що довгий час претендувала на роль основного інструменту аналізу даних, не відповідала виниклим проблемам. Головна причина – концепція усереднювання по вибірці, що приводить до операцій над фіктивними величинами. Методи математичної статистики виявилися корисними головним чином для перевірки наперед сформульованих гіпотез і для "грубого розвідувального аналізу", що становить основу оперативної аналітичної обробки даних OLAP.

В основу сучасної технології Data Mining встановлена концепція шаблонів (pattern), що відображають фрагменти багатоаспектних взаємостосунків в даних. Цими шаблонами є закономірності, властиві підвибіркам даних, які можуть бути компактно виражені у формі, зрозумілій людині. Пошук шаблонів

проводиться методами, не обмеженими рамками апріорних припущень про структуру вибірки і вид розподілів значень аналізованих показників. Причини популярності Data Mining:

• стрімке накопичення даних (рахунок йде вже на экзабайти);

• загальна комп'ютеризація бізнес-процесів;

• проникнення Інтернет у всі сфери діяльності;

• прогрес в області інформаційних технологій: вдосконалення СУБД і сховищ даних;

• прогрес в області виробничих технологій: стрімке зростання продуктивності комп'ютерів, об'ємів накопичувачів, впровадження Grid систем.

II. Методи і задачі Data Mining

Основна особливість Data Mining - це поєднання широкого математичного інструментарію (від класичного статистичного аналізу до нових кібернетичних методів) і останніх досягнень у сфері інформаційних технологій. В технології Data Mining гармонійно об'єдналися строго формалізовані методи і методи неформального аналізу, тобто кількісний і якісний аналіз даних.

До методів і алгоритмів Data Mining відносяться наступні: штучні нейроні мережі, дерева рішень, символьні правила, методи найближчого сусіда і к- найближчого сусіда, метод опорних векторів, байесові мережі, лінійна регресія, кореляційно-регресійний аналіз; ієрархічні методи кластерного аналізу, неієрархічні методи кластерного аналізу, у тому числі алгоритми к-середніх і к-медіани; методи пошуку асоціативних правил, у тому числі алгоритм Apriori; метод обмеженого перебору, еволюційне програмування і генетичні алгоритми, різноманітні методи візуалізації даних і безліч інших методів.

Більшість аналітичних методів, що використовуються в технології Data Mining - це відомі математичні алгоритми і методи. Новою в їх застосуванні є можливість їх використовування при рішенні тих або інших конкретних проблем, обумовлена новими можливостями технічних і програмних засобів, що з'явилися. Слід зазначити, що більшість методів Data Mining була розроблена в рамках теорії штучного інтелекту.

Єдиної думки щодо того, які задачі слід відносити до Data Mining, немає. Більшість авторитетних джерел перераховує наступні: *класифікація, кластеризація, прогнозування, асоціація, візуалізація, аналіз і виявлення відхилень, оцінювання, аналіз зв'язків, підведення підсумків.* Розглянемо деякі з них.

Класифікація (Classification). Це найпростіша і поширена задача Data Mining. В результаті рішення задачі класифікації виявляються ознаки, які характеризують групи об'єктів досліджуваного набору даних - класи; по цих ознаках новий об'єкт можна віднести до того або іншого класу. Для вирішення задачі класифікації можуть використовуватися методи: найближчого сусіда (Nearest Neighbor); к-ближайшего сусіда (к-Nearest Neighbor); байесові мережі (Bayesian Networks); індукція дерев рішень; нейроні мережі (neural networks).

Кластеризація (Clustering) Кластеризація є логічним продовженням ідеї класифікації. Це задача складніша, особливість кластеризації полягає в тому, що класи об'єктів спочатку не визначені. Результатом кластеризації є розбиття

об'єктів на групи. Прикладом методу задачі кластеризації є особливий вид нейроних мереж (карти Кохонена), що самоорганізовуються без вчителя..

Асоціація (Associations). В ході рішення задачі пошуку асоціативних правил відшукуються закономірності між зв'язаними подіями в наборі даних. Відмінність *асоціації* від двох попередніх задач Data Mining: пошук закономірностей здійснюється не на основі властивостей аналізує мого об'єкту, а між декількома подіями, які відбуваються одночасно. Самий відомий алгоритм рішення задачі пошуку асоціативних правил - алгоритм Apriori.

Послідовність *(*Sequence), або послідовна асоціація *(sequential association)* Послідовність дозволяє знайти тимчасові закономірності між транзакціями. Задача послідовності подібна асоціації*, але* її метою є встановлення закономірностей не між одночасно наступаючими подіями, а між подіями, зв'язаними в часі (тобто що відбуваються з деяким певним інтервалом в часі. Цю задачу Data Mining також називають задачею знаходження послідовних шаблонів (sequential *pattern).* Правило послідовності*:* після події X через певний час відбудеться подія У.

Прогнозування (Forecasting). В результаті рішення задачі прогнозування на основі особливостей існуючих даних оцінюються пропущені або ж майбутні значення цільових чисельних показників. Для вирішення таких задач широко застосовуються методи математичної статистики, нейроні мережі і ін.

Візуалізація (Visualization, Graph Mining) В результаті візуалізації створюється графічний образ аналізованих даних. Для вирішення задачі візуалізації використовуються графічні методи, що показують наявність закономірностей в даних. Приклад методів візуалізації - представлення даних в 2-D і 3-D вимірюваннях.

Підведення підсумків (Summarization) - задача, мета якої - опис конкретних груп об'єктів з аналізованого набору даних та інші.

Задачі Data Mining, залежно від моделей, що використовуються, можуть бути ***дескриптивними*** і ***прогнозуючими***. В результаті рішення описових (descriptive) задач аналітик одержує шаблони, що описують дані, які піддаються інтерпретації. Ці задачі описують загальну концепцію аналізованих даних, визначають інформативні, підсумкові, відмітні особливості даних.

Прогнозуючі (predictive) задачі грунтуються на аналізі даних, створенні моделі, прогнозі тенденцій або властивостей нових або невідомих даних.

III. Класифікація стадій Data Mining

Data Mining може складатися з двох або трьох стадій:

Стадія 1. Виявлення закономірностей (вільний пошук**).**

Стадія 2. Використовування виявлених закономірностей для прогнозу невідомих значень (прогностичне моделювання).

На додаток до цих стадій іноді вводять стадію оцінювання (валідації) , наступну за стадією вільного пошуку. Мета валідації - перевірка достовірності знайдених закономірностей. Проте, ми вважатимемо валідацію частиною першою стадії, оскільки в реалізації багатьох методів, зокрема, нейроних мереж і дерев рішень, передбачений розподіл загальної множини даних на навчальні і

перевірочні, і останні дозволяють перевіряти достовірність отриманих результатів.

Стадія 3. Аналіз виключень - стадія призначена для виявлення і пояснення аномалій, знайдених в закономірностях.

Вільний пошук (Discovery). На стадії вільного пошуку здійснюється дослідження набору даних з метою пошуку прихованих закономірностей. Попередні гіпотези щодо виду закономірностей тут не визначаються. **Закономірність** (law) - істотний і постійно повторюється взаємозв'язок, що визначає етапи і форми процесу становлення, розвитку різних явищ або процесів.

Система Data Mining на цій стадії визначає шаблони, для отримання яких в системах OLAP, наприклад, аналітику необхідно обдумувати і створювати множину запитів. Тут же аналітик звільняється від такої роботи - шаблони шукає за нього система. Особливо корисно застосування даного підходу в надвеликих базах даних, де уловити закономірність шляхом створення запитів достатньо складно, для цього вимагається перепробувати безліч різноманітних варіантів. Вільний пошук представлений такими діями:

• виявлення закономірностей умовної логіки (conditional logic);

• виявлення закономірностей асоціативної логіки (associations and affinities);

• виявлення трендів і коливань (trends and variations).

Описані дії в рамках стадії вільного пошуку виконуються при допомозі:

• індукції правил умовної логіки (задачі класифікації і кластеризації, опис в компактній формі близьких або схожих груп об'єктів);

• індукції правил асоціативної логіки (задачі асоціації і послідовності і витягування при їх допомозі інформація);

• визначення трендів і коливань (початковий етап задачі прогнозування).

На стадії вільного пошуку також повинна здійснюватися валідація закономірностей, тобто перевірка їх достовірності на частини даних, які не брали участь у формуванні закономірностей.

Прогностичне моделювання (Predictive Modeling) .Друга стадія Data Mining - прогностичне моделювання - використовує результати роботи першої стадії. Тут знайдені закономірності використовуються безпосередньо для прогнозування. Прогностичне моделювання включає такі дії:

• прогноз невідомих значень (outcome prediction);

• прогнозування розвитку процесів (forecasting).

В процесі прогностичного моделювання розв'язуються задачі класифікації і прогнозування. При рішенні задачі класифікації результати роботи першої стадії (індукції правил) використовуються для віднесення нового об'єкту з певною упевненістю до одного з відомих, наперед визначених класів на підставі відомих значень. При рішенні задачі прогнозування результати першої стадії (визначення тренда або коливань) використовуються для прогнозу невідомих (пропущених або ж майбутніх) значень цільової змінної (змінних).

Порівняємо вільного пошуку і прогностичного моделювання з погляду логіки Вільний пошук розкриває загальні закономірності. Він по своїй природі *індуктивний*.

Закономірності, отримані на цій стадії, формуються від часткового до загального. В результаті ми одержуємо деяке загальне знання про деякий клас об'єктів на підставі дослідження окремих представників цього класу.

Прогностичне моделювання, навпаки, *дедуктивне*. Закономірності, отримані на цій стадії, формуються від загального до часткового. Тут ми одержуємо нове знання про деякий об'єкт або ж групі об'єктів на підставі:

• знання класу, до якого належать досліджувані об'єкти;

• знання загального правила, діючого в межах даного класу об'єктів.

Аналіз виключень (forensic analysis). На третій стадії Data Mining аналізуються виключення або аномалії, виявлені в знайдених закономірностях. Дія, виконувана на цій стадії, - виявлення відхилень (deviation detection). Для виявлення відхилень необхідно визначити норму, яка розраховується на стадії вільного пошуку. Стадія аналізу виключень може бути використана як очищення даних.

1. Препроцесінг інформації

Чотири закони теорії інформації:
Інформація, яка є, - не та, яка потрібна.
Інформація, яку хотілось би отримати, - не та, яка насправді потрібна.
Інформація, яка насправді потрібна, - недоступна.
Інформація, яка доступна, коштує більше, ніж можна заплатити.

Успішне розв'язання задач ідентифікації, прогнозування та діагностики неможливе без попередньої формалізації задачі та попередньої підготовки даних. Препроцесінгу апріорної інформації присвячені численні монографії та статті. У більшості випадків він зводиться до визначення інформативних або вагомих параметрів (ознак), вирівнюванню їх розподілу та приведенню до безрозмірного вигляду. Увага до релевантних задач викликана декількома аспектами:

- «прокляття розмірності» зумовлює необхідність скорочення кількості вхідних факторів;
- вплив шумових ефектів на вхідні фактори спотворює їх значення і, відповідно, значення результуючих характеристик. Тому необхідно вилучати найбільш зашумлені фактори без зниження загальної інформативності та визначати і зменшувати шумову присутність;
- необхідність обробки різнотипних факторів визначає необхідність нормалізації та стандартизації їх значень;
- нерівномірний розподіл значень факторів зменшує точність ідентифікації та прогнозування, тому актуальною є задача вирівнювання їх розподілу.

Інформативність факторів визначається тим, наскільки вони впливають на точність ідентифікації шуканих залежностей. Очевидно, що склад множини інформативних факторів залежить від конкретної задачі. У загальному випадку залежності можна розділити на лінійні та нелінійні. І якщо для лінійних залежностей інформативні фактори визначаються за відомими методами кореляційного та дисперсійного аналізу, то для нелінійних залежностей такі процедури найчастіше є емпіричними.

Майже парадоксальним є твердження про те, що максимальна ентропія значень вхідних факторів є необхідною умовою мінімальної ентропії значень результуючої характеристики. В даній лекції розглянуто моделі, методи та алгоритми, які вказують на конструктивні елементи реалізації такої умови, що включають у себе нормалізацію значень факторів, встановлення вагомих факторів та збільшення їх інформативності. Саме реалізація такого комплексу задач супроводжує процес максимально ефективного розв'язання задачі ідентифікації і прогнозування.

1.1. Ентропія і кількість інформації

Виникнення теорії інформації пов'язують з появою фундаментальної роботи американського вченого К. Шеннона «Математична теорія зв'язку» в 1948 році. Ним була запропонована, а радянським вченим Л.Я. Хінчиним доведена одиничність функціоналу ентропії інформації

$$H(U) = -C\sum_{i=1}^{p} p_i \log p_i , \qquad (1.1)$$

де C - додатна константа.

Цей функціонал вказує на міру невизначеності вибору дискретного стану з ансамблю U (потрапляння в один із станів, що утворюють множину станів U). Якщо для системи можливі n станів $u_1, u_2, ..., u_n$ і відомі ймовірності настання цих станів $p_1, p_2, ..., p_n$, то міра невизначеності має наступні властивості:

1) $H(U)$ - неперервна функція ймовірностей станів з виконанням умови $\sum_{i=1}^{n} p_i = 1$.

2) $H(U) = H_{\max}$, якщо $p_i = 1/n$ для всіх $i = \overline{1,n}$.

3) $H(U) = H_{\min} = 0$, якщо $\exists i : p_i = 1$ та $p_j = 0, \forall j \neq i$.

4) $H(U) \in R_+$ (дійсна, невід'ємна функція).

5) $H(X \cup Y) = H(X) + H(Y)$, якщо X та Y статистично незалежні.

6) Ентропія характеризує середню невизначеність вибору одного стану з ансамблю.

Міру знятої (виключеної) невизначеності називають кількістю інформації $I(U)$ та обчислюють як різницю

$$I(U) = H_{apriori}(U) - H_{aposteriori}(U) \qquad (1.2)$$

між ентропією до проведення досліду та її ж значенням після проведення досліду. Помилковим було б вважати, що кожний новий дослід лише зменшує невизначеність. Іноді значення $I(U)$ може бути від'ємним.

Таблиця 1.2. Початкові дані

X_1	X_2	...	X_n	Y
x_{11}	x_{12}	...	x_{1n}	y_1
x_{21}	x_{22}	...	x_{2n}	y_2
...	...	...	...	...
x_{m1}	x_{m2}	...	x_{Nn}	y_m
$x_{m+1,1}$	$x_{m+1,2}$	...	$x_{m+1,n}$	$y_{m+1} = ?$

Припустімо, початкові дані знаходяться в табл. 1.2, де $X = (x_1, x_2, ..., x_n)$ - вектор вхідних факторів, а Y - результуюча характеристика. Точність прогнозування значення y_{m+1} залежатиме від інформативності m навчальних образів, яка, у свою чергу, визначається існуванням залежностей між факторами та виконанням певних операцій, що становлять зміст процедури препроцесінгу вихідної інформації.

Виходячи з викладеного, на початковому етапі інтелектуального аналізу даних необхідно виконати 4 задачі:

- нормалізація факторів і приведення їх до єдиної шкали;

- перевірка на мультиколінеарність та гетероскедастичність;
- вибір оптимального набору факторів, що описують досліджуване явище;
- підвищення інформативності ознак, що лишилися;

1.2. Нормалізація і стандартизація вихідних значень

Оскільки значення векторів $X = (x_1, x_2, ..., x_n)$ у загальному випадку різнотипні, то їх необхідно привести до єдиної шкали. Це потрібно для адекватного застосування математичних методів і комп'ютерних розрахунків при пов'язаних із великими і малими абсолютними величинами обчисленнях, а також для того, аби встановити відповідність між кількісними та якісними значеннями.

Наприклад, вкрай важко відповісти на питання у стилі: „Що більш природно для людини в 25 років: мати 60 кг ваги або 165 см росту?", оскільки згадані числові показники відносяться не просто до різних вимірів, а ще й вимірюються на різних шкалах. А тим часом відповіді на питання подібного типу і їх комбінації важливі при оцінці схильності людини до певних захворювань.

Важливим кроком, який дає можливість порівняння, є нормування. Основними формулами, що реалізовують нормування і стандартизацію, є такі:

$$x_i' = \frac{x_i - \min(x_i)}{\max(x_i) - \min(x_i)}; \quad (1.3)$$

$$x_i' = \frac{\max(x_i) - x_i}{\max(x_i) - \min(x_i)}; \quad (1.4)$$

$$x_i' = \frac{x_i - \overline{x_i}}{\sigma_{x_i}}; \quad (1.5)$$

$$x_i' = \frac{2(x_i - \min(x_i))}{\max(x_i) - \min(x_i)} - 1; \quad (1.6)$$

$$x_i' = \frac{1}{1 + e^{-x_i}}. \quad (1.7)$$

Дамо їм коротку характеристику.

Перетворення (1.3). Область значень - [0,1]. Рекомендується використовувати, якщо значення початкових даних рівномірно заповнюють область дослідження. Для деяких методів прогнозування формула неефективна у випадку рівності значень нулю або їх зосередження біля кінців відрізка [0,1].

Перетворення (1.4). Аналогічне першому, але дозволяє зворотно пропорційно розвернути шкалу, що зручно у випадках, коли більшість характеристик мають максимізуватися, а дана характеристика - мінімізуватися. Недоліки ті ж самі.

Перетворення (1.5). Відрізняється тим, що отримані в результаті його застосування значення є безрозмірними, знаходяться з дисперсією та СКВ, рівними 1, на відрізку $\left[\frac{x_{\min} - \bar{x}}{\sigma_x}; \frac{x_{\max} - \bar{x}}{\sigma_x}\right]$ переважно в околі нуля. У перетворенні використовується $\bar{x}_i$ - вибіркове середнє значення $i-$тої змінної, σ_{x_i} - її ж ви-

біркове середньоквадратичне відхилення. Для використання такого перетворення, зокрема при навчанні нейронних мереж, необхідно застосовувати додаткові перетворення, наприклад

$$x \to \frac{x-\bar{x}}{\sigma_x} \to \frac{1}{1+e^{-\left(\frac{x-\bar{x}}{\sigma_x}\right)}} \tag{1.8}$$

Останнє перетворення, окрім належності значень інтервалу $(0;1)$, гарантує також більш рівномірний розподіл значень.

Перетворення (1.6). Область значень $[-1;1]$. Формула зручна для використання при прогнозуванні із застосуванням нейронних мереж, в яких активаційною функцією є гіперболічний тангенс. Має всі ті ж переваги й недоліки, що і перетворення (1.3) та (1.4).

Перетворення (1.7). Область значень $(0;1)$. Використовується рідко, здебільшого для значного підсилення реакції на зміни значень в околі нуля. Функція є допоміжною, вона не позбавляє розмірності значення факторів.

У загальному випадку вважатимемо, що використання функцій нормування веде до відображення вхідних значень в одиничний гіперкуб. Якщо вони будуть нерівномірно розподілені і зосереджені в невеликих гіперколах, то такі дані є малоінформативними і прогнозування буде неточним (приклад - рис. 1.1).

Найбільшу інформативність (у сенсі отримання більш точного прогнозу) мають дані з рівномірним розподілом (відомо, що вони мають найбільшу ентропію) (приклад – рис. 1.2). Таким чином, однією із головних задач після одержання безрозмірних величин і нормалізації буде максимізація ентропії.

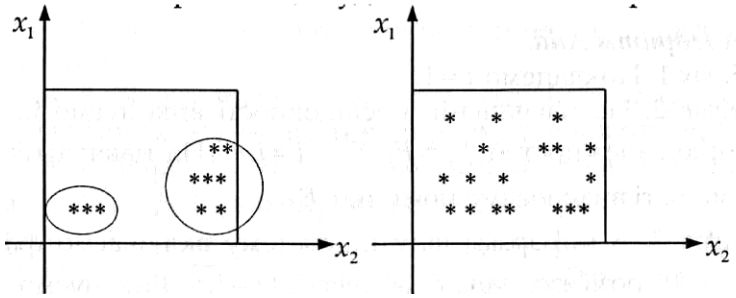

Рис. 1.1 - Неінформативні дані Рис. 1.2 - Інформативні дані

1.3. Алгоритм „вибілювання" входів

Як вже визначено вище, нормування та приведення до єдиної шкали збільшують інформативність даних. Проте цього виявляється недостатньо. Відомо, що якщо фактори статистично залежні, то їх спільна ентропія менше суми ентропій окремих факторів, тобто:

$$H(X,Y) \le H(X) + H(Y) \tag{1.9}$$

Найпростішим прикладом цього є процес купівлі телевізора і DVD-програвача. Очевидно, що невизначеність при одночасному придбанні комплекту відеотехніки однієї марки менша, ніж якби вони були різних марок або ку-

пувались у різний час. В якості визначальних факторів тут виступають споживчі властивості.

Досягнення статистичної незалежності входів тим самим буде забезпечувати максимальну інформаційну насиченість кожного з вхідних факторів окремо. Статистична незалежність - умова, досягти якої досить складно, тому на першому кроці здійснимо декореляцію входів за наступним алгоритмом „вибілювання" входів.

Крок 1. Для кожного вхідного фактора (за схемою, представленою в таблиці 2.2) знайдемо його середнє значення

$$\bar{x}_j = \frac{1}{m}\sum_{i=1}^{m} x_{ij}\ ,\ j = \overline{1,n} \tag{1.10}$$

Крок 2. Обчислимо коваріаційну матрицю K, елементи якої розрахуємо за формулою:

$$k_{ij} = \frac{1}{m-1}\sum_{l=1}^{m}\left(x_{li} - \bar{x}_i\right)\left(x_{lj} - \bar{x}_j\right),\ i,j = \overline{1,n} \tag{1.11}$$

Крок 3. Визначимо лінійне перетворення, яке діагоналізуватиме коваріаційну матрицю. Це дозволить зробити матриця, складена із стовпчиків, які є власними векторами матриці K: $KV = \lambda V$, де λ - власні числа матриці K, V - матриця з власних векторів матриці K.

Крок 4. Виконаємо перетворення

$$\widetilde{X} = X_{norm} \cdot V / \sqrt{\lambda}\ , \tag{1.12}$$

де матрицю X_{norm} одержують з X за (1.10) відніманням від елементів кожного стовпчика його середнього значення.

Крок 5. Закінчення алгоритму.

У результаті застосування "вибілювання" входів усі вхідні фактори стають некорельованими і мають дисперсію, що дорівнює одиниці. Очевидно, що внаслідок такого перетворення сумісна ентропія будь-якої пари факторів збільшується, оскільки розподіл елементів у вибірці вирівнюється і стає ближчим до рівномірного. Легко здійснити і зворотне перетворення.

У методичних вказівках до лабораторних робіт наведено лістинг функцій для прямого й зворотного перетворення. Не вдаючись у подробиці векторної алгебри, зазначимо, що модуль використовує у своїх перетвореннях стандартні функції отримання власних чисел та власних векторів пакету Matlab.

1.4. Виключення неінформативних факторів

Продовжуючи оптимізувати структуру початкової інформації, необхідно розв'язувати задачу вилучення лінійної залежності серед вхідних факторів. Одним з методів її розв'язання є універсальний алгоритм Фаррара-Глобера щодо вилучення мультиколінеарності, який базується на одночасній перевірці критеріїв Пірсона, Фішера та Стьюдента.

Алгоритм передбачає виконання наступних кроків:

Крок 1. Нормування та центрування факторів за (1.5).

Крок 2. Знайти вибіркову кореляційну матрицю

$$R = \frac{1}{n}(X')^T X' \tag{1.13}$$

Крок 3. Розрахувати значення критерію Пірсона

$$\chi^2 = -\left(m - 1 - \frac{1}{6}(2n+5)\right) \cdot \ln(\det(R)), \tag{1.14}$$

де n - кількість факторів; m - кількість рядків у таблиці спостережень.

Порівнюючи отримане за (1.14) значення з табличним при обраному рівні значущості α і кількості ступенів свободи $t = n(n-1)/2$, роблять наступний висновок: якщо $\chi^2 > \chi^2_{табл}|_{t,\alpha}$, то між векторами факторів має місце мультиколінеарність, інакше – кінець алгоритму.

Крок 4. Для визначення, яка саме змінна колінеарна з іншою (іншими), визначаємо обернену матрицю кореляцій

$$D = R^{-1}. \tag{1.15}$$

Крок 5. Для кожної зі змінних $k = \overline{1,n}$ розраховуємо критерій Фішера

$$F_k = \frac{m-n}{n-1}|d_{kk} - 1| \tag{1.16}$$

де d_{kk} - діагональні елементи матриці D за (1.15).

Розраховані значення порівнюються з табличними при $m-n$ та $n-1$ ступенях свободи по координатах і рівні значущості α. Якщо $F_k > F_{табл}|_{m-n,n-1,\alpha}$, то $k-$та змінна мультиколінеарна з іншими.

Крок 6. Знайти вибіркові часткові коефіцієнти кореляції

$$P_{kj} = \frac{id_{kj}}{\sqrt{d_{kk}d_{jj}}} \tag{1.17}$$

Крок 7. Обчислюємо для кожної пари $t-$критерій Стьюдента

$$t_{kj} = \frac{P_{kj}\sqrt{m-n}}{\sqrt{1-P_{kj}^2}} \tag{1.18}$$

Розраховані значення t_{kj} порівнюють з табличними при $m-n$ ступенях свободи і рівні значущості α. Якщо $|t_{kj}| > t_{табл}|_{m-n,\alpha}$, то змінна X_k колінеарна змінній X_j.

Одним із традиційних методів подолання мультиколінеарності є *вилучення* з множини вхідних факторів лінійно залежних, визначених за наведеним вище алгоритмом. Інший метод полягає у *заміні* одного з лінійно залежних факторів на лінійну комбінацію факторів (найпоширенішою є різниця входів).

Ще один метод полягає в наступному:

Обчислюємо матрицю коваріацій $K = (k_{ij})_{i,j=1}^n$ та її власні числа $\Lambda = \{\lambda_i\}_{i=1}^n$ із рівняння $K \cdot x = \Lambda \cdot x$, де x - власний вектор матриці K. Відомо, що власні числа є квадратами дисперсій матриці K уздовж її головних осей. Якщо власні

числа є достатньо малими, це свідчитиме про те, що значення дисперсії є малим. Відповідно, гіперповерхня, що описує вхідні дані, втрачає виміри (перетворює вимір чи їх групу на константу регресійного рівняння). Як наслідок, подібна ситуація вказує на те, що реальна розмірність вхідної множини менше заданої. Тоді розмірність входів необхідно знизити, виключаючи ті з них, яким відповідають власні числа, що мають абсолютні значення менше певного заданого $\varepsilon > 0$. Точність моделі при цьому, у більшості випадків, зменшується незначно.

1.5. Аналітико-евристичні алгоритми визначення вагомих інформативних ознак

Новосибірською школою під керівництвом професора М.Г. Загоруйка розроблені методи визначення інформативних ознак, які базуються на евристичних судженнях та аналітичних розрахунках. Деякі з них розглянемо далі.

Алгоритм Del.

Крок 1. Покладемо $i = 0$ та $j = 1$. На навчальній послідовності виконуємо ідентифікацію залежності $Y = F(X_1, X_2, ..., X_n)$ одним з обраних методів, а на контрольній послідовності знаходимо помилку E_{i0}.

Крок 2. Вилучаємо X_j і аналогічно кроку 1 виконуємо ідентифікацію залежності $Y = F(X_1, ..., X_{j-1}, X_{j+1} ..., X_n)$ та знаходимо помилку E_{ij}.

Крок 3. Якщо $j = n$, то здійснюємо перехід на крок 4, в іншому випадку покладемо $j = j + 1$ і переходимо на крок 2.

Крок 4. Із множини факторів вилучаємо X_k, де k є розв'язком задачі пошуку $\arg\min_j (E_{ij} - E_{i0})$, для $j = \overline{1, n}$. Оновлюємо нумерацію факторів з урахуванням вилученого $j = k$ та $n = n - 1$.

Крок 5. Якщо $n = N$, де N - наперед задана максимальна припустима кількість вхідних факторів, то виконуємо перехід на крок 6, в іншому випадку $i = i - 1$ і переходимо до кроку 1.

Крок 6. Закінчення алгоритму.

Алгоритм Add.

Крок 1. Покладемо $i = 1$.

Крок 2. На навчальній послідовності виконуємо ідентифікацію залежностей $Y = F(X_j)$, де $j = \overline{1, n}$. На контрольній послідовності визначаємо помилки E_j.

Крок 3. В інформативну підсистему включаємо фактор X_k, де k - розв'язок задачі $\arg\min_j (E_j)$ для $j = \overline{1, n}$. Оновлюємо нумерацію факторів з урахуванням використаного $j = k$.

Крок 4. Кладемо $i = i + 1$. Якщо $i = N$, де N - наперед задана максимальна припустима кількість вхідних факторів, то переходимо на крок 7, в іншому випадку – перехід на крок 2.

Крок 5. На навчальній послідовності виконуємо ідентифікацію залежностей $Y = F(X_i, X_j)$ де $j = \overline{1,n}$. На контрольній послідовності визначаємо помилки E_{ij}.

Крок 6. Включає 3-5, але виконується додавання 1 фактора до існуючої множини.

Крок 7. Закінчення алгоритму.

Комбіновані алгоритми.

Комбінованими є такі алгоритми, в яких відбувається композиція елементів алгоритмів *Add* і *Del*. У першому із них вилучаємо певну кількість факторів згідно із алгоритмом *Del*, потім деяку частину факторів додаємо відповідно до алгоритму *Add*. Таку послідовність операцій повторюємо задану кількість разів. У другому випадку послідовність чергування алгоритмів *Add* і *Del* виконується навпаки: спочатку алгоритм *Add*, потім - *Del*.

Експериментально доведено, що комбіновані алгоритми мають більш високу точність. Найнижчу точність мають результати застосування алгоритму *Del*.

Алгоритм випадкового пошуку з адаптацією (ВПА).

Крок 1. Покладемо $k = 1$. Нехай кількість ознак, які вважатимемо інформативними, є n (кількість ознак, які мають залишитися у функції, відповідно N). Розіб'ємо інтервал (0;1) на n однакових відрізків таким чином, щоб $i-$тому відрізку відповідала $i-$та ознака, $i = \overline{1,n}$. Нарешті, покладемо $j = 1$.

Крок 2. Генеруємо рівномірно розподілене випадкове число з інтервалу (0;1).

Крок 3. Якщо число належить $i-$тому інтервалу, то $i-$та ознака включається в множину інформативних ознак.

Крок 4. Якщо $j = N$, то переходимо на крок 5, в іншому випадку $j = j + 1$ і виконуємо перехід на крок 2.

Крок 5. Виконуємо ідентифікацію залежності $Y_k = F(X_{k_1}, X_{k_2}, ..., X_{k_N})$ і на контрольній вибірці знаходимо помилку E_k.

Крок 6. Якщо $k = r$, де r може розглядатися як кількість незалежних агентів (мурах, хромосом, бактерій), то здійснюємо перехід на крок 7, в іншому випадку - $k = k + 1,\ j = 1$ і переходимо до кроку 2.

Крок 7. Знаходимо значення $k_{\min}$ та $k_{\max}$ серед усіх k_l, $l = \overline{1,n}$ за значенням похибки $extr(E_{k_l})$.

Крок 8. Для факторів, які входять до залежності з номером $k_{\max}$, зменшуємо на величину d (задана наперед константа) відповідні інтервали належності,

а для факторів, що входять до залежності з номером $k_{\min}$, збільшуємо їх на ту ж величину $d < 1/n$. Сумарна довжина інтервалів залишається рівною одиниці.

Крок 9. Якщо у послідовність інформативних ознак декілька разів підряд потрапляють одні й ті самі ознаки (інший варіант – кілька поколінь поспіль не забезпечують покращення результату), то процес їх підбору закінчено і переходимо на крок 10, в іншому випадку – $j = 1$, $k = 1$ і виконуємо перехід на крок 2.

Крок 10. Закінчення алгоритму.

Особливістю алгоритму є суб'єктивізм при виборі значення $\frac{1}{r \cdot n} < d < \frac{1}{n}$. Якщо d є порівняно великим, то процес пошуку інформативної підсистеми є швидким, але точність - низькою. У протилежному випадку точність є вищою. В літературі зазначено, що практично прийнятні результати були одержані при $r = 10$ та кількості циклів ітерацій від $10N$ до $15N$.

Алгоритм спрямованого таксономічного пошуку ознак (СТПО).

Крок 1. Розв'язати задачу кластеризації n початкових ознак на m кластерів (позначення – аналогічні попереднім методам), $N < m < n$.

Крок 2. Вибрати типову ознаку в кожному кластері. Такі ознаки будуть максимально незалежними одна від іншої.

Крок 3. Визначити число N. Виконуючи перебір факторів із m по N (всього C_m^N разів) та ідентифікуючи на навчальній вибірці відповідні залежності і перевіряючи їх точність на контрольній вибірці, робимо висновок про найкращий склад множини з N інформативних ознак.

1.6. Методика „Box-counting"

Розвинена теорія лінійної множинної регресії не залишає і краплі сумніву в правильності отриманих результатів та розроблених методів. Безсумнівним є і той факт, що переважна більшість природних процесів носить нелінійний характер, а тому застосування лінійної регресійної моделі є обмеженим, а сама вона, її розвиток і вдосконалення служить засобом переважно науково-теоретичного пошуку.

Теорія нелінійних процесів в частині їх ідентифікації, оцінки якості, статистичних оцінок, а також застосування для прогнозування, розвинута недостатньо. Для такого стану справ існують об'єктивні і суб'єктивні причини. Не заглиблюючись в них, виконаємо аналіз одного з методів визначення вагомості вхідних факторів, як аспекту зменшення початкової ентропії, що грає особливо важливу роль при прогнозуванні на „коротких" вибірках.

Розглянемо методику „box-counting". Її сутність полягає в наступному. Є вхідні фактори $X_1, X_2, ..., X_n$, значення кожного з них знаходяться в обмеженій області, тобто $x_{ij} \in \Omega_i$, $X_i = \{x_{ij}\}$, $j = \overline{1, m}$.

Згідно із положеннями теорії інформації та теорії ймовірностей, мірою передбачуваності значення фактора X_k є його ентропія, яка визначається за формулою (1.1). Ентропія є максимальною, якщо всі значення фактора рівно ймові-

рні. В методиці „box-counting" ентропія оцінюється за кількістю заповнених чарунок, на які розбивається інтервал її можливих значень (рис. 1.3). Таким чином, кількісно ентропія є логарифмом ефективного числа заповнених чарунок $H(X_k) = \log N_{X_k}$.

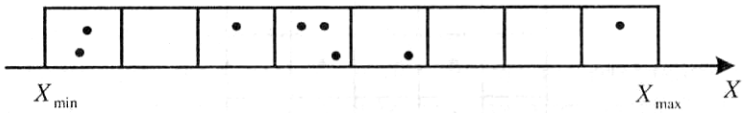

Рис. 1.3. Розподіл значень по чарунках

Очевидно, що ентропія збільшується із збільшенням кількості заповнених чарунок. Зростання ентропії (міри невизначеності) призводить до зменшення міри передбачуваності значень фактора. Якщо всі значення зосереджені в одній чарунці, то ентропія фактора дорівнює нулю (має місце повна визначеність). Рівномірному заповненню чарунок відповідає максимальна ентропія.

Передбачуваність результуючої характеристики Y, яка визначається знанням значень фактора X, розраховується як крос-ентропія (кількість інформації)

$$I(X,Y) = H(X) + H(Y) - H(Y/X). \quad (1.19)$$

Крос-ентропія є логарифмом відношення розсіювання значень змінної Y до типового розсіювання цієї змінної при відомому значенні змінної X

$$I(X,Y) = \log(N_x N_y / N_{xy}), \quad (1.20)$$

де N_{xy} - кількість чарунок, в яких знаходяться точки з координатами (X_i, Y_i). На рисунку 1.4 N_{xy} = 8. Кількість стовпчиків, у яких присутні значення X, позначено N_x і для рис. 1.5 N_x = 8. Аналогічно, кількість стовпчиків, у яких є значення Y, позначено N_y і N_y = 5. Таким чином, для наведеного прикладу $I(X,Y) = \log(5 \cdot 5/8) = 1{,}64$.

Зауважимо, що розрахунок проведено для логарифму за основою 2.

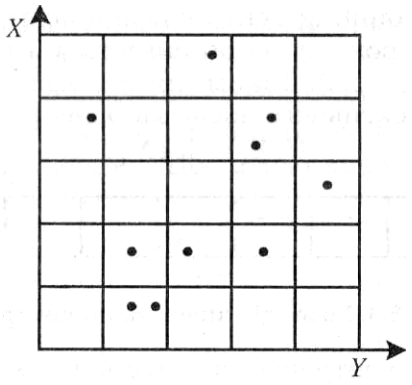

Рис. 1.4. Вихідні дані для розрахунку ***I(X,Y).***

Що більшим с значення крос-ентропії, то більше визначеності вносить знання значень X в прогноз значення Y. Підводячи підсумок, запропонуємо ***алгоритм реалізації методики „box-counting"***.

Крок 1. Припустимо, що існує залежність $Y = F(X_1, X_2, ..., X_n)$, вигляд якої наперед невідомий. Оскільки апріорно фактори X_j, $j = \overline{1,n}$ мають різну розмі-

рність, то їх необхідно нормувати і привести до шкали [0;1], що дозволить проводити адекватний аналіз.

Крок 2. Оберемо одиницю дискретності ε, що визначається точністю досліджень. Область визначення кожного фактора (відрізки [0;1]) розбиваємо на інтервали довжиною ε і розраховуємо N_{x_j}.

Крок 3. В двовимірному просторі (див. рис. 1.4) визначаємо N_{xy} для кожного фактора.

Крок 4. Для кожної пари (X_j, Y) розраховуємо значення відношення (1.20) та впорядковуємо послідовність $I(X_1, Y), I(X_2, Y), ..., I(X_n, Y)$ за зростанням значень.

Крок 5. Виходячи з початкових даних та досвіду, певну кількість факторів з найменшими значеннями крос-ентропії відкидаємо. Залишаються лише вагомі фактори.

Крок 6. Закінчення алгоритму.

Зауваження 1.1. На кроці 2 одиниці дискретності можуть бути різними для кожного фактора.

Зауваження 1.2. Одиниці дискретності для вхідних факторів на кроках 2 і 3 є однаковими.

2. Методи кластеризації

Обґрунтуванням актуальності розв'язання задачі кластеризації є необхідність усунення усереднення при ідентифікації невідомих залежностей, оскільки в цьому випадку прогнозні оцінки є зміщеними, а відтак їх не можна вважати адекватними. Тому ідентифікацію потрібно здійснювати в класі однотипних об'єктів чи процесів, передумовою чого є попередня кластеризація.

Важливо зауважити, що кластеризація і класифікація є основою як повсякденної діяльності людини, так і фундаментальним процесом наукової практики, оскільки навіть діти класифікують об'єкти навколишнього світу, а система класифікацій містить поняття, які є необхідною умовою розробки теорій та методів науки.

Задача **кластеризації** полягає у визначенні груп об'єктів (процесів), які є найближчими один до іншого за деяким критерієм. При цьому ніяких припущень про їх структуру, як правило, не робиться. Більшість методів кластеризації базується на аналізі матриці коефіцієнтів схожості, до яких належать відстань, кореляція та інші. Якщо критерієм або метрикою виступає відстань, то кластером називають групу точок Ω, таку, що середній квадрат внутрішньогрупової відстані до центру групи менше середньої відстані до загального центру в початковому наборі об'єктів, тобто $\bar{d}_\Omega < d^2$, де $\bar{d}_\Omega = \frac{1}{N}\sum_{X_i \in \Omega}\left(X_i - \bar{X}_\Omega\right)^2$, $\bar{X}_\Omega = \frac{1}{N}\sum_{X_i \in \Omega}\left(X_i\right)$, N - кількість точок в кластері Ω.

У загальному випадку, критеріями є:

1. Відстань Евкліда

$$d\left(X_k, X_l\right) = \sqrt{\frac{1}{m}\sum_{j=1}^{m}\left(X_{kj} - X_{lj}\right)^2}.$$

2. Максимальна відстань за ознаками (відстань Чебишева)

$$d\left(X_k, X_l\right) = \max_{1 \le j \le m}\left|X_{kj} - X_{lj}\right|.$$

3. Відстань Махаланобіса

$$d\left(X_k, X_l\right) = \sqrt{\left(X_k - X_l\right)R^{-1}\left(X_k - X_l\right)^T},$$

де R^{-1} - обернена до коваріаційної матриця.

4. Відстань Хеммінга

$$d\left(X_k, X_l\right) = \frac{1}{m}\sum_{j=1}^{m}\left|X_{kj} - X_{lj}\right|.$$

Розв'язання задачі мінімізації відстані між об'єктами рівносильне розв'язанню задачі мінімізації відстані до об'єкту, що має усереднені характеристики, оскільки, наприклад, для відстані Хеммінга

$$\sum_{j=1,k<l}^{m}\left|X_{kj}-X_{lj}\right|=\sum_{j=1,k<l}^{m}\left|X_{kj}-\overline{X}_j+\overline{X}_j-X_{lj}\right|\le\sum_{j=1,k<l}^{m}\left|X_{kj}-\overline{X}_j\right|+\sum_{j=1,k<l}^{m}\left|X_{lj}-\overline{X}_j\right|\le$$

$$\le\sum_{j=1}^{m}\left|X_{kj}-\overline{X}_j\right|+\sum_{j=1}^{m}\left|X_{lj}-\overline{X}_j\right|\le 2\cdot\max_{p\in\{k,l\}}\sum_{j=1}^{m}\left|X_{pj}-\overline{X}_j\right|$$

де $\overline{X}=\left(\overline{X}_1,\overline{X}_2,...,\overline{X}_m\right)$ - об'єкт з усередненими значеннями характеристик.

Задачі кластеризації супроводжують дві проблеми: визначення оптимальної кількості кластерів та розрахунок координат центрів цих кластерів. Початковими даними для задачі кластеризації є значення параметрів об'єктів дослідження. Найчастіше визначення оптимальної кількості кластерів є прерогативою дослідника. Припустимо, що число кластерів K задано наперед і при цьому $K<<m$, де m - кількість об'єктів. Отримаємо задачу

$$\sum_{i=1}^{K}\sum_{j=1}^{m_i}\left\|X_j-\overline{X}_i\right\|\to\infty \qquad (2.1)$$

де m_i, $i=\overline{1,K}$ - кількість об'єктів в $i-$тому кластері; $\overline{X}_i$, $i=\overline{1,K}$ - середнє значення в кластері; $\left\|X_j-\overline{X}_i\right\|$ - відстань між об'єктами.

Розв'язком задачі (2.1) є координати центрів кластерів $\overline{X}_i$, які можуть міститися серед початкових об'єктів (що є достатньо строгою умовою), але частіше – можуть бути представленими будь-якими точками області дослідження.

До **традиційних методів кластерного аналізу** відносять деревовидну кластеризацію, двохвходове об'єднання, метод $k-$середніх, метод дендритів, метод кореляційних плеяд, метод шарів та інші.

Перевагами вказаних методів є простота, інваріантність техніки реалізації щодо характеру початкових даних та метрик, що використовуються.

До недоліків відносять слабку формалізованість, що ускладнює застосування обчислювальної техніки, а також низьку точність, наслідком чого є попередні оцінки структури простору факторів та їх інформативності.

Ще один метод розв'язання задачі кластеризації базується на застосуванні спеціальної нейронної мережі – карти Кохонена, яка використовує самоорганізацію. Проблемою використання такої нейронної мережі є вибір початкових вагових коефіцієнтів, неперервний характер функціонування і ефективність, оцінка якої на сьогоднішній день залишається актуальною задачею.

Відомо, що головною ціллю кластерного аналізу є знаходження груп схожих об'єктів у вибірці даних. Кластер визначається такими характеристиками: щільність, дисперсія, розміри, форма та віддільність. Розглядаючи метричні простори, *щільність* визначимо як властивість, яка дозволяє вважати кластер згущенням точок у просторі даних, відносно щільне у порівнянні з іншими областями простору, які містять малу кількість точок або не містять взагалі. *Дисперсія* визначає міру розсіювання точок у просторі відносно центра кластера. *Розміри* кластеру тісно пов'язані з дисперсією: якщо кластер ідентифікований,

то і його радіус можна виміряти. Ця властивість корисна лише тоді, коли кластер є гіперсферою. *Форма* - розміщення точок кластеру в просторі. *Віддільність* характеризує міру перекриття кластерів і те, і наскільки далеко один від іншого вони розташовані.

Розроблені на наш час класичні методи кластеризації поділяються на такі групи:

– ієрархічні методи об’єднання та розподілу;
– ітеративні методи групування;
– методи пошуку модальних значень щільності;
– факторні методи;
– методи згущень;
– методи, що використовують теорію графів.

Ієрархічні методи об’єднання та розподілу.

Метод одиночного зв'язку. Нехай маємо матрицю подібності, приклад якої наведено в табл. 6.1. Аналіз значень коефіцієнтів подібності свідчить про те, що найбільш схожими є об'єкти O_4 і O_5, об'єднуємо їх (рис. 2.1). Другим за абсолютним значенням є коефіцієнт 0,7, що визначає схожість між O_3 і O_6, їх теж об'єднуємо. Наступним значенням є 0,6, яке визначає схожість між парами O_4 і O_5 та O_3 і O_6, як найбільше значення схожості між їх елементами, а також схожість між O_1 та парою O_4 і O_5. З'єднуємо їх. На останньому кроці з'єднуємо об'єкт O_2 з іншими об'єктами на рівні 0,4.

Таблиця 2.1. Початкові дані (коефіцієнти подібності)

	O_1	O_2	O_3	O_4	O_5	O_6
Q_1	-	0	0,2	0,6	0,24	0,45
O_2	-	-	0,3	0,4	0,35	0,3
Q_3	-	-	-	0,5	0,42	0,7
O_4	-	-	-	-	0,8	0,5
O_5	-	-	-	-	-	0,6
O_6	-	-	-	-	-	-

Зробимо деякі зауваження щодо методу одиночного зв'язку. По-перше, необхідно проглядати матрицю подібності та послідовно об'єднувати схожі об'єкти. По-друге, послідовність об'єднання кластерів представляється у вигляді деревоподібної діаграми (дендрограми). По-третє, для повної кластеризації необхідно здійснити $n-1$ цикл перегляду, де n - кількість об'єктів. По-четверте, внаслідок виконання достатньо простої процедури маємо вкладені кластери.

Метод одиночного зв'язку має як переваги, так і недоліки. Він є одним із небагатьох методів, результати застосування якого не змінюються при будь-яких перетвореннях даних, що залишають без змін відносне впорядкування елементів матриці подібності. Головним недоліком є те, що метод призводить до появи „ланцюжків", тобто великих продовгуватих кластерів, що робить майже неможливим адекватне визначення кількості кластерів.

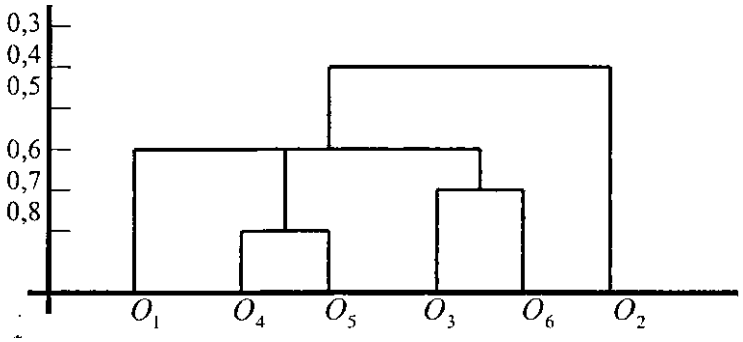

Рис. 2.1. Дендрограма методу одиничного зв’язку.

Метод повних зв'язків. На відміну від методу одиночного зв'язку, згідно із правилом об'єднання, коефіцієнт подібності між кандидатами на включення в кластер та елементами цього кластеру повинен бути не меншим від певного порогового значення. Дерево, одержане в результаті застосування методу, дає краще уявлення про кластерну структуру даних.

Метод середнього зв'язку має декілька варіантів застосування. У першому із них обчислюється середня подібність об'єкта, який потрібно кластеризувати, з усіма об'єктами кластеру. Якщо знайдене середнє значення подібності досягає або перевищує деяке задане порогове значення, об'єкт приєднується до кластеру. В іншому варіанті обчислюють коефіцієнт подібності між центрами ваги кластерів, що підлягають об'єднанню.

Метод Уорда призначений для визначення кластерів таким чином, щоб оптимізувати мінімальну дисперсію всередині кластерів. Цільова функція є такою $F_i = X_i^2 - \frac{1}{n}\left(\sum_{j=1}^{n} X_{ij}\right)^2$.

На першому кроці її значення дорівнює нулю. Далі об'єднуються ті об'єкти, для яких значення вказаної функції має найменший приріст. Метод має тенденцію до знаходження кластерів приблизно однакових розмірів.

Ітеративні методи групування.

В основі застосування ітеративних методів групування лежить наступний *базовий алгоритм, що також називають* ***Методом k-середніх****:*

Крок 1. Емпірично розбити об'єкти на вказане число кластерів. Обчислити центр кожного кластеру.

Крок 2. Помістити кожну точку даних у кластер із найближчим центром ваги.

Крок 3. Обчислити нові центри ваги кластерів. Кластери не змінюються до тих пір, поки не будуть переглянуті всі об'єкти.

Крок 4. Кроки 2 і 3 повторюються до тих пір, поки не перестануть змінюватись кластери.

Застосування ітеративних методів групування пов'язане з рядом проблем обчислювального характеру, тому використовують підготовчі процедури для вибору вихідного поділу, типу ітерацій, статистичних критеріїв.

Вибір початкового розбиття здійснюється двома шляхами: перший полягає у тому, що визначаються початкові точки - центри кластерів і далі обчислюється відстань від кожного об'єкту до центрів кластерів. Об'єкт належить тому кластеру, відстань до якого є найменшою. Згідно другого способу об'єкти довільно розбивають на кластери і знаходять їх центри як середні значення.

Існує два основні типи ітерацій: за принципом «$k-$середніх» і за принципом «сходження на гору». Ітерації за принципом „$k-$середніх" полягають у переміщенні об'єкта в кластер з найближчим центром ваги. Вони можуть бути комбінаторними або некомбінаторними. У першому випадку перерахунок центра кластеру здійснюється після кожної зміни його складу, в іншому - лише після того, як буде завершено перегляд усіх даних. Крім того, ітерації за принципом «$k-$середніх» є включаючими та виключаючими, у включаючих ітераціях після обчислення центру кластера об'єкт включається до складу кластера, у виключаючих - вилучається із кластера.

В ітераціях, що реалізуються за принципом «сходження на гору», переміщення об'єктів відбувається, виходячи із того, чи буде таке переміщення оптимізувати значення деякого статистичного критерію.

До функцій, що визначають **якість кластеризації** (статистичні критерії), належать trW, $trW^{-1}B$, $\det W$ та найбільше власне число матриці $W^{-1}B$.

У всіх них W - об'єднана внутрішньогрупова коваріаційна матриця, B - об'єднана міжгрупова коваріаційна матриця. Використовуючи кожний із статистичних критеріїв, знаходять кластери визначеного вигляду. Так, критерій trW орієнтований на утворення гіперсферичних однорідних кластерів. За критерієм $\det W$ передбачається, що у кластерів буде однакова форма, не обов'язково гіперсферична.

Проблема всіх ітеративних методів - при розв'язуванні задачі кластеризації мають місце також і субоптимальні розв'язки.

Інші методи кластеризації.

Реалізація *методів факторного аналізу* починається з формування кореляційної матриці подібності об'єктів. За її елементами визначаються факторні навантаження і виконується розподіл об'єктів по кластерах. До недоліків таких методів належать: необхідність обґрунтування застосування лінійної множинної регресії; проблема множинних факторних навантажень, оскільки існує проблема прийняття рішень, якщо об'єкт має високе навантаження більше ніж для одного фактора.

В основі *ієрархічних дивізимних (розділяючих) методів* лежить ідея поділу: спочатку всі об'єкти належать одному кластеру, а потім вони діляться на більшу кількість кластерів. При цьому можлива реалізація двох варіантів: монотетичного і політетичного. У першому варіанті кластери визначаються за однаковістю або близькістю значень однієї ознаки, у другому - до кластеру належать об'єкти, що мають певні співвідношення значень із деякої множини ознак.

У *методах пошуку модальних значень щільності* кластер визначають як область простору із високою щільністю образів у порівнянні з навколишнім се-

редовищем. Розрізняють два підходи: у першому підході базуються на методі одиночного зв'язку, у другому – на поділі "сумішей" багатовимірних імовірнісних розподілів. Особливістю першого підходу є те, що при появі нового образу він з певним пріоритетом утворює новий кластер, ніж приєднується до вже існуючого. Другий підхід базується на статистичній моделі, в якій елементи різних груп чи кластерів мають різний розподіл значень ознак.

Такі методи є чутливими до проблеми субоптимальних розв'язків, компоненти "сумішей" є багатовимірними нормальними розподілами, що визначає їх недоліки.

Методи згущення дозволяють утворювати кластери, які перекриваються. Вони вимагають обчислення матриці подібності між образами і встановлення оптимального значення стратегічного критерію. Оскільки ці методи на початку утворюють лише дві групи, то раціонально пропонувати декілька конфігурацій, кожна з яких оцінюється на придатність. Недоліком методів є те, що через невдалу пошукову процедуру відбувається повторне знаходження одних і тих же груп, що не надає нової інформації.

Порівняно новим напрямком у розробці методів кластеризації є методи, що базуються на теорії графів. Розвинений математичний апарат є альтернативою численним евристичним методам. Значного поширення набули методи, які базуються на пірамідальних мережах, що ростуть. Такі мережі дозволяють виконувати кластеризацію в режимі реального часу.

Гіпотеза компактності полягає в тому, що реалізації одного і того ж образу відображуються в просторі ознак у геометрично близькі точки, утворюючи «компактні» згустки у припущенні, що проведена попередня обробка образів. Міра компактності може бути різною, але найчастіше це Евклідова відстань.

Алгоритм *Forel*

Крок 1. Ознаки об'єктів нормуються так, щоб їх значення знаходились на відрізку [0;1], наприклад, $x'_{ij} = \dfrac{x_{ij} - x_{\min j}}{x_{\max j} - x_{\min j}}$, та лічильник $k = 0$.

Крок 2. Будуємо гіперсферу мінімального радіуса, що охоплює всі m точок. При нормуванні, запропонованому на попередньому кроці, такий радіус дорівнює $R_k = \sqrt{n}/2$, де n - кількість факторів або ознак об'єкта.

Крок 3. Зменшуємо радіус гіперсфери за формулою $R_{k+1} = R_k - \dfrac{k+1}{10} R_k$ та розміщуємо центр сфери в одній із точок (вибраної випадково).

Крок 4. Визначаємо точки, відстань від яких до центра гіперсфери менше R_{k+1} і обчислюємо координати їх центра ваги.

Крок 5. Переносимо центр сфери в центр ваги і знову визначаємо внутрішні точки. Ця операція циклічно повторюється.

Крок 6. Якщо склад множини внутрішніх точок і, як наслідок, координати центра ваги не змінюються, то сфера зупинилась в області локального максимуму щільності точок в просторі ознак.

Крок 7. Вилучаємо з розгляду точки, що належать сфері (таксону 1).

Крок 8. Якщо ще залишились «вільні» точки, то кроки 2-7 повторити, інакше - перейти на крок 9.

Крок 9. Якщо кількість таксонів є меншою, ніж задана, $k = k + 1$ і перейти на крок 3, інакше - на крок 10.

Крок 10. Закінчення алгоритму.

Існує ще модифікація наведеного базового алгоритму *ForeI*, яку називають *Forel-2*. У ній передбачена зміна радіуса на фіксовану величину на кожній ітерації. Разом із тим, відзначимо значний суб'єктивізм вибору як радіуса, так і його приросту, що часто призводить до неоптимальної кластеризації. Алгоритм *Forel-2* дозволяє отримати точно задане число кластерів.

Альтернативним методом розв'язання задачі кластеризації є використання ідей, які лежать в основі **еволюційного моделювання** і зокрема генетичного алгоритму. Базовою операцією є формування фітнес-функції.

Нагадаємо, що початковими даними задачі кластеризації є значення факторів-параметрів об'єктів (табл. 2.3). Заздалегідь, виконаємо їх нормування, наприклад, за формулою $x'_{ij} = \dfrac{x_{ij} - x_{\min j}}{x_{\max_j} - x_{\min_j}}$. Внаслідок такого перетворення значення всіх факторів належатимуть одиничному гіперкубу $[0;1]^n$.

Таблиця 2.3. Значення факторів дослідження

№ з/п	X_1	X_2	...	X_n
1	x_{11}	x_{12}	...	x_{1n}
2	x_{21}	x_{22}	...	x_{2n}
...	...	...	...	...
m	x_{m1}	x_{m2}	...	x_{Nn}

Реалізація фітнес-функції здійснюється за наступним алгоритмом (назвемо його *EvoClast*).

Крок 1. Значення фітнес-функції покласти рівним нулю $(F = 0)$.

Крок 2. Задати кількість кластерів k і вказати значення m.

Крок 3. Виконати ініціалізацію бінарної матриці належності елементів до кластерів T_k.

Крок 4. Для всіх об'єктів виконати наступні кроки. Покласти $i = 1$.

Крок 5. Обчислити відстані від $i-$того об'єкта до центрів всіх k кластерів, які є індивідами вибіркової популяції.

Крок 6. Серед усіх відстаней d_j, $j = \overline{1,k}$, вибрати мінімальну d_q і віднести $i-$тий об'єкт до $q-$того кластера. Внести відповідний запис в матрицю T_k.

Крок 7. $F = F + d_q$; $i = i + 1$.

Крок 8. Якщо кроки 5-7 виконані для всіх об'єктів, то отримано значення фітнес-функції F, в іншому випадку перейти на крок 5.

Крок 9. Закінчення алгоритму.

Очевидно, що алгоритм отримання фітнес-функції можна оптимізувати. Підвищення ефективності є його внутрішньою властивістю. Різноманіття варіантів операцій генетичного алгоритму представляють множину зовнішніх властивостей процесу отримання фітнес-функції. Можливість розв'язання задачі її оптимізації також припускає двійкове та десяткове представлення початкових даних. І якщо в першому випадку в процедурах генетичного алгоритму домінуючим є рівномірний розподіл, то у другому - при пошуку оптимального розв'язку перевага віддається значенням, що мають нормальний розподіл з математичним сподіванням, яке співпадає з центром кластеру. Визначення оптимальної дисперсії - ще одна задача, яка залишається нерозв'язаною.

Запропонований метод еволюційного моделювання, що базується на використанні генетичного алгоритму, ефективно функціонує при обробці масивів великої розмірності, оскільки в ньому оптимально поєднуються цілеспрямований пошук і елементи випадковості, направлені на вибивання цільової функції з локальних мінімумів. Ніяких попередніх умов для його використання не вимагається. Головною умовою оптимізації обчислень є правильна алгоритмізація розрахунку значень цільової функції. Багатовекторність процесу покращення швидкості алгоритму (для генетичних алгоритмів це особливо необхідно) і його точності (пошуку глобального мінімуму фітнес-функції), а також його актуальність свідчать про необхідність розв'язання задачі оптимізації еволюційного методу.

3. **Асоціативні правила**

Афінітивний аналіз (affinity analysis) — один з розповсюджених методів Data Mining. Його назва походить від англійського слова affinity, що у перекладі означає «близькість», «подібність». Ціль даного методу — дослідження взаємного зв'язку між подіями, які відбуваються спільно. Різновидом афінітивного аналізу є ***аналіз ринкового кошика*** (market basket analysis), ціль якого - виявити асоціації між різними подіями, тобто знайти правила для кількісного опису взаємного зв'язку між двома або більше подіями. Такі правила називаються ***асоціативними правилами*** (association rules).

Прикладами застосування асоціативних правил можуть бути наступні задачі:

1) виявлення наборів товарів, які в супермаркетах часто купуються разом або ніколи не купуються разом;
2) визначення частки клієнтів, які позитивно ставляться до нововведень у їхньому обслуговуванні;
3) визначення профілю відвідувачів веб-ресурсу;
4) визначення частки випадків, у яких нові ліки дають небезпечний побічний ефект.

Базовим поняттям у теорії асоціативних правил є ***транзакція*** — деяка множина подій, що відбуваються спільно. Типова транзакція - покупка клієнтом товару в супермаркеті. У переважній більшості випадків клієнт купує не один товар, а набір товарів, що називається ринковим кошиком. При цьому виникає питання: чи є покупка одного товару в кошику наслідком або причиною покупки іншого товару, тобто, чи пов'язані дані події? Цей зв'язок і встановлюють асоціативні правила. Наприклад, може бути виявлене асоціативне правило, котре стверджує, що клієнт, який купив молоко, з імовірністю 75 % купить і хліб.

Наступне важливе поняття — ***предметний набір***. Це непорожня множина предметів, що з'явилися в одній транзакції.

Аналіз ринкового кошика - це аналіз наборів даних для певної комбінації товарів, пов'язаних між собою. Іншими словами, виконується пошук товарів, присутність яких у транзакції впливає на ймовірність наявності інших товарів або комбінацій товарів.

Сучасні касові апарати в супермаркетах дозволяють збирати інформацію про покупки, що може зберігатися в базі даних. Потім накопичені дані можуть використовуватися для побудови систем пошуку асоціативних правил.

У табл. 1.1 представлений простий приклад, що містить дані про ринковий кошик. У кожному рядку вказується комбінація продуктів, придбаних за одну покупку. Хоча на практиці доводиться мати справу з мільйонами транзакцій, у яких беруть участь десятки й сотні різних продуктів, приклад обмежений 10 транзакціями, що містять 13 видів продуктів: аби проілюструвати методику виявлення асоціативних правил, цього досить.

Таблиця 3.1. Приклад набору транзакцій

№	Транзакція
1	Сливи, салат, помідори
2	Селера, цукерки
3	Цукерки
4	Яблука, морква, помідори, картопля, цукерки
5	Яблука, апельсини, салат, цукерки, помідори
6	Персики, апельсини, селера, помідори
7	Квасоля, салат, помідори
8	Апельсини, салат, морква, помідори, цукерки
9	Яблука, банани, сливи, морква, помідори, цибуля, цукерки
10	Яблука, картопля

Візуальний аналіз прикладу показує, що всі чотири транзакції, у яких фігурує салат, також включають помідори, і що чотири із семи покупок, що містять помідори, також містять салат. Салат і помідори в більшості випадків купуються разом. Асоціативні правила дозволяють виявляти й кількісно описувати такі збіги.

Асоціативне правило складається із двох наборів предметів, що мають назву ***умова*** (antecedent) та ***наслідок*** (consequent), й записуються у вигляді $X \to Y$. Таким чином, асоціативне правило формулюється у вигляді: «*Якщо* умова, *то* наслідок».

Умова може обмежуватися тільки одним предметом. Правила звичайно відображаються за допомогою стрілок, спрямованих від умови до наслідку, наприклад, *помідори* → *салат*.

Основними **характеристиками**, що описують асоціативне правило, є ***підтримка*** (support) і ***вірогідність*** (confidence).

Якщо позначити базу даних транзакцій через D, а число транзакцій у цій базі N, то кожна транзакція $d_i, \quad i = 1 \ldots N$ являє собою певний набір предметів. Позначимо підтримку правила через S, а вірогідність – через C.

Підтримка асоціативного правила — це число транзакцій, які містять як умову, так і наслідок. Наприклад, для асоціації $A \to B$ можна записати

$$S(A \to B) = P(A \cap B) = \frac{n(\{A;B\} \in d_i)}{N} \tag{3.1}$$

Вірогідність асоціативного правила $A \to B$ являє собою міру точності правила й визначається як відношення кількості транзакцій, що містять умову і наслідок, до кількості транзакцій, що містять тільки умову:

$$C(A \to B) = P(A \mid B) = \frac{n1(\{A;B\} \in d_i)}{n1(\{A\} \in d_i)} \tag{3.2}$$

Якщо підтримка й вірогідність досить високі, можна з великою ймовірністю стверджувати, що будь-яка майбутня транзакції, що включає умову, буде містити й наслідок.

Для прикладу розглянемо правило *салат* → *помідори*, що напрошується з попередніх спостережень . Для нього

$S(\textit{салат} \rightarrow \textit{помідори}) = 4/10 = 0{,}4;$

$C(\textit{салат} \rightarrow \textit{помідори}) = 4/4 = 1.$

Дане правило, що зустрічається в 40% транзакцій, є для вихідних даних абсолютно вірним – у всіх випадках, коли клієнт купує салат, він разом з тим купує й помідори. Це легко пояснити логічно - обидва продукти використовуються для готування овочевих блюд і дійсно часто купуються разом.

Тепер розглянемо асоціацію *конфеты* → *помидоры***,** у якій містяться слабко сумісні в гастрономічному плані продукти.

Підтримка даної асоціації $S = 4/10 = 0{,}4$ – та ж, що в попереднього правила, а вірогідність $C = 4/6 = 0{,}67$. Таким чином, порівняно невисока вірогідність даної асоціації дає привід засумніватися в тому, що вона є правилом.

Число 0,67 здається не таким вже й малим. Чому ж ми говоримо про «незначну вірогідність» цієї асоціації? Справа в тому, що помідори зустрічаються в 7 чеках з 10 ($P(B) = 0{,}7$ *). Прийнято, що всі правила з вірогідністю меншою, ніж проста ймовірність випадання наслідку не повинні розглядатися, адже вони, по суті, випадкові. Для прийняття асоціативного правила його вірогідність повинна бути не меншою ніж імовірність наслідку.*

Останнє зауваження при досить великій номенклатурі товарів призводить до необхідності подвоювати кількість розрахунків. На практиці аналітики можуть віддавати перевагу правилам, які мають високу підтримку (вище певного рівня, наприклад, 0,3) або високу вірогідність (не менше 0,8-0,85). Висування одночасних вимог щодо підтримки й вірогідності дозволяють значно позм'якшити критерії (підтримку до 0,1-0,15, вірогідність – до 0,67-0,75). Правила, для яких значення підтримки й вірогідності перевищують певні, задані користувачем пороги, називають ***сильними правилами*** (strong rules). Всі наведені вище числові значення - емпіричні. Наприклад, у задачах виявлення шахрайських операцій значення підтримки може знижуватися й до 1%, оскільки із шахрайством пов'язане порівняно невелике число транзакцій.

Як видно з викладеного вище, знайти асоціації не важко. Але як визначити, чи є кожне окреме правило **значимим**, таким, що дозволяє не вгадувати, а прогнозувати наслідок залежно від умови? Наприклад, у кожних 100 аваріях в Росії одним з автомобілів є ВАЗ. Але правило *аварія* → *ВАЗ* , як і зворотне йому, буде малозначимим, оскільки в деяких регіонах ВАЗ складає більше 80% автомобільного парку. Аналогічно в жіночих бутиках, де крім одягу й білизни продаються шоколад і цукерки, вводити правило *білизна* → *шоколад* не раціонально, адже, за статистикою, шоколад і цукерки в них купують 87% відвідувачок, які при цьому можуть не купувати білизни.

Крім об'єктивних оцінок (підтримки й вірогідності) кожного зі згенерованих правил, різні джерела радять використовувати деякі суб'єктивні оцінки. Всі вони, так чи інакше, базуються на об'єктивних.

Ліфт (від interest lift – підвищення інтересу) обчислюється в такий спосіб

$$L(A \rightarrow B) = C(A \rightarrow B) / P(B) \qquad (3.3)$$

Ліфт - це відношення частоти появи умови в транзакціях, які містять й умову, і наслідок, до частоти появи наслідку в цілому. Чим більше значення ліфта, тим частіше наслідок визначається умовою в порівнянні з випадками, коли умова відсутня. Якщо ліфт дорівнює 1, зв'язок відсутній, близькі ж до нуля значення свідчать про сильну зворотну залежність.

Для нашого прикладу в таблиці 3.1. візьмемо два правила з однаковою вірогідністю:

$P(салат) = 4/10 = 0{,}4.\ C(помидоры \to салат) = 4/7 = 0{,}57.$

$P(конфеты) = 6/10 = 0{,}6.\ C(помидоры \to конфеты) = 4/7 = 0{,}57.$

Здавалося б, правила однаково достовірні. Після розрахунку ліфта все стає на свої місця:

$L(помидоры \to салат) = 0{,}57/0{,}4 = 1{,}425;$

$L(помидоры \to конфеты) = 0{,}57/0{,}6 = 0{,}95.$

Не варто вважати ліфт універсальним мірилом адекватності. Справа в тому, що правило з меншою підтримкою й більшим ліфтом може бути менш значимим, ніж альтернативне правило з більшою підтримкою й меншим ліфтом. Це пов'язане з тим, що останнє застосовується для більшого числа покупців.

Левередж (leverage – важіль, плече) – це різниця між спостережуваною частотою, з якої умова й наслідок з'являються спільно, та добутком частот появи умови й наслідку окремо

$$T(A \to B) = S(A \to B) - P(A) \cdot P(B) \qquad (3.4)$$

Левередж дозволяє впоратися із ситуаціями, коли й підтримка, й ліфт у правил ідентичні, але їх значимість явно відрізняється. Наприклад, у нашому овочевому магазині в правила $салат \to помідори$

$C = 1;\ S = 0{,}7;\ L = 1/0{,}7 = 1{,}43.$

Морква, як показує таблиця 1.1, також продається тільки з помідорами, і також зустрічається чотири рази, тому й у правила $морква \to помідори$

$C = 1;\ S = 0{,}7;\ L = 1/0{,}7 = 1{,}43.$

А от левередж у цих правил відрізняється на 30%:

$T(морква \to помідори) = 0{,}3 - 0{,}3 * 0{,}7 = 0{,}09.$

$T(салат \to помідори) = 0{,}4 - 0{,}4 * 0{,}7 = 0{,}12.$

Таким чином, значимість другої асоціації більша, ніж першої.

Альтернативою *левередж* є *поліпшення*.

Поліпшення (improvement) – це відношення частоти спостережуваних виконань правила до добутку частот появи умови й наслідку окремо.

$$I(A \to B) = \frac{S(A \to B)}{P(A) \cdot P(B)} \qquad (3.5)$$

фактично, поліпшення показує, у скільки разів розглянуте правило забезпечує правильний прогноз краще, ніж випадкове вгадування. Всі правила $I(A \to B) \le 1$ не є значимими.

Такі міри, як ліфт, левередж і поліпшення, можуть використовуватися для обмеження набору розглянутих асоціацій шляхом встановлення граничних значень значимості, нижче яких асоціації відкидаються.

3.2 Пошук асоціативних правил

В процесі пошуку асоціативних правил може виконуватися виявлення всіх асоціацій, підтримка й вірогідність для яких перевищують заданий мінімум. Найпростіший алгоритм пошуку асоціативних правил розглядає всі можливі комбінації умов і наслідків, оцінює для них підтримку й вірогідність, а потім виключає всі асоціації, які не задовольняють заданим обмеженням. Число можливих асоціацій зі збільшенням предметів зростає експоненційно. Якщо в базі даних транзакції присутні *k* предметів і всі асоціацій є бінарними (тобто містять по одному предмету в умові й наслідку), потрібно буде проаналізувати $k \cdot 2^{k-1}$ асоціацій. Оскільки реальні бази даних транзакцій, розглянуті при аналізі ринкового кошика, звичайно містять тисячі предметів, обчислювальні витрати при пошуку асоціативних правил величезні**.** Наприклад, якщо аналізувати вибірку, що містить усього 100 предметів**,** то кількість асоціацій, утворених цими предметами, складе $100 \cdot 2^{99} \approx 6{,}4 \cdot 10^{31}$.

Природно, що побудова такої кількості правил і перевірка вірогідності кожного з них - задача непід'ємна. Тому для її вирішення використовуються деякі алгоритмічні прийоми, що дозволяють скорочувати простір можливих рішень. Однією з найпоширеніших є методика, заснована на виявленні так званих частих наборів, коли аналізуються тільки ті асоціації, які зустрічаються досить часто.

На цій концепції заснований відомий алгоритм пошуку асоціативних правил Apriori.

3.3. Ієрархічні асоціативні правила.

Якщо провадити пошук асоціативних правил серед окремих предметів (наприклад, товарів у магазині), то можна виявити, що в багатьох випадках асоціації з високою підтримкою для окремих товарів практично відсутні. Особливо це характерно для супермаркетів, де асортимент товарів кожного виду дуже великий. Наприклад, у продажу можуть бути десятки різновидів таких товарів, як *кетчуп* і *макарони.* Тому, незважаючи на те, що в цілому підтримка асоціації *макарони* → *кетчуп* може бути дуже високою, підтримка асоціацій між окремими видами цих товарів, швидше за все, буде низькою. Отже, такі асоціації, хоча й можуть становити інтерес, виявляться виключеними з розгляду, оскільки не задовольнятимуть деякому мінімальному порогу підтримки $S_{\min}$.

Для вирішення даної проблеми при пошуку асоціативних правил розглядають не окремі предмети, а їх ієрархію. Якщо на нижніх ієрархічних рівнях подібні цікаві асоціації відсутні, то на більш високих вони можуть мати місце. Іншими словами, підтримка окремого предмета завжди буде меншою, ніж підтримка групи, до якої він входить:

$$S(I) > S(i_j),$$

де $I -$ група в ієрархії; i_j - $j-$тий предмет, що входить у дану групу. Причини цього очевидні: загальна підтримка групи дорівнює сумі підтримки включених у неї предметів:

$$S(I) = \sum_{j=1}^{n} i_j ,$$

де N – число предметів у групі.

Розглянемо ієрархію продуктів, представлену на рис. 3.1.

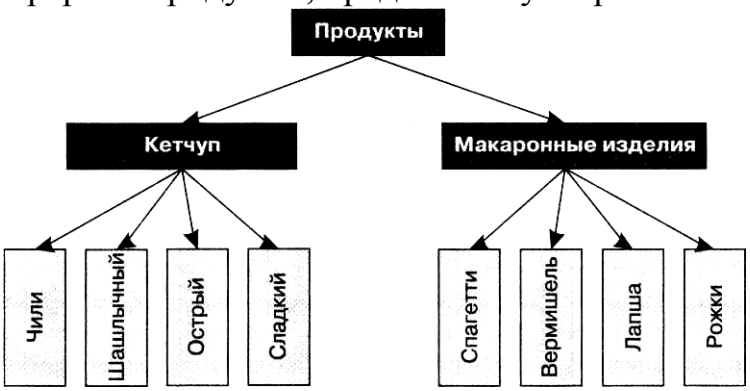

Рис. 3.1. Ієрархія продуктів

У цій ієрархічній схемі кетчуп і макаронні вироби мають безліч підвидів. Найчастіше при обробці касових чеків може бути отримана інформація лише про конкретний товар. Якщо ж структура бази даних транзакцій дозволяє відображати ієрархію товарів, можна досліджувати всі утворені ними ієрархічні рівні. Асоціативні правила, виявлені для предметів, розташованих на різних ієрархічних рівнях, отримали назву *ієрархічні асоціативні правила.* У закордонній літературі вони також відомі як **багаторівневі правила** (multilevel rules) або **узагальнені правила** (generalized rules).

Ієрархія предметів, пов'язаних з комп'ютерною технікою, одна з найбільш складних - вона містить 5 рівнів. Звичайно ієрархічні рівні нумеруються згори донизу, починаючи з нульового. Вузол, що відповідає нульовому рівню, називають **кореневим** (root node). У комп'ютерній техніці перший рівень відповідає всім товарам, другий - категоріям комп'ютерного забезпечення, рівень 3 - конкретним предметам (товарам), рівень 4 - фірмі-виробникові, рівень 5 - конкретній моделі того або іншого товару.

Наприклад, товар (1) - принтер (2) лазерний (3) фірми Xerox (4) модель FX3100 (5). Шукати навіть у величезній базі даних транзакцій його асоціацію з іншим товаром (1) - мишею (2) бездротовою (3) фірми Genius (4) модель KW-09 (5) - украй невдячна річ. Підтримка такого правила навряд чи перевалить за 1%, та й то лише з урахуванням того, що дана конкретна модель миші або принтера мають велику абсолютну популярність.

Універсальним методом для подібних випадків є послідовний спуск за рівнями ієрархії від першого до останнього з виписуванням всіх асоціацій на кожному рівні. Логічно, що на вищих рівнях буде високою й підтримка (наприклад, правил «*ноутбук → миша*»), і вірогідність.

Спускаючись до більш низьких рівнів абстракції, аналізуються нащадки лише тих категорій і підкатегорій, які є частими наборами, тобто зустрічаються не менше наперед заданої кількості разів, де $k-$ номер рівня.

Як на недолік ієрархічного підходу до пошуку асоціативних правил іноді вказують на те, що отримані правила в більшості випадків стосуються не окремих предметів, а їхніх груп, а це не завжди відповідає вимогам аналізу. Однак у деяких предметних областях кількість окремих найменувань може бути настільки значною, що виявлення асоціативних правил навіть на високому рівні узагальнення — це вже успіх. Крім того, технологія бізнес-аналізу в більшій мірі орієнтована на узагальнені дані, тому виробники й тим пак конкретні моделі рідко цікавлять дослідника при побудові асоціацій.

Існує кілька підходів до пошуку ієрархічних асоціативних правил. Найчастіше використовуються спадні методи, коли часті предметні набори досліджуються на кожному ієрархічному рівні, починаючи з першого й закінчуючи рівнем з найбільшою деталізацією. Простіше кажучи, щойно виявляються всі часті предметні набори на першому рівні, починається пошук популярних предметних наборів на другому й т.д. На кожному рівні для відкриття частих наборів може використовуватися будь-який алгоритм, наприклад Apriori та його модифікації. Відомі кілька стратегій ведення пошуку правил.

1. **Використання однакового порога мінімальної підтримки** $S^k_{\min} = const$ *на всіх ієрархічних рівнях.* При пошуку правил одноразово задається певний поріг мінімальної підтримки (наприклад, 5%) при досягненні якого набір вважається частим, у протилежному випадку - не включається до списку правил. Перевага підходу: висока швидкість аналізу предметної області, обумовлена відмовою від оцінки часткових наборів, отриманих з тих з них, які недостатньо часто зустрічаються. Недолік: ризик пропустити тонкі асоціації на нижніх рівнях ієрархії, а саме перевагу користувачів одного виробника або зв'язку певних моделей товарів. Іноді цей недолік намагаються обійти за рахунок зниження мінімального рівня підтримки. Як наслідок - поява десятків і сотень незмістовних правил з низькою підтримкою й вірогідністю.

2. **Зниження порога мінімальної підтримки** при переході до нижніх рівнів ієрархії. Може здійснюватися індивідуально для кожної підгрупи (наприклад, S («Комп'ютери») = 0,1, тоді при розподілі комп'ютерів на категорії «настільний» та «портативний» встановлюється S («Настільний») = 0,06 й S («Портативний») = 0,04) або функціонально.

Функціональний вид зниження порога, як правило, пов'язаний з кількістю підкатегорій або порядковим номером рівня. Наприклад, якщо категорія «Комп'ютери» з рівнем підтримки $S^2 = 0{,}1$ ділиться на дві підкатегорії: «настільний» та «портативний», то в обох підкатегорій $S^3 = S^2 / 2 = 0{,}05$.

Інший варіант функціонального підходу полягає у встановленні порогів мінімальної підтримки винятково в залежності від рівня, скільки б там підкатегорій не було $S^k_{\min} = S^1_{\min} / k$. При такому варіанті функціонального обмеження, якщо, наприклад, для всієї бази даних прийнятий мінімальний рівень підтримки

$S^1_{\min} = 0{,}2$, для категорії «комп'ютери» (2 рівень) він буде 0,2/2=0,1; «настільні комп'ютери» - 0,2/3=0,067; «настільні комп'ютери Dell» - 0,2/4=0,05 і т.д.

Перевага даного підходу очевидна - він дозволяє піти набагато далі по ієрархії в пошуку асоціацій. Однак, у випадку з індивідуальним завданням граничного рівня для кожної підкатегорії, левову частину часу займатиме саме ця процедура. У випадку ж встановлення порогу в залежності від рівня чи кількості підкатегорій не враховуються індивідуальні переваги й недоліки окремих виробників і моделей.

3. **Міжрівнева фільтрація** заснована на так називаному рівні проходу (level passage). Відносно високий для верхніх рівнів ієрархії граничний рівень підтримки зберігається на нижніх в першому проході. Потім, у кожному проході, цей рівень знижується. Ті товари низького рівня, які мають асоціації, таким чином, ідентифікуються раніше, ніж їхні батьківські категорії, які, можливо, і не будуть мати необхідної підтримки. Рекомендується виконувати три-чотири проходи бази даних транзакцій для сполучення рівнів довіри представників різних рівнів.

3.4. Послідовні шаблони.

Все викладене раніше не дає відповіді на питання, як пов’язати асоціації із клієнтом, а також встановити часові залежності, тобто – не тільки відповісти на запитання, ЩО купує певний клієнт, а ЩО ЗА ЧИМ він купує. І якщо є можливість - спрогнозувати, що ж він купить НАСТУПНОГО РАЗУ.

Відповіді на дане питання дає використання **послідовних шаблонів** (sequential patterns), заснованих на теорії асоціацій, обов'язковими полями яких є дата/час та ідентифікатор клієнта. Останні можуть бути отримані при розрахунку кредитними картами, клієнтськими картками або будь-якими іншими ідентифікаторами - законними або не дуже.

При розгляді послідовностей транзакцій використовується одне припущення - той самий клієнт у той самий момент не здійснює двох різних транзакцій.

Послідовність — це впорядкований список предметних наборів (товарів або послуг, замовлених або оплачених ідентифікованим клієнтом в ідентифікований момент часу).

Послідовність S називається максимальною, якщо вона не міститься в будь-якій іншій послідовності.

Послідовність S називається клієнтською, якщо вона, крім набору предметів, дати й часу, містить також ідентифікатор клієнта.

Послідовність S_1 міститься в послідовності S_2, якщо всі предметні набори S_1 містяться в предметних наборах S_2. Наприклад, послідовність <(3); (4, 5); (8)> міститься в послідовності <(7); (3, 8); (9); (4, 5, 6); (8)>, оскільки (3) $\subseteq$ (3, 8), (4, 5) $\subseteq$ (4, 5, 6) і (8) $\subseteq$ (8). Однак <(3); (5)> $\not\subset$ <(3, 5)>) і навпаки, оскільки в першій послідовності предмети 3 й 5 були куплені один за іншим, а в другій — разом.

Послідовність S називається підтримуваною клієнтом, якщо вона міститься в його клієнтській послідовності. Тоді підтримка послідовності визначається як число клієнтів, що її підтримують, та зазвичай виражається у відсотках від загального числа клієнтів. Таким чином, поняття підтримки для послідовних шаблонів дещо відрізняється від аналогічного поняття для асоціативних правил.

Для бази даних клієнтських транзакцій задача пошуку послідовних шаблонів полягає у виявленні максимальних послідовностей серед всіх, що мають підтримку вище заданого порога. Кожна така максимальна послідовність і буде послідовним шаблоном. Далі будемо називати послідовності, що задовольняють обмеженню мінімальної підтримки, частими (по аналогії із наборами, що часто зустрічаються, у теорії асоціативних правил).

Процес пошуку послідовних шаблонів складається з наступних кроків:

1. **Сортування**. Транзакції вихідної бази даних сортуються за кодом клієнтів, а транзакції кожного клієнта – за датою, часом і номером візиту. Результат - база даних клієнтських послідовностей.

2. **Пошук частих предметних наборів** F. Частими називаються предметні набори, які купувалися числом клієнтів, більшим від мінімально-граничного значення. Обрана множина частих предметних наборів переводиться до числового або символьного представлення.

3. **Перетворення**. Необхідно визначити, які з частих послідовностей містяться в клієнтській послідовності. Для цього кожна транзакція клієнтської послідовності заміщується множиною її частих предметних наборів. Якщо в транзакції немає жодного частого набору, вона більше не розглядається. Більш того, якщо в певного клієнта в послідовності немає жодного частого набору, він теж виключається з розгляду. Після перетворення кожна клієнтська послідовність - це впорядкована множина частих наборів.

4. **Пошук частих послідовностей**. На множині частих предметних наборів виконується пошук частих послідовностей. Мінімальний рівень частоти - параметр алгоритму.

5. **Пошук максимальних послідовностей**. Серед частих послідовностей відшукуються максимальні. Іноді даний етап об’єднують із попереднім, аби зменшити витрати часу на обчислення не максимальних послідовностей.

Найбільш проблемним етапом пошуку послідовних шаблонів є виявлення частих послідовностей, оскільки велика кількість предметних наборів вимагає розгляду величезної кількості можливих комбінацій і багаторазового проходу по набору транзакцій. Кожен прохід починається з вихідного набору послідовностей, що використовується для генерації нових потенційних частих послідовностей, які мають назву послідовності-кандидати, або просто кандидати. Для цього обчислюється їх підтримка та по завершенні проходу визначається, чи є виявлені кандидати в дійсності частими. Виявлені часті послідовності-кандидати стануть вихідними для нового проходу.

Приклад. *Після етапу 3 з урахуванням виключення клієнтів були отримані клієнтські послідовності*

Клієнт	**Послідовність**
1	**< {1,5}; {2}; {3}; {4} >**
3	**< {1}; {3}; {4}; {3, 5} >**
5	**< {1}; {2}; {3}; {4} >**
6	**< {1}; {3}; {5}; {4} >**
8	**< {4}; {5} >**

Пошук частих послідовностей виконується від рівня 1 до максимального можливого. Результати послідовних проходів представимо в таблиці

1-послідовність		**2-послідовність**		**3-послідовність**		**4-послідовність**	
F_1	**Підтримка**	F_2	**Підтримка**	F_3	**Підтримка**	F_4	**Підтримка**
<1>	**4**	**<1; 2>**	**2**	**<1; 2:3>**	**2**	**<1; 2; 3; 4>**	**2**
<2>	**2**	**<1; 3>**	**4**	**<1; 2:4>**	**2**		
<3>	**4**	**<1; 4>**	**3**	**<1; 3; 4>**	**3**		
<4>	**4**	**<1; 5>**	**3**	**<1; 3; 5>**	**2**		
<5>	**4**	**<2; 3>**	**2**	**<2; 3:4>**	**2**		
		<2; 4>	**2**				
		<3; 4>	**3**				
		<3; 5>	**2**				
		<4; 5>	**2**				

Таким чином, максимальними послідовностями є <1; 2; 3; 4>, <1; 3; 5> й <4; 5>, оскільки вони не містяться в послідовностях більшої довжини. Вони й будуть шуканими послідовними шаблонами.

4. Відновлення інформації та прогнозування

Сучасні наукові та практичні дослідження базуються на обробці поточної та ретроспективної інформації. Від того, наскільки вона є якісною (інформативною, точною, достовірною тощо), залежить точність результатів. Особливий інтерес становить ситуація, коли частина даних відсутня. Це можливо через відмову обладнання, втрати інформації з технічних причин, а також суб'єктивних обставин.

Задача відновлення пропусків має декілька варіантів постановки, що визначається структурою пропущених даних. Зокрема, пропуски можуть бути серед значень вхідних факторів, результуючих характеристик, вхідних факторів і результуючих характеристик, а також серед значень ознак певного об'єкта, де такі фактори та характеристики явно не виділені.

Методи, якими розв'язуються задачі відновлення пропусків у даних, теж мають свою класифікацію. Розглядають методи, що базуються на елементарних обчисленнях, статистичні методи, імовірнісні методи, нейромережні, еволюційні методи. Визначаючи, який метод застосовувати, необхідно знати особливості та обмеження його використання. Так, методи, що базуються на елементарних обчисленнях, раціонально застосовувати тоді, коли кількість пропусків є незначною. Одержані оцінки, найчастіше, є зміщеними.

Статистичні методи застосовують, якщо передбачається існування лінійної залежності між вхідними факторами та результуючими характеристиками. Необхідність апріорних знань про імовірнісні характеристики визначає аспекти застосування імовірнісних методів.

Нейромережні методи, у загальному випадку, дозволяють обробляти різні структури пропусків, але точність одержаних оцінок визначатиметься інформативністю та повнотою даних для навчання нейромереж, а також їх архітектурою та законами функціонування. Аспекти застосування нейромереж розглядались раніше.

Оскільки задача відновлення пропусків має оптимізаційний характер, то для її розв'язання запропоновано використовувати еволюційні методи, які інтегрують у собі нейромережну ідентифікацію та генетичну оптимізацію. Результати експериментів засвідчили порівняно високу точність результатів. Недоліком є необхідність використання значних обчислювальних ресурсів.

4.1. Математична постановка задачі відновлення пропусків у таблицях даних

У загальній постановці задача відновлення пропусків у таблицях даних має наступну постановку. Припустімо, $X=(X_1,X_2,...,X_n)$ - вектор вхідних факторів, $Y=(Y_1,Y_2,...,Y_m)$ - вектор результуючих характеристик, p - кількість експериментів або періодів ретроспективи, $A_{p\times n+m}=(a_{ij})$ - матриця вихідної інформації. Вона має пропуски, які позначені зірочками (приклад – у табл. 4.1).

Припустимо, що між вхідними факторами і результуючими показниками існують залежності

$$\widehat{Y}_i = F_i(X_1, X_2, ..., X_n),\ i = \overline{1,m} \tag{4.1}$$

Тоді задача відновлення пропусків у даних полягає в знаходженні

$$\min_{*} \|Y - F(X)\|, \tag{4.2}$$

де $F = (F_1, F_2, ..., F_m)$ і $Y = (Y_1, Y_2, ..., Y_m)$ - вектори значень, що одержані за ідентифікованими залежностями та наведені у вихідній таблиці, відповідно.

Таблиця 4.1. Структура вихідної інформації

№ з/п	X_1	X_2	X_2	...	X_n	Y_1	Y_2	...	Y_m
1	x_1^1	x_2^1	x_2^1	...	x_n^1	y_1^1	y_2^1	...	y_m^1
2	x_1^2	x_2^2	*	...	x_n^2	y_1^2	y_2^2	...	*
3	x_1^3	*	x_3^3		x_n^3	*	y_2^3	...	y_m^3
...	...	...		...	...	...	...	...	...
p	x_1^p	x_2^p	x_3^p	...	x_n^p	y_1^p	y_2^p	...	y_m^p

Задачу (4.2) деталізуємо і перепишемо у вигляді

$$\min_{*} \frac{1}{pm} \sum_{i=1}^{p} \sum_{j=1}^{m} \left(Y_{ij} - F_j\left(X_1^i, X_2^i, ..., X_n^i\right)\right)^2 \tag{4.3}$$

або, використовуючи загальне позначення елементів матриці

$$\min_{*} \frac{1}{pm} \sum_{i=1}^{p} \sum_{j=1}^{m} \left(a_{ij} - \widehat{Y}_{ij}\right)^2 \tag{4.4}$$

Якщо припустити, що залежності (4.1) лінійні, тобто

$$\widehat{Y}_i = b_{i0} + b_{i1}X_1 + b_{i2}X_2 + ... + b_{in}X_n, \tag{4.5}$$

тоді задача відновлення пропусків зводиться до знаходження

$$\min_{*} \|Y - BX\|, \tag{4.6}$$

де $Y = \left(a_{ij}\right)_{i=1,\, j=n+1}^{p,\, n+m}$, $B = \left(b_{ij}\right)_{i=1,\, j=1}^{m,\, n+1}$ та $X = \left(a_{ij}\right)_{i=1,\, j=1}^{p,\, n}$

Розв'язання задач (4.2)-(4.6) має перший етап, який, у загальному випадку, полягає в ідентифікації залежностей F_i, $i = \overline{1,m}$. Зауважимо, що в задачі відновлення пропусків у таблицях даних процедури ідентифікації та оптимізації ітеративно повторюються.

7.2. Евристичні методи обробки некомплектних даних

Наведемо та виконаємо аналіз методів відновлення втрачених даних, що застосовуються найчастіше. Матриця, рядок та стовпчик, що мають пропуски даних, називаються некомплектними.

Метод заповнення середнім значенням.

Згідно із цим методом відсутнє значення на перетині $i-$того рядка та $j-$того стовпчика розраховується за формулою:

$$a_{ij}^{*}=\frac{1}{q}\sum_{\substack{i=1\\ a_{ij}\neq *}}^{p}a_{ij} \tag{4.7}$$

де q - кількість заповнених елементів у $j-$тому стовпчику.

Перевагою методу є простота.

Недоліки: в одному стовпчикові може бути велика кількість пропусків і всі вони будуть заповненні однаковими значеннями; не враховується зв'язок некомплектного рядка з іншими рядками, що веде до зміщеної і недостовірної оцінки невідомого значення. Цей висновок справедливий і для випадку здійснення перерахунку з урахуванням кожного заповненого пропуску.

Метод виключення некомплектних рядків.

Застосовується у випадку незначної кількості пропусків.

Метод є простим, але вилучення даних збільшує ентропію прогнозних значень та веде до зміщеності параметрів моделі.

Метод підстановки.

Метод має декілька модифікацій. Розглянемо одну із них. Припустимо, що на перетині $i-$того рядка та $j-$того стовпчика знаходиться відсутнє значення. Тоді, серед усіх інших рядків вибираємо ті, у яких лише у $j-$тому стовпчикові пропуск, або рядки, які є некомплектними. Знаходимо їх відстань до цільового рядка за формулою:

$$d_k=\sqrt{\sum_{l=1,l\neq j}^{n+m}\left(a_{kl}-a_{il}\right)^2},\ k=\overline{1,Z} \tag{4.8}$$

де Z - кількість рядків, що визначаються вищезазначеною умовою. Впорядковуємо значення d_k за спаданням і задаємо деяке порогове значення $D>0$. Серед усіх d_k, $k=\overline{1,Z}$, обираємо перші h рядків, для яких виконується умова $d_k>D$. Лише за значеннями елементів цих рядків знаходимо пропуск

$$a_{ij}^{*}=\frac{\sum_{l=1}^{h}\frac{a_{lj}}{1+d_{li}}}{\sum_{l=1}^{h}\frac{1}{1+d_{li}}}. \tag{4.9}$$

Ідея методу базується на гіпотезі існування залежностей між факторами, що, найчастіше, не відповідає дійсності. Значні обчислювальні затрати зменшують і так низьку ефективність методу, оскільки для адекватних обчислень відстаней між рядками дані необхідно нормувати.

Метод множинної лінійної регресії.

Застосовується у припущені, що залежність (4.1) є лінійною. Для її ідентифікації використовують лише комплектні дані і з використанням МНК одержують (4.5). Очевидно, що надалі (4.5) використовуються для відновлення пропусків, але адекватно це можна робити лише у випадку одного пропуску серед значень рядка $X_1^i,X_2^i,...,X_n^i,Y_i$. Якщо таких пропусків два і більше, то задача

розв'язується за додаткових припущень та обмежень. Метод вимагає виконання ряду застережень та перевірок вхідних факторів на мультиколінеарність, гетероскедастичність, автокореляцію та застосування модифікованих версій МНК.

Метод множинної нелінійної регресії.

На відміну від лінійної регресії, алгоритм нелінійної множинної регресії застосовують лише у випадку пропуску значень вихідної характеристики. недоліки та застереження – ті ж, що й для методу множинної лінійної регресії.

4.3. Відновлення пропусків значень залежної змінної

Розглянемо випадок, коли проводиться активний експеримент і значення факторів $X=(X_1,X_2,...,X_n)$ задані дослідником, а Y - залежна від цих факторів змінна. Очевидно, що тоді пропуски серед значень вхідних факторів містяться набагато рідше, ніж серед значень вихідної характеристики.

Метод Бартлетта заповнення пропусків.

Зробимо припущення про те, що пропущені значення є лише серед значень результуючої характеристики Y і рядки, які їм відповідають, знаходяться вгорі таблиці вихідних даних. Кожний пропуск y_i, $i=\overline{1,m_0}$, заповнимо початковими значеннями $\tilde{y}_i$. Побудуємо матрицю Z супутніх значень змінних пропусків. За визначенням $i-$та супутня змінна пропусків є індикатором $i-$того пропущеного значення, тобто завжди є нулем, за виключенням випадку, якщо пропущено $i-$те значення (тоді вона дорівнює одиниці). Перший рядок матриці Z матиме вигляд $z_1=(1,0,0,...,0)$, відповідно m_0-тий рядок $z_{m_0}=(0,0,...,0,1)$. Рядки, починаючи з m_0+1 до $n-$того, дорівнюють (0,0,...,0). Таким чином, маємо вираз

$$\begin{pmatrix} x_{11} & x_{12} & ... & x_{1p} \\ x_{21} & x_{22} & ... & x_{2p} \\ ... & ... & ... & ... \\ x_{n1} & x_{n2} & ... & x_{np} \end{pmatrix}\begin{pmatrix} \beta_1 \\ \beta_2 \\ ... \\ \beta_p \end{pmatrix}+\begin{pmatrix} 1 & 0 & 0 & ... & 0 \\ 0 & 1 & 0 & ... & 0 \\ ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & 1 \\ 0 & 0 & 0 & ... & 0 \\ ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & 0 \end{pmatrix}\begin{pmatrix} \gamma_1 \\ \gamma_2 \\ ... \\ \gamma_{m_0} \\ ... \\ \gamma_p \end{pmatrix}+\begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ ... \\ \varepsilon_n \end{pmatrix}=\begin{pmatrix} \tilde{y}_1 \\ \tilde{y}_2 \\ ... \\ \tilde{y}_{m_0} \\ y_{m_0+1} \\ ... \\ y_n \end{pmatrix}$$

або у матричному вигляді

$$Y=X\beta+Z\gamma+\varepsilon. \qquad (4.10)$$

В моделі (4.10) $\varepsilon=(\varepsilon_1,\varepsilon_2,...,\varepsilon_n)^T$ - вектор залишків, які є незалежними, однаково розподіленими, з нульовим середнім і однаковою дисперсією; β - оцінюваний параметр - вектор довжиною p.

Вважаючи, що виконуються всі передумови застосування МНК, класична (вважаючи, що Y є комплектним) оцінка β є такою

$$\beta^*=(X^TX)^{-1}X^TY. \qquad (4.11)$$

Для нашої задачі (4.10) згідно із МНК цільова функція матиме вигляд

$$\Psi(\beta,\gamma)=\sum_{i=1}^{m_0}(\tilde{y}_i - x_i\beta - z_i\gamma)^2 + \sum_{i=m_0+1}^{n}(y_i - x_i\beta - z_i\gamma)^2 . \quad (4.12)$$

Необхідно знайти $\min\limits_{\beta,\gamma}\Psi(\beta,\gamma)$.

Використовуючи визначення матриці Z із (4.12), одержимо

$$\Psi(\beta,\gamma)=\sum_{i=1}^{m_0}(\tilde{y}_i - x_i\beta - z_i\gamma)^2 + \sum_{i=m_0+1}^{n}(y_i - x_i\beta)^2 . \quad (4.13)$$

Припустимо, що $\hat{\beta}^*$ - оцінка, одержана за МНК за формулою (4.11) для існуючих значень Y, тобто за останніми $m=n-m_0$ рядками. Вона мінімізує другу суму у виразі (4.13). Якщо при $\beta=\hat{\beta}^*$ покласти вектор доповнень $\hat{\gamma}=(\hat{\gamma}_1,\hat{\gamma}_2,..,\hat{\gamma}_{m_0})^T$, у якому $\hat{\gamma}_i=\tilde{y}_i - x_i\hat{\beta}^*$, для $i=\overline{1,m_0}$, то перша сума в (4.13) не тільки мінімізується, а й перетворюється в нуль. Інакше кажучи, мінімізувати необхідно функцію

$$\Psi(\beta,\gamma)=\sum_{i=m_0+1}^{n}(y_i - x_i\hat{\beta}^*)^2 .$$

Таким чином, сполучення векторів $\hat{\beta}^*$ та $\hat{\gamma}$ мінімізує $\Psi(\beta,\gamma)$ і є оцінкою МНК, одержаною з моделі (4.10). Рівняння (4.13) означає також, що точна оцінка МНК відсутнього значення y_i, має вигляд $\hat{\gamma}_i=x_i\hat{\beta}^*$. Таким чином прогноз $i-$того пропущеного значення за МНК є початковим значенням для $i-$того пропуску мінус коефіцієнт для супутньої змінної і -го пропуску $y_i^*=\tilde{y}_i-\hat{\gamma}_i$.

Відомі два способи ініціалізації пропущених значень: за першим із них вони дорівнюють нулю, за другим - є середнім відомих значень. До переваг методу належить його неітеративність. Якщо структура пропусків є виродженою, то результат відсутній.

Метод Resampling

Метод відновлення пропусків resampling є різновидом відомого методу „bootstrap", запропонованого американським статистиком Бредлі Ефроном. Його сутність полягає у багатократній обробці різних частин одних і тих самих даних, що дозволяє здійснювати їх різносторонній аналіз, та співставляти одержані результати.

Припустимо, дані та пропуски мають ту ж структуру як і для методу Бартлетта. Застосування методу resampling може бути виконано двома способами.

Алгоритм Resampling-1.

Крок 1. Формуємо матрицю повних спостережень $H=(X_1,X_2,...,X_n,Y)$, кількість рядків в якій m_0.

Крок 2. Випадковим чином вибираємо $j-$тий рядок із матриці H, та замінюємо $i-$тий рядок початкової матриці, i належить тій частині матриці, що не містить пропусків $i \in (m_0+1;n)$. Цей рядок може вибиратись випадково або по порядку, починаючи з m_0+1-того.

Крок 3. Якщо всі пропуски заповнені, то за МНК знаходимо коефіцієнти регресійного рівняння β_k, $k=\overline{0,p}$, в іншому випадку виконуємо перехід на крок 2.

Крок 4. Якщо одержано r векторів $\beta^q=(\beta_0^q,\beta_1^q,...,\beta_p^q)$, $q=\overline{1,r}$, то знаходимо середні значення коефіцієнтів регресійної моделі $\hat{\beta}_k=\frac{1}{r}\sum_{q=1}^{r}\beta_k^q$, $k=\overline{0,p}$.

Алгоритм Resampling-2.

Крок 1. За матрицею H будується регресійна модель і знаходяться оцінки коефіцієнтів $\hat{\beta}_i$, $i=\overline{0,p}$.

Крок 2. Розраховуємо оцінку $\hat{Y}_i$ за регресійною моделлю для $i=\overline{1,m_0}$.

Крок 3. Знаходимо помилки $\varepsilon_i=\left|Y_i-\hat{Y}_i\right|$, $i=\overline{1,m_0}$.

Крок 4. Для кожного пропуску, підставляючи значення супутніх змінних $X_1,X_2,...,X_n$ в одержане регресійне рівняння, знаходимо оцінку $\hat{Y}_i$, $i=\overline{m_0+1,n}$.

Крок 5. Значення, якими замінюють пропуски, одержують за формулою:

$$Y_i=\hat{Y}_i+\varepsilon_i,\ i=\overline{m_0+1,n},$$

де ε_i обирають випадковим чином із результатів кроку 3.

Крок 6. За даними, одержаними після заповнення, будуємо регресійну модель і повторно знаходимо оцінки коефіцієнтів $\hat{\beta}_i$, $i=\overline{0,p}$.

Крок 7. Аналогічний кроку 4 із алгоритму resampling-l.

У запропонованого методу resampling перевагою є повне використання вихідної інформації, водночас її повторне використання зменшує інформативність даних.

4.4. Локальні методи відновлення пропусків

Алгоритм ZET

Сутність алгоритму полягає у тому, що кожне пропущене значення оцінюють по «компетентній» матриці, яка складається із визначеного числа рядків і стовпчиків вихідної матриці. Пропущене значення у рядку знаходять, використовуючи обчислення відстаней, у стовпчику - обчислення коефіцієнтів кореля-

ції. Остаточну оцінку знаходять, усереднюючи попередні оцінки з ваговими коефіцієнтами, значення яких визначаються певними параметрами.

Крок 1. Виконаємо нормування значень матриці A за формулою $a'_{ij} = \dfrac{a_{ij} - \overline{a}_j}{\sigma_j}$, де $\overline{a}_j$ - середнє значення $j-$того стовпчика, σ_j - середньоквадратичне відхилення для комплектних даних.

Крок 2. Припустимо, що $a_{ij} = *$. Задамо коефіцієнт впливу компетентності на результат прогнозування α.

Крок 3. Знаходимо відстані від усіх комплектних рядків матриці A до $i-$того рядка за формулою:

$$d_{ki} = \sqrt{\sum_{l=1, l\neq j}^{n+m} (a_{kl} - a_{il})^2}\ ,\ k = \overline{1, p}\ ,\ k \neq i \qquad (4.14)$$

Крок 4. Визначаємо q рядків, для яких d_{ki} має найменші значення, і з них та $i-$того рядка формуємо матрицю A_q (порядок розміщення рядків зберігається, $i-$тий рядок стає $ii-$тим).

Крок 5. Знаходимо модулі коефіцієнтів кореляції усіх стовпчиків матриці A_q з $j-$тим стовпчиком:

$$r_{jk} = \frac{\frac{1}{q}\sum_{l=1}^{q}(a_{lj} - \overline{a}_j)(a_{lk} - \overline{a}_k)}{\sqrt{\frac{1}{q}\sum_{l=1}^{q}(a_{lj} - \overline{a}_j)^2 \frac{1}{q}\sum_{l=1}^{q}(a_{lk} - \overline{a}_k)^2}},\ k = \overline{1, n+m}\ ,\ k \neq j \qquad (4.14)$$

Крок 6. Визначимо v стовпчиків, для яких значення r_{jk} має найменші значення, і з них та $j-$того стовпчика формуємо «компетентну» матрицю A_{qv}, при цьому порядок розміщення стовпчиків зберігається, а $j-$тий стовпчик стає $jj-$тим. Таким чином, $a_{ii,jj} = *$.

Крок 7. Обчислюємо «компетентності» рядків k_l^p, $l = \overline{1, q}$, як величини обернено пропорційні відстаням до рядка, що містить відсутнє значення $k_l^p = \dfrac{1}{1 + d_{l,ii}}$

Крок 8. Обчислюємо «компетентності» стовпчиків k_l^c, $l = \overline{1, v}$, як величини прямо пропорційні (або рівні) модулям r_{jk}.

Крок 9. Для кожного $i-$того рядка, $i = \overline{1, q}$, та $ii-$того рядка згідно із методом МНК знаходимо рівняння парної лінійної регресії $a_{ii} = k_i a_i + b_i$ і, прирівнюючи $a_i = a_{i,jj}$, знаходимо оцінку пропущеного значення $\hat{a}_i^p$, $i = \overline{1, q}$.

Крок 10. Аналогічний кроку 9, але для стовпчиків знаходимо оцінку $\hat{a}_i^c$, $i=\overline{1,v}$.

Крок 11. Для компетентної матриці знаходимо прогнозні величини для рядка і стовпчика:

$$\hat{a}^p=\frac{\sum_{l=1}^{q}\left(\hat{a}_l^p\cdot\left(k_l^p\right)^{\alpha}\right)}{\sum_{l=1}^{q}\left(k_l^p\right)^{\alpha}};\qquad \hat{a}^c=\frac{\sum_{l=1}^{v}\left(\hat{a}_l^c\cdot\left(k_l^c\right)^{\alpha}\right)}{\sum_{l=1}^{v}\left(k_l^c\right)^{\alpha}}$$

Крок 12. Обчислення пропущеного значення $a_{ij}^*=\left(\hat{a}^p+\hat{a}^c\right)/2$.

Крок 13. Закінчення алгоритму.

Алгоритм ZET застосовується у припущенні про виконання 3-х гіпотез: надмірності, аналогічності та локальної компетентності, згідно з якими припускають, що у таблицях даних є подібні рядки та залежні стовпчики; якщо пара об'єктів має подібні значення $(n-1)$-того параметра, то і значення $n-$того параметра подібні; немає сенсу використовувати для заповнення відсутнього пропуску усіх рядків і стовпчиків матриці, а достатньо брати лише їх „компетентну" частину.

До недоліків алгоритму ZET можна віднести певний „волюнтаризм" дослідника, який полягає у суб'єктивному визначенні розмірів „компетентної" матриці та коефіцієнта „компетентності", що впливає на присутність шумового ефекту при передбаченні пропущеного значення та точність обчислень результату.

Алгоритм ZETBraid.

На відміну від алгоритму ZET, в ZETBraid реалізована ідея поступового додавання в «компетентну» матрицю рядків і стовпчиків. Сутність алгоритму полягає у специфічному підрахунку відстаней між рядками та стовпчиками.

Відстань між рядками обчислюємо за формулою $r_{ij}=\sum_{k=1}^{n}b_k\left(a_{ik}-a_{jk}\right)^2$, де b_k - ваговий коефіцієнт, значення якого залежить від того, чи входить $i-$тий стовпчик в «компетентну» матрицю.

При обчисленні коефіцієнтів b_k, $k=\overline{1,n}$, дотримуються трьох принципів:

1) Всі вагові коефіцієнти стовпчиків, що входять в «компетентну» матрицю, рівні.

2) Всі вагові коефіцієнти стовпчиків, що не входять в «компетентну» матрицю, рівні.

3) Сума вагових коефіцієнтів стовпчиків, що входять в «компетентну» матрицю, поділена на суму вагових коефіцієнтів, що не входять в «компетентну» матрицю, є константою C (параметр алгоритму).

Якщо із n стовпчиків p належать «компетентній» матриці, то ваговий коефіцієнт стовпчика $w=1$, якщо стовпчик не належить "компетентній" матриці, та $w=1+C\left(\frac{n}{p}-1\right)$ в іншому випадку.

Для знаходження відстані між стовпчиками необхідно будувати рівняння лінійної регресії. Нехай $X=(x_1,x_2,...,x_n)$ та $Y=(y_1,y_2,...,y_m)$ - стовпчики, тоді потрібно одержати рівняння загального вигляду $y=Ax+B$.

Для знаходження коефіцієнтів A та B мінімізуємо функцію

$$D(A,B)=\sum_{i=1}^{m} b_i\left(y_i-Ax_i-B\right)^2, \qquad (4.16)$$

де вагові коефіцієнти рядків b_i, $i=\overline{1,m}$, знаходяться аналогічно ваговим коефіцієнтам стовпчиків, тобто $w=1$, якщо рядок не належить "компетентній" матриці, та $w=1+C\left(\frac{m}{q}-1\right)$ в іншому випадку, де q - кількість рядків, що належать «компетентній» матриці.

Важливою задачею залишається підбір розміру «компетентної» матриці. Критерієм оцінки адекватності цієї матриці є оцінка якості передбачення невідомого елемента. Таким чином, при побудові «компетентної» матриці рядки і стовпчики додаються до тих пір, поки значення критерію (абсолютного відхилення точного і прогнозованого значення) зменшується.

Відомими є два варіанти розрахунку цього критерію. Згідно із першим, методом «хреста», за рівнянням лінійної регресії розраховують всі відомі значення рядка і (або) стовпчика, що містять невідомий елемент і знаходять середню помилку. Ця середня помилка і є оцінкою передбачення даної «компетентної» матриці.

Другий варіант, дисперсійний метод, полягає в обчисленні дисперсії передбачень невідомого елемента. Для цього за рівнянням лінійної регресії для кожного стовпчика прогнозують значення невідомого елемента та знаходять дисперсію цих $n-1$ прогнозів, яка і є шуканою оцінкою.

«Компетентна» матриця за побудовою є квадратною. В крайньому випадку кількість рядків та стовпчиків можуть відрізнятися на одиницю. Для того, аби розмірність матриці могла бути довільною, але адекватною, виконаємо такі кроки:

Крок 1. Знаходимо рядок, найбільш близький до цільового рядка, що не входить до «компетентної» матриці. Якщо додавання цього рядка не погіршує її оцінку, то додаємо його до "компетентної" матриці.

Крок 2. Аналогічний кроку 1 для стовпчика.

Кроки 1 і 2 повторюють до моменту погіршення оцінки «компетентної» матриці як для рядка, так і для стовпчика.

Для того, аби уникнути помилок при початковій побудові «компетентної» матриці, на перших k кроках (зазвичай, $k=6$, тобто до утворення матриці

3×3) додають рядки і стовпчики в «компетентну» матрицю, не зважаючи на її оцінку.

4.5. Ітераційний метод головних компонент для даних з пропусками

Методи, що представляють даний напрям, розроблені вченими Красноярської школи нейроматематики під керівництвом професора Горбаня О.М. Їх головна ідея полягає в тому, що набір точок, який є многовидом (об'єкт, який локально має характер метричного простору розмірності n) за наявності пропусків, дозволяє будувати лінійні і нелінійні наближення - моделі, за допомогою яких відновлюють пропущені значення. Результати алгоритмізації цих методів і експериментальних перевірок засвідчили достатньо високу точність. Проведені дослідження вказують на задовільне функціонування алгоритмів при 10-15% пропусків. У той же час, математичні викладки базуються на достатньо сильних припущеннях про розподіл вхідних даних, гладкості функцій та обумовленості матриці початкових значень. До недоліків слід також віднести складність реалізації і верифікації алгоритму.

Припустимо, наявна прямокутна матриця, чарунки якої містять дійсні числа або символ @ в разі, якщо дані відсутні. Задача – найбільш достовірно відновити відсутні дані. При більш детальному розгляді задача поділяється на три підзадачі:

- заповнити пропуски у таблиці;
- відредагувати таблицю – змінити значення відомих даних так, щоб найкращим чином працювали моделі, що використовуються при відновленні пропущених даних;
- побудувати на основі таблиці обчислювальну процедуру, яка заповнюватиме пропуски у поточному рядку даних в припущенні, що дані в цьому рядку зв’язані тими ж співвідношеннями, що і дані в таблиці.

Для розв’язання цього сімейства задач пропонується використовувати метод послідовного наближення множини векторів даних (рядків таблиці) прямими. Основна процедура – пошук найкращого наближення таблиці з пропусками $A=\left(a_{ij}\right)$ матрицею вигляду $x_i k_j + b_j$. Найкраще наближення A матрицею $x_i k_j + b_j$ будемо шукати за МНК. Критерій прийме вигляд

$$\Phi = \sum_{\substack{i,i \\ a_{ij}\neq @}} \left(a_{ij} - x_i k_j - b_j\right)^2 \to \min \qquad (4.17)$$

Якщо два з трьох векторів x_i, k_j або b_j фіксовані, то третій знаходиться за явними формулами. Однак, це не є необхідним, адже задаючись практично довільним початковим наближенням для двох векторів, знаходять значення третього. Потім «відомі» вектори міняються і розраховується інший вектор, і так по колу до досягнення певної наперед заданої точності. Збіжність ітераційного процесу доведена авторами.

Для прискорення ітераційного процесу при фіксованому значенні x_i (а так найчастіше є на практиці) можна одночасно обчислити перші наближення векторів k_j та b_j.

Наведемо розрахункові вирази для кожного з випадків.

Отже, якщо попередньо задатися значеннями векторів k_j та b_j, то x_i знаходиться з умови $\frac{\partial \Phi}{\partial x_i} = 0$, а саме

$$x_i = \left(\sum_{\substack{j \\ a_{ij} \neq @}} (a_{ij} - b_j) k_j \right) \Bigg/ \left(\sum_{\substack{j \\ a_{ij} \neq @}} (k_j)^2 \right). \tag{4.18}$$

При фіксованому значенні x_i значення k_j та b_j, що мінімізують (4.17), визначаються з одночасного виконання двох рівностей $\frac{\partial \Phi}{\partial k_j} = 0$ та $\frac{\partial \Phi}{\partial b_j} = 0$.

Для кожного j маємо систему рівнянь відносно k_j та b_j

$$\begin{cases} k_j A_{01}^j + b_j A_{00}^j = B_0^j \\ k_j A_{11}^j + b_j A_{10}^j = B_1^j \end{cases}, \text{ де } A_{kl}^j = \sum_{\substack{i \\ a_{ij} \neq @}} x_i^{k+l};\ B_k^j = \sum_{\substack{i \\ a_{ij} \neq @}} a_{ij} x_i^k;\ k = 0,1;\ l = 0,1.$$

Методом підстановки неважко отримати вирази для невідомих

$$k_j = \frac{B_1^j - B_0^j A_{10}^j / A_{00}^j}{A_{11}^j - A_{01}^j A_{10}^j / A_{00}^j}; \qquad b_j = \frac{B_0^j - k_j A_{01}^j}{A_{00}^j}. \tag{4.19}$$

Слід звернути велику увагу на вибір початкових значень. Вектор k_j має на першому кроці приймати випадкові значення, нормовані щодо одиниці (тобто $\sum_j k_j^2 = 1$). Водночас $b_j = \frac{1}{n_j} \sum_{\substack{i \\ a_{ij} \neq @}} a_{ij}$, де n_j - кількість відомих даних у стовпчику j, тобто є середнім значенням відомих елементів стовпчика.

Критерієм зупинки ітераційного алгоритму може слугувати досягнення якогось певного рівня зміни цільової функції $\Delta\Phi / \Phi < \varepsilon$, або ж досягнення певного ступеня зменшення самої цільової функції $\Phi < \delta$. У загальному випадку зупинка розрахунків виконується за будь-якою з умов, необхідно лише задати достатньо малі значення $\varepsilon, \delta > 0$.

Послідовне вичерпання матриці A досягається за рахунок того, що для даної матриці A шукаємо найкраще наближення матрицею P_1 вигляду $x_i k_j + b_j$. Далі для $A - P_1$ знаходимо найкраще наближення такого ж виду P_2 і т.д. Контроль ведеться, наприклад, за залишковою дисперсією стовпчиків.

$Q-$факторне заповнення пропусків полягає в тому, що пропущені елементи визначення із суми Q отриманих матриць виду $x_i k_j + b_j$, $Q-$факторний «ремонт» таблиці - заміна її на суму Q отриманих матриць виду $x_i k_j + b_j$.

Нехай в результаті описаного процесу побудована послідовність матриць P_q виду $x_i k_j + b_j$ $\left(P_q = x_i^q k_j^q + b_j^q\right)$, що вичерпує початкову матрицю A із заданою точністю. Опишемо операцію відновлення даних в рядку a_j з пропусками, що поступає на обробку (в ньому певні $a_{ij} = @$).

Для кожного q по заданому рядку визначимо число $x^q(a)$ та вектор a_j^q:

$$a_j^0 = a_j \text{ для } a_j \neq @;$$

$$x^1(a) = \left(\sum_{\substack{j \\ a_j \neq @}} \left(a_j^0 - b_j^1\right) k_j^1 \right) \Bigg/ \left(\sum_{\substack{j \\ a_j \neq @}} \left(k_j^1\right)^2 \right);$$

$$a_j^1 = a_j^0 - b_j^1 - x^1(a) k_j^1 \text{ для } a_j \neq @;$$

.....

$$x^q(a) = \left(\sum_{\substack{j \\ a_j \neq @}} \left(a_j^{q-1} - b_j^q\right) k_j^q \right) \Bigg/ \left(\sum_{\substack{j \\ a_j \neq @}} \left(k_j^q\right)^2 \right);$$

$$a_j^q = a_j^{q-1} - b_j^q - x^q(a) k_j^q \text{ для } a_j \neq @;$$

Для $Q-$факторного відновлення даних достатньо обчислити суму

$$a_j^* = \sum_{q=1}^{Q} b_j^q + x^q(a) k_j^q \text{ для } a_j \neq @; \qquad (4.20)$$

Якщо пропуски відсутні, то описаний метод приводить до звичайних головних компонентів - сингулярного розкладу початкової таблиці даних, у цьому випадку, починаючи з $q = 2$, $P_q = x_i^q k_j^q$, а $b_j^q = 0$. В загальному випадку це не так і центрування для даних з пропусками є непридатним.

Також слід врахувати, що за відсутності пропусків отримані прямі будуть ортогональними, тобто отримаємо ортогональну систему факторів (прямих). Виходячи з цього, при неповних даних можливий процес ортогоналізації отриманої системи факторів, який полягає в тому, що початкова таблиця відновлюється за допомогою отриманої системи, після чого ця система перераховується заново, але вже при повних даних.

4.6. EM-алгоритм

Алгоритм EM базується на максимізації математичного сподівання, чим і обґрунтовується його назва - expectation-maximization approach, яку він одержав у 1977 році. В алгоритмі реалізована ітераційна процедура обчислення оцінки максимальної правдоподібності, а складається він з двох кроків:

Крок 1. Множина даних неповної задачі і поточне значення вектора параметрів використовуються для одержання розширеного повного набору даних.

Крок 2. Обчислюється нова оцінка вектора параметрів шляхом максимізації функції логарифмічної правдоподібності повної множини даних.

Кроки 1 і 2 повторюються до повної збіжності.

Наведемо математичний запис алгоритму EM. Нехай r - повний вектор даних, яким він повинен був би бути, але не є; $d = d(r)$ - вектор неповних даних, який фактично спостерігається.

Таким чином, вектори r та d є елементами відповідних просторів: $r \in R$, $d \in D$. Позначимо $f_c(r/\theta)$ - функцію щільності умовної ймовірності r для даного вектора параметрів θ. Звідси випливає, що функція щільності умовної ймовірності випадкової змінної d для даного вектора θ визначається так:

$$f_c(d/\theta) = \int_{R(d)} f_c(r/\theta) dr$$

де $R(d)$ - підпростір R, що визначається рівністю $d = d(r)$.

Необхідно знайти значення θ, що максимізує функцію логарифмічної правдоподібності на неповних даних:

$$L(\theta) = \log f_D(d/\theta).$$

Ця задача розв'язується ітеративним застосуванням функції логарифмічної правдоподібності на повних даних $L_c(\theta) = \log f_c(r/\theta)$, яка є випадковою змінною, оскільки відсутні дані є невідомими.

Якщо припустити, що $\hat{\theta}(n)$ - значення вектора параметрів на $n-$тій ітерації, то на першому її кроці знаходимо математичне сподівання

$$Q(\theta, \hat{\theta}(n)) = M(L_c(\theta))$$

по $\hat{\theta}(n)$. На другому кроці обчислюємо максимум функції $Q(\theta, \hat{\theta}(n))$ по θ в просторі параметрів, таким чином, щоб знайти оцінку вектора $\hat{\theta}(n+1)$:

$$\hat{\theta}(n+1) = \arg\max_{\theta} Q(\theta, \hat{\theta}(n+1)).$$

Алгоритм починається із того, що задається деяке початкове значення $\hat{\theta}(0)$ вектора параметрів θ. Кроки алгоритму ітеративно повторюються, поки різниця між $L(\hat{\theta}(n+1))$ та $L(\hat{\theta}(n))$ не стане меншою деякого малого наперед заданого значення. Тоді робота алгоритму завершується.

Для розуміння математичних викладок і нюансів алгоритму EM бажано вільно володіти матеріалом з теорії ймовірностей та математичної статистики.

4.7. Еволюційний метод відновлення пропусків

Припустимо, що пропуски є лише серед значень вхідних факторів $X=(X_1,X_2,...,X_n)$, результуюча характеристика Y одна та існує залежність

$$Y=F(X)=F(X_1,X_2,...,X_n) \tag{4.21}$$

А.М. Колмогоров та В.І. Арнольд довели теорему про те, що кожна неперервна функція n змінних, задана на одиничному кубі $n-$вимірного простору, може бути представленою у вигляді

$$f(x_1,x_2,...,x_n)=\sum_{q=1}^{2n+1}h_q\left[\sum_{p=1}^{n}\varphi_q^p(x_p)\right],$$

де функції $h_q(u)$ неперервні, а функції $\varphi_q^p(x_p)$, крім того, ще і стандартні, тобто не залежать від вибору функції f.

В термінах теорії нейронних мереж ця теорема вказує на те, що будь-яка неперервна функція ідентифікується мережею з одним, як мінімум, прихованим шаром нейронів з нелінійними функціями активації. Для ідентифікації (4.21) в якості моделі візьмемо прямозв'язну нейромережу з пороговим алгоритмом зворотного поширення похибки. Надалі структура мережі та її елементний базис в експериментах залишаються постійними.

Оскільки вхідні образи для навчання нейронної мережі мають пропуски значень, то необхідно розв'язати задачу параметричної оптимізації. Як метод оптимізації запропоновано використати генетичний алгоритм. Для гарантування його збіжності використаємо теорему, доведену Р. Харті (R.E. Harti). Якщо використовувати бінарне представлення розв'язків і для формування їх популяції - елітний відбір, то теорема вказує на збіжність ГА за ймовірністю.

Для роботи ГА необхідно сформувати генеральну та вибіркову сукупність хромосом-розв'язків. Хромосома складається з фрагментів, які відповідають пропускам в таблиці даних: $Xr=\langle miss_1;miss_2,...,miss_z\rangle$, де z - кількість пропусків даних.

Дані в таблиці нормуємо без врахування пропущених значень. Якщо активаційною функцією буде вибрано гіперболічний тангенс, то нормування раціональніше здійснювати у відрізок [-1;1]. Кількість хромосом в генеральній сукупності визначається заданою точністю результату, у вибірковій - дослідником.

На наступному кроці формуємо навчальну та контрольну послідовність для навчання нейромережі. Пропонується всі образи з пропусками вважати елементами навчальної послідовності. Для контрольної послідовності їх використання є проблематичним, оскільки неможливо застосовувати для верифікації недостовірні значення. Співвідношення потужності множини образів навчальної та контрольної послідовності може бути різним, на що впливає співвідношення кількості образів з пропусками й без пропусків у початковій таблиці.

Алгоритм відновлення пропусків (EvoGap) **буде таким:**

Крок 1. Ініціалізація $K_{\max}$ хромосом-розв'язків вибіркової послідовності.

Крок **2.** $K=1$**.**

Крок 3. Навчання нейромережі на точках навчальної послідовності, де значення пропусків заповнені значеннями $K-$тої хромосоми. При цьому розв'язується задача пошуку

$$M_K = \min_W \frac{1}{2} \sum_{i=1}^{p_{train}} \left(\dot{a}_{i,n+1} - a_{i,n+1} \right)^2 \tag{4.22}$$

де W - матриця вагових коефіцієнтів нейромережі, p_{train} - кількість образів в навчальній послідовності.

Крок 4. Обчислення цільової функції ГА (fitness-function):

$$G_K = \frac{1}{2} \sum_{i=1}^{p_{contr}} \left(\dot{a}_{i,n+1} - a_{i,n+1} \right)^2 \tag{4.23}$$

де p_{contr} - кількість образів контрольної послідовності. Якщо $G_K < G_{\min}$ (заданої наперед точності визначення пропусків), то перехід на крок 7.

Крок 5. $K = K + 1$. Якщо $K > K_{\max}$, де $K_{\max}$ - кількість елементів у вибірковій послідовності, то перехід на крок 6, інакше перехід на крок 3.

Крок 6. Виконання процедур кросоверу, мутації, визначення і відбір хромосом наступної епохи. Перехід на крок 2.

Крок 7. Закінчення алгоритму.

Еволюційний метод відновлення пропусків в даних має ряд переваг. Так, його використання не вимагає виконання обмежень на вихідну інформацію. Таблиця початкових даних може мати довільну розмірність і структуру пропусків. Перспективним є дослідження ефективності використання нейромережі з неітеративними алгоритмами навчання. Необхідно з'ясувати вплив розподілу значень факторів на точність відновлення пропусків. Додаткові дослідження у вказаних напрямках дозволять сформувати методику відновлення пропусків з використанням еволюційного підходу.