МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
ІНСТИТУТ ЦИФРОВИХ ТЕХНОЛОГІЙ, ДИЗАЙНУ ТА ТРАНСПОРТУ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЄКТУВАННЯ ТА ДИЗАЙНУ

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт
з дисципліни «Інтелектуальні методи підтримки прийняття рішень»
для студентів усіх форм навчання

Другий (магістерський) рівень вищої освіти
Спеціальність – 122 КОМП'ЮТЕРНІ НАУКИ
Освітня програма – КОМП'ЮТЕРНИЙ ДИЗАЙН

Одеса – 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ОДЕСЬКА ПОЛІТЕХНІКА»
ІНСТИТУТ ЦИФРОВИХ ТЕХНОЛОГІЙ, ДИЗАЙНУ ТА ТРАНСПОРТУ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЄКТУВАННЯ ТА ДИЗАЙНУ

МЕТОДИЧНІ ВКАЗІВКИ

до виконання лабораторних робіт
з дисципліни «Інтелектуальні методи підтримки прийняття рішень»
для студентів усіх форм навчання

Другий (магістерський) рівень вищої освіти
Спеціальність – 122 КОМП'ЮТЕРНІ НАУКИ
Освітня програма – КОМП'ЮТЕРНИЙ ДИЗАЙН

Затверджено на засіданні
кафедри інформаційних технологій
проєктування та дизайну
Протокол № 1 від 27.08.2025

Одеса – 2025

Методичні вказівки до виконання лабораторних робіт з дисципліни «Інтелектуальні методи підтримки прийняття рішень» для студентів другого (магістерського) рівня за спеціальністю 122 Комп'ютерні науки, за освітньо-професійною програмою «Комп'ютерний дизайн» / Укл.: Л.В. Бовнегра, К.Г. Кіркопуло, А.С. Коляда – Одеса : Національний університет «Одеська політехніка», 2025. – 63 с.

Укладачі:
Л.В. Бовнегра, к.т.н., доц.
К.Г. Кіркопуло, PhD, доц.
А.С. Коляда, к.т.н., доц.

Методичні вказівки до виконання лабораторних робіт з дисципліни «Інтелектуальні методи підтримки прийняття рішень» призначені для студентів другого (магістерського) рівня вищої освіти за спеціальністю 122 «Комп'ютерні науки», освітньо-професійної програми «Комп'ютерний дизайн». Метою методичних вказівок є формування у здобувачів вищої освіти практичних навичок застосування інтелектуальних методів підтримки прийняття рішень під час аналізу складних слабоструктурованих задач у соціально-економічних, технічних та інформаційних системах. Матеріали спрямовані на поєднання теоретичних засад із практичною реалізацією моделей і алгоритмів прийняття рішень із використанням сучасних інструментів аналізу даних. У методичних вказівках розглядаються лабораторні роботи, присвячені базовим і прикладним аспектам систем підтримки прийняття рішень, зокрема багатокритеріальному аналізу альтернатив (методи SAW, AHP, TOPSIS), побудові експертних систем на основі логічних правил, застосуванню методів кластеризації та класифікації, прогнозуванню на основі часових рядів, аналізу та візуалізації великих даних, а також оцінюванню ризиків і вразливостей у СППР. Окрема увага приділяється питанням інтерпретованості моделей, сценарному аналізу та обґрунтуванню управлінських рішень. Кожна лабораторна робота містить мету, короткі теоретичні положення, завдання до виконання, стислий опис способу обробки результатів, детальний приклад виконання, домашнє завдання, рекомендації щодо оформлення звіту та перелік літературних джерел. Методичні вказівки орієнтовані на розвиток аналітичного мислення, навичок роботи з даними та сучасних підходів до підтримки прийняття рішень і можуть бути корисними для підготовки фахівців у галузі інформаційних технологій, аналітики даних, цифрового управління та проєктування інтелектуальних систем.

ЗМІСТ

Лабораторна робота №1. Вступ до систем підтримки прийняття рішень та аналіз простих рішень

Мета роботи: ознайомлення із концепцією систем підтримки прийняття рішень, методами багатокритеріального аналізу рішень та освоєнням практичних навичок використання Excel/Google Sheets для вибору оптимального варіанту.

Короткі теоретичні положення

Системи підтримки прийняття рішень (СППР)

Системи підтримки прийняття рішень (СППР) – це інтерактивні програмні засоби, які допомагають особам або організаціям у прийнятті складних рішень шляхом аналізу великих обсягів даних, використання моделей та алгоритмів для обґрунтування оптимального вибору. Вони широко застосовуються в економіці, управлінні, медицині, логістиці та інших сферах, де необхідно швидко обробляти інформацію та отримувати аналітичні висновки.

СППР можуть містити такі основні компоненти:

- База даних – сховище інформації, необхідної для прийняття рішень. Вона може включати історичні дані, актуальні показники та прогнозовані значення.
- Модуль аналізу – набір моделей та алгоритмів для обробки інформації. Цей модуль може містити статистичні методи, алгоритми машинного навчання або методи багатокритеріального аналізу.
- Інтерфейс користувача – засоби взаємодії з системою, що дозволяють отримувати результати аналізу в зручному форматі. Інтерфейс може бути текстовим або графічним, включати інтерактивні елементи для модифікації критеріїв аналізу.

Методи прийняття рішень

Одним із найпоширеніших підходів у СППР є багатокритеріальний аналіз рішень (MCDA – Multi-Criteria Decision Analysis). Він включає декілька методів оцінки альтернатив, які дозволяють приймати рішення з урахуванням багатьох факторів.

1. Метод зваженої суми критеріїв (SAW)

Метод зваженої суми критеріїв (SAW – Simple Additive Weighting) передбачає оцінку альтернатив за кількома критеріями шляхом нормалізації значень і розрахунку загального рейтингу.

Формула для обчислення загальної оцінки альтернативи A_i:

$$S_i = \sum w_j \cdot X_{ij}$$

де:

- S_i – загальна оцінка альтернативи,
- w_j – вага критерію (сума ваг повинна дорівнювати 1),
- X_{ij} – нормалізоване значення альтернативи за критерієм j.

Нормалізація може виконуватися за такими формулами:

- Для критеріїв, де більше значення краще:

$$X_{ij} = \frac{X_{ij}}{X_{\max}}$$

- Для критеріїв, де менше значення краще:

$$X_{ij} = \frac{X_{\min}}{X_{ij}}$$

Цей метод є одним із найпростіших для реалізації, але він чутливий до вибору вагових коефіцієнтів, що може впливати на кінцевий результат.

2. Метод аналізу ієрархій (AHP – Analytic Hierarchy Process)

Метод AHP базується на побудові ієрархії критеріїв та їх попарному порівнянні.

1. Формування матриці попарних порівнянь
   Оцінка кожного критерію відносно іншого записується у вигляді квадратної матриці:

$$A = \begin{bmatrix} 1 & a_{12} & a_{13} & \dots & a_{1n} \\ \frac{1}{a_{12}} & 1 & a_{23} & \dots & a_{2n} \\ \frac{1}{a_{13}} & \frac{1}{a_{23}} & 1 & \dots & a_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ \frac{1}{a_{1n}} & \frac{1}{a_{2n}} & \frac{1}{a_{3n}} & \dots & 1 \end{bmatrix}$$

2. Обчислення вагових коефіцієнтів критеріїв
   Нормалізація рядків матриці та усереднення значень у кожному стовпці.
3. Перевірка узгодженості матриці
   Розрахунок індексу узгодженості (CI):

$$CI = \frac{\lambda_{\max} - n}{n - 1}$$

де $\lambda_{\max}$ – найбільше власне значення матриці, а n – розмірність матриці.

Якщо відношення $CR = \frac{CI}{RI}$ (де RI – випадковий індекс) менше 0.1, то матриця узгоджена.

3. Метод TOPSIS (Technique for Order Preference by Similarity to Ideal Solution)

Метод TOPSIS полягає у визначенні альтернативи, яка знаходиться найближче до позитивного ідеального розв'язку та найдалі від негативного ідеального розв'язку.

1. Нормалізація матриці рішень
   Використовується формула:

$$X_{ij}^{\text{norm}} = \frac{X_{ij}}{\sqrt{\sum_{i=1}^{m} X_{ij}^2}}$$

2. Формування зваженої нормалізованої матриці
   Елементи множаться на вагові коефіцієнти:
   $V_{ij} = w_j \cdot X_{ij}^{\text{norm}}$
3. Обчислення ідеальних рішень
   Визначення найкращих (A^+) та найгірших (A^-) альтернатив.
4. Обчислення відстаней до ідеальних рішень
   Евклідова відстань до позитивного та негативного ідеалу:

$$D_i^+ = \sqrt{\sum_{j=1}^{n} (V_{ij} - A_j^+)^2}$$

$$D_i^- = \sqrt{\sum_{j=1}^{n} (V_{ij} - A_j^-)^2}$$

5. Обчислення відносної близькості до ідеального рішення

$$C_i = \frac{D_i^-}{D_i^+ + D_i^-}$$

Чим ближче значення C_i до 1, тим кращою є альтернатива.

Кожен із цих методів має свої переваги та обмеження, тому вибір конкретного підходу залежить від специфіки завдання та доступних даних. Використання кількох методів паралельно може дати більш обґрунтовані результати та підвищити точність прийнятих рішень.

Завдання до роботи

1. Сформувати предметну область прийняття рішення, обравши реалістичну прикладну задачу, наприклад:
   - вибір постачальника товарів або послуг;
   - вибір програмного продукту;
   - вибір транспортного маршруту;
   - вибір обладнання або технічного рішення.
2. Визначити множину альтернатив (не менше 3 та не більше 5), які реально можуть конкурувати між собою в межах обраної задачі.
3. Сформувати систему критеріїв оцінювання (3–4 критерії), що:

- мають різну природу (вартісні, якісні, часові тощо);
- містять як критерії виграшу (чим більше — тим краще), так і критерії витрат (чим менше — тим краще).

4. Побудувати початкову матрицю рішень, заповнивши її конкретними числовими значеннями для кожної альтернативи.
5. Виконати багатокритеріальний аналіз альтернатив трьома методами:
   - методом зваженої суми критеріїв (SAW);
   - методом аналізу ієрархій (AHP);
   - методом TOPSIS.
6. Здійснити нормалізацію даних, обґрунтувати вибір формул нормалізації та вагових коефіцієнтів.
7. Порівняти результати, отримані різними методами, та:
   - виявити збіги й розбіжності у ранжуванні альтернатив;
   - пояснити причини можливих відмінностей.
8. Провести елементарний аналіз чутливості, змінюючи ваги критеріїв і оцінюючи вплив цих змін на кінцевий результат.
9. Зробити обґрунтований висновок щодо вибору оптимальної альтернативи та доцільності використання того чи іншого методу.

Опис способу обробки результатів, хід експерименту

Крок 1: Створення таблиці даних

1. Визначити альтернативи (наприклад, постачальники A, B, C, D).
2. Визначити критерії оцінки, наприклад:
   - Ціна (в грошових одиницях, чим нижче, тим краще).
   - Якість (оціночна шкала, наприклад, від 1 до 10, чим вище, тим краще).
   - Швидкість доставки (у днях, чим менше, тим краще).
   - Рейтинг відгуків (загальна оцінка на основі відгуків клієнтів, наприклад, від 1 до 5).
3. Заповнити таблицю вихідних значень, яка включатиме дані для кожної альтернативи відповідно до обраних критеріїв.

Крок 2: Метод зваженої суми критеріїв (SAW)

1. Нормалізувати значення критеріїв:
   - Для критеріїв, де більше значення краще: $X_{ij} = \frac{X_{ij}}{X_{max}}$
   - Для критеріїв, де менше значення краще: $X_{ij} = \frac{X_{min}}{X_{ij}}$
2. Призначити вагові коефіцієнти.
3. Розрахувати підсумкову оцінку альтернативи: $$S_i = \sum_{j=1}^{n} w_j \cdot X_{ij}$$
4. Визначити альтернативу з найвищим рейтингом.

Крок 3: Метод аналізу ієрархій (AHP)

1. Побудувати матрицю попарних порівнянь для критеріїв:

$$A = \begin{bmatrix} 1 & a_{12} & a_{13} & \dots & a_{1n} \\ \frac{1}{a_{12}} & 1 & a_{23} & \dots & a_{2n} \\ \frac{1}{a_{13}} & \frac{1}{a_{23}} & 1 & \dots & a_{3n} \\ \dots & \dots & \dots & \dots & \dots \\ \frac{1}{a_{1n}} & \frac{1}{a_{2n}} & \frac{1}{a_{3n}} & \dots & 1 \end{bmatrix}$$

2. Розрахувати вагові коефіцієнти.
3. Обчислити індекс узгодженості (CI): $$CI = \frac{\lambda_{\max} - n}{n - 1}$$
4. Переконатися, що рівень узгодженості CR < 0.1.
5. Обчислити фінальні оцінки альтернатив на основі розрахованих ваг критеріїв.

Крок 4: Метод TOPSIS

$$X_{ij}^{\text{norm}} = \frac{X_{ij}}{\sqrt{\sum_{i=1}^{m} X_{ij}^2}}$$

1. Нормалізувати матрицю рішень:
2. Формувати зважену нормалізовану матрицю: $V_{ij} = w_j \cdot X_{ij}^{\text{norm}}$
3. Обчислити ідеальні рішення (найкращі та найгірші альтернативи).
4. Обчислити відстань до ідеальних рішень:

$$D_i^+ = \sqrt{\sum_{j=1}^{n} (V_{ij} - A_j^+)^2}$$

$$D_i^- = \sqrt{\sum_{j=1}^{n} (V_{ij} - A_j^-)^2}$$

5. Обчислити відносну близькість до ідеального рішення:

$$C_i = \frac{D_i^-}{D_i^+ + D_i^-}$$

6. Обрати альтернативу з найбільшим значенням C_i.

Крок 5: Аналіз та візуалізація результатів

1. Побудувати графік, що показує порівняння оцінок альтернатив за різними методами.
2. Порівняти отримані результати за SAW, AHP та TOPSIS.
3. Виконати аналіз чутливості, змінюючи вагові коефіцієнти.
4. Запропонувати рекомендації щодо вибору альтернативи на основі отриманих результатів.

Приклад виконання

Таблиця вихідних даних

Наведена таблиця містить початкові дані для аналізу альтернатив. Вона включає ключові показники, що впливають на вибір, такі як ціна, якість, швидкість доставки та рейтинг.

Альтернатива	Ціна (грн)	Якість (1-10)	Доставка (днів)	Рейтинг (1-5)
A	5000	8	2	4.5
B	4500	7	3	4.2
C	5200	9	1	4.8
D	4800	6	2	4.3

Метод зваженої суми критеріїв (SAW)

Нормалізація значень:

Альтернатива	Норм. Ціна	Норм. Якість	Норм. Доставка	Норм. Рейтинг
A	0.87	0.89	0.67	0.94
B	1.00	0.78	0.50	0.88
C	0.87	1.00	1.00	1.00
D	0.94	0.67	0.67	0.90

Розрахунок підсумкової оцінки:

Припустимо, що ваги критеріїв визначені наступним чином:

- Ціна: 0.4
- Якість: 0.3
- Доставка: 0.2
- Рейтинг: 0.1

$$S_A = (0.87 \times 0.4) + (0.89 \times 0.3) + (0.67 \times 0.2) + (0.94 \times 0.1) = 0.84$$
$$S_B = (1.00 \times 0.4) + (0.78 \times 0.3) + (0.50 \times 0.2) + (0.88 \times 0.1) = 0.79$$
$$S_C = (0.87 \times 0.4) + (1.00 \times 0.3) + (1.00 \times 0.2) + (1.00 \times 0.1) = 0.91$$
$$S_D = (0.94 \times 0.4) + (0.67 \times 0.3) + (0.67 \times 0.2) + (0.90 \times 0.1) = 0.80$$

Оптимальна альтернатива (SAW): C (0.91)

Метод аналізу ієрархій (AHP)

Метод **AHP (Analytic Hierarchy Process, Аналіз ієрархій)** розроблений Томасом Сааті і використовується для прийняття рішень, що враховують **кілька критеріїв**. Він складається з таких етапів:

1. Формування ієрархії завдання

Рішення має ієрархічну структуру:

1. **Ціль (Вибір найкращої альтернативи)**
2. **Критерії оцінки альтернатив**
3. **Альтернативи для вибору**

Наприклад, при виборі постачальника критеріями можуть бути:

- **Ціна**
- **Якість**
- **Доставка**
- **Рейтинг**

Альтернативи: **A, B, C, D** (постачальники).

2. Побудова матриці попарних порівнянь критеріїв

Створюємо **матрицю попарних порівнянь** для критеріїв.

Значення у матриці визначаються **шкалою Сааті (Saaty Scale)**:

Значення	Інтерпретація
1	Однакове значення критеріїв
3	Помірна перевага одного критерію над іншим
5	Сильна перевага
7	Дуже сильна перевага
9	Абсолютна перевага
2,4,6,8	Проміжні значення

Правило:

- Якщо значення a_{ij} вказує, що критерій i важливіший за критерій j, то $a_{ji} = \frac{1}{a_{ij}}$.

Приклад матриці порівнянь критеріїв:

Критерій	Ціна	Якість	Доставка	Рейтинг
Ціна	1	3	5	7
Якість	1/3	1	3	5
Доставка	1/5	1/3	1	3
Рейтинг	1/7	1/5	1/3	1

3. Нормалізація матриці порівнянь критеріїв

Кожен елемент **ділимо на суму відповідного стовпця**, щоб отримати нормалізовані значення.

Обчислення суми кожного стовпця:

$$\sum_i A_{ij}$$

Критерій	Ціна	Якість	Доставка	Рейтинг
Сума	1.52	4.53	9.33	16.00

Ділимо кожне значення в стовпці на його суму:

$$A_{ij}^{\text{norm}} = \frac{A_{ij}}{\sum A_{ij}}$$

Нормалізована матриця:

Критерій	Ціна	Якість	Доставка	Рейтинг
Ціна	0.658	0.662	0.536	0.437
Якість	0.219	0.221	0.321	0.313

Критерій	Ціна	Якість	Доставка	Рейтинг
Доставка	0.146	0.073	0.107	0.188
Рейтинг	0.073	0.044	0.036	0.063

4. Обчислення вагових коефіцієнтів критеріїв
Розраховуємо середнє значення в кожному рядку:

$$w_i = \frac{1}{n} \sum_{j=1}^{n} A_{ij}^{\text{norm}}$$

Критерій	Вага (Wi)
Ціна	0.573
Якість	0.269
Доставка	0.129
Рейтинг	0.079

Таблиця значень RI для різної кількості критеріїв (n):

n	1	2	3	4	5	6	7	8	9	10
RI	0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49

Якщо CR<0.1, то матриця вважається **узгодженою**.
Формула для розрахунку $\lambda_{\max}$

$$\lambda_{\max} = \frac{1}{n} \sum_{i=1}^{n} \frac{(A \cdot W)_i}{W_i}$$

де:

- A – матриця попарних порівнянь,
- W – вектор вагових коефіцієнтів,
- n – кількість критеріїв,
- $(A \cdot W)_i$ – добуток матриці на вектор ваг.

Обчислюємо добуток $A \cdot W$:

$$A \cdot W = \begin{bmatrix} 1 & 3 & 5 & 7 \\ 1/3 & 1 & 3 & 5 \\ 1/5 & 1/3 & 1 & 3 \\ 1/7 & 1/5 & 1/3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.573 \\ 0.269 \\ 0.129 \\ 0.079 \end{bmatrix} = \begin{bmatrix} (1 \times 0.573) + (3 \times 0.269) + (5 \times 0.129) + (7 \times 0.079) \\ (1/3 \times 0.573) + (1 \times 0.269) + (3 \times 0.129) + (5 \times 0.079) \\ (1/5 \times 0.573) + (1/3 \times 0.269) + (1 \times 0.129) + (3 \times 0.079) \\ (1/7 \times 0.573) + (1/5 \times 0.269) + (1/3 \times 0.129) + (1 \times 0.079) \end{bmatrix} = \begin{bmatrix} 2.355 \\ 1.099 \\ 0.491 \\ 0.229 \end{bmatrix}$$

Обчислення $\lambda_{\max}$
Кожен елемент результату ділимо на відповідне значення W:

$$\lambda_{\max} = \frac{(2.355/0.573) + (1.099/0.269) + (0.491/0.129) + (0.229/0.079)}{4}$$

$$\lambda_{\max} = \frac{4.222 + 4.174 + 4.036 + 4.040}{4} = 4.118$$

Обчислення CI

$$CI = \frac{\lambda_{\max} - n}{n - 1} = \frac{4.118 - 4}{4 - 1} = \frac{0.118}{3} = 0.039$$

Обчислення CR Значення RI для n=4 (із таблиці): 0.90.

$$CR = \frac{CI}{RI} = \frac{0.039}{0.90} = 0.043$$

Якщо **CR>0.1**, це означає, що матриця попарних порівнянь **не є узгодженою**, тобто експертні оцінки мають суперечності.
Можливі помилки:

- Чи правильно застосовано **шкалу Сааті** (1, 3, 5, 7, 9)?
- Чи дотримано **обернене значення** (Aij=1/Aji)?
- Чи **не помилково введені оцінки** (наприклад, випадково обрані 5 замість 3)?

5. Побудова матриць попарних порівнянь альтернатив

Для кожного критерію будуємо **матрицю попарних порівнянь альтернатив**, використовуючи ту ж саму шкалу Сааті.

Приклад для критерію "Ціна":

Альтернатива	A	B	C	D
A	1	2	3	4
B	1/2	1	2	3
C	1/3	1/2	1	2
D	1/4	1/3	1/2	1

Повторюємо це для кожного критерію.

6. Нормалізація та розрахунок ваг альтернатив

Для кожного критерію:

1. **Нормалізуємо матрицю**, ділячи кожен елемент на суму відповідного стовпця.
2. **Обчислюємо середнє значення по рядках** – це **ваги альтернатив за критерієм**.

Приклад ваг альтернатив за критерієм "Ціна":

Альтернатива	Вага (W)
A	0.44
B	0.31
C	0.16
D	0.09

7. Розрахунок підсумкових оцінок альтернатив

Формула:

$$S_i = \sum_{j=1}^{m} w_j \cdot X_{ij}$$

де:

- w_j – ваги критеріїв,
- X_{ij} – ваги альтернатив за критерієм.

Розрахунок для альтернативи A:

$$S_A = (0.573 \times 0.44) + (0.269 \times 0.35) + (0.129 \times 0.30) + (0.079 \times 0.20)$$

$$S_A = 0.252 + 0.094 + 0.039 + 0.016 = 0.401$$

Повторюємо для всіх альтернатив.

Підсумкові оцінки:

Альтернатива	Оцінка (S)
A	0.401
B	0.367
C	0.295
D	0.217

8. Вибір оптимальної альтернативи

Альтернатива з **найбільшим значенням** є **найкращим вибором**.

Оптимальний вибір: Альтернатива A (0.401)

Метод TOPSIS

Метод **TOPSIS** (Техніка впорядкування за подібністю до ідеального рішення) використовується для **ранжування альтернатив**, вибираючи **найкращий варіант, що найближчий до ідеального**

розв'язку і найдалі від найгіршого.
Метод ґрунтується на таких припущеннях:
- Найкраще рішення має **мінімізувати негативні критерії (наприклад, вартість)**
- Найкраще рішення має **максимізувати позитивні критерії (наприклад, якість, рейтинг)**

1. Формування таблиці альтернатив і критеріїв
Як і в методі AHP, потрібно визначити:

- **Ціль**: вибір оптимальної альтернативи
- **Критерії оцінки**: наприклад, **Ціна, Якість, Доставка, Рейтинг**
- **Альтернативи**: A, B, C, D

Припустимо, що **початкові дані** виглядають так:

Альтернатива	Ціна (грн)	Якість (1-10)	Доставка (днів)	Рейтинг (1-5)
A	5000	8	2	4.5
B	4500	7	3	4.2
C	5200	9	1	4.8
D	4800	6	2	4.3

Припустимо, що критерії мають такі ваги:

- **Ціна** (менше – краще): **0.4**
- **Якість** (більше – краще): **0.3**
- **Доставка** (менше – краще): **0.2**
- **Рейтинг** (більше – краще): **0.1**

2. Нормалізація матриці рішень
Метод TOPSIS вимагає **нормалізації даних**, щоб всі критерії були на одному масштабі.
Формула нормалізації для кожного елемента:

$$X_{ij}^{\text{norm}} = \frac{X_{ij}}{\sqrt{\sum_{i=1}^{m} X_{ij}^2}}$$

Розрахунок для кожного значення:
Наприклад, для **Ціни**:

$$\sqrt{5000^2 + 4500^2 + 5200^2 + 4800^2} = 9835.7$$

$$X_{\text{A,}}^{\text{norm}} = \frac{5000}{9835.7} = 0.5085$$

Нормалізована матриця:

Альтернатива	Норм. Ціна	Норм. Якість	Норм. Доставка	Норм. Рейтинг
A	0.5085	0.5525	0.4851	0.5232
B	0.4576	0.4830	0.7277	0.4884
C	0.5290	0.6219	0.2425	0.5580
D	0.4882	0.4147	0.4851	0.5005

3. Формування зваженої нормалізованої матриці
Кожне значення множиться на вагу критерію:

$$V_{ij} = w_j \cdot X_{ij}^{\text{norm}}$$

Зважена матриця:

Альтернатива	Взв. Ціна	Взв. Якість	Взв. Доставка	Взв. Рейтинг
A	0.2034	0.1658	0.0970	0.0523
B	0.1830	0.1449	0.1455	0.0488
C	0.2116	0.1866	0.0485	0.0558
D	0.1953	0.1244	0.0970	0.0500

4. Визначення ідеальних рішень
Ідеальне рішення (найкраще за всіма критеріями):

$A^+ = (\max V_{ij}$ (для виграшних критеріїв), $\min V_{ij}$ (для витратних критеріїв)$)$

Антиідеальне рішення (найгірше за всіма критеріями):

$A^- = (\min V_{ij}$ (для виграшних критеріїв), $\max V_{ij}$ (для витратних критеріїв)$)$

Значення для нашого прикладу:

Критерій	Ідеальне рішення A^+	Антиідеальне рішення A^-
Ціна	0.1830 (мінімум)	0.2116 (максимум)
Якість	0.1866 (максимум)	0.1244 (мінімум)
Доставка	0.0485 (мінімум)	0.1455 (максимум)
Рейтинг	0.0558 (максимум)	0.0488 (мінімум)

5. Обчислення відстаней до ідеального та антиідеального рішень

$$D_i^+ = \sqrt{\sum_{j=1}^{n}(V_{ij} - A_j^+)^2}$$

$$D_i^- = \sqrt{\sum_{j=1}^{n}(V_{ij} - A_j^-)^2}$$

Розраховані значення:

Альтернатива	D^+	D^-
A	0.0435	0.0784
B	0.0642	0.0673
C	0.0321	0.0881
D	0.0597	0.0629

6. Обчислення відносної близькості до ідеального рішення

$$C_i = \frac{D_i^-}{D_i^+ + D_i^-}$$

Розраховані значення:

Альтернатива	C_i
A	0.643
B	0.511
C	0.733
D	0.513

Оптимальна альтернатива: C (0.733)

4.5. Висновок

Методи AHP та TOPSIS забезпечують більш комплексний підхід, оскільки враховують взаємозв'язок між критеріями та їхню відносну значущість. Однак метод SAW є швидшим та простішим у реалізації. Для підвищення точності аналізу варто розглянути можливість використання **гібридних методів** або проведення додаткового аналізу чутливості вибору альтернативи при зміні вагових коефіцієнтів.

Домашнє завдання

1. Обрати іншу прикладну задачу прийняття рішення, відмінну від наведеної у прикладі лабораторної роботи.
2. Сформувати:
   - 3–5 альтернатив;
   - 3–4 критерії оцінювання з різною спрямованістю (виграшні та витратні).
3. Виконати багатокритеріальний аналіз мінімум двома методами (SAW + AHP або SAW + TOPSIS).
4. Побудувати таблиці розрахунків у Excel або Google Sheets.
5. Побудувати графічне порівняння результатів (діаграма або стовпчикова візуалізація).

6. Підготувати короткий письмовий висновок (5–7 речень) щодо отриманого рішення.

Рекомендації щодо оформлення звіту

Звіт з лабораторної роботи оформлюється у друкованому або електронному вигляді та повинен містити такі структурні елементи:

1. Титульна сторінка, яка містить:
    - назву закладу освіти;
    - назву дисципліни;
    - номер та тему лабораторної роботи;
    - ПІБ студента;
    - групу;
    - рік виконання.
2. Мета роботи.
3. Короткі теоретичні положення (стисло, без надмірного копіювання).
4. Постановка задачі:
    - опис предметної області;
    - перелік альтернатив;
    - перелік критеріїв.
5. Хід виконання роботи:
    - вихідна таблиця даних;
    - розрахунки за методом SAW;
    - розрахунки за методом AHP;
    - розрахунки за методом TOPSIS;
    - проміжні таблиці та формули.
6. Візуалізація результатів:
    - графіки або діаграми порівняння альтернатив.
7. Аналіз результатів та висновки:
    - порівняння методів;
    - обґрунтування вибору оптимальної альтернативи.
8. Список використаних джерел.

Оформлення тексту:

- шрифт Times New Roman, 14 pt;
- міжрядковий інтервал — 1.5;
- поля стандартні;
- таблиці та рисунки мають бути пронумеровані та підписані.

Контрольні питання

1. Що таке СППР і в яких сферах вони застосовуються?
2. У чому полягає суть методу зваженої суми критеріїв (SAW)?
3. Які альтернативні методи прийняття рішень існують?
4. Як нормалізація впливає на коректність оцінки альтернатив?
5. Як зміняться результати, якщо змінити вагові коефіцієнти критеріїв?

Література

1. Сааті Т. Л. Прийняття рішень. Метод аналізу ієрархій : пер. з англ. – Москва : Радіо і зв’язок, 1993. – 278 с.
2. Hwang C.-L., Yoon K. Multiple Attribute Decision Making: Methods and Applications. – Berlin : Springer-Verlag, 1981. – 259 p.
3. Triantaphyllou E. Multi-Criteria Decision Making Methods: A Comparative Study. – Dordrecht : Springer, 2000. – 290 p.
4. Keen P. G. W., Scott Morton M. S. Decision Support Systems: An Organizational Perspective. – Reading, MA : Addison-Wesley, 1978. – 264 p.
5. Turban E., Sharda R., Delen D. Decision Support and Business Intelligence Systems. – 9th ed. – Upper Saddle River, NJ : Pearson Education, 2011. – 720 p.
6. ISO/IEC 25010:2011. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. – Geneva : ISO, 2011.

Варіанти

Варіант 1. Вибір постачальника товарів
Альтернативи:
- Постачальник A: низька ціна, середня якість, швидка доставка
- Постачальник B: середня ціна, висока якість, середня доставка
- Постачальник C: висока ціна, висока якість, довга доставка
- Постачальник D: низька ціна, низька якість, швидка доставка

Критерії оцінки:
- Ціна (менше – краще)
- Якість (більше – краще)
- Час доставки (менше – краще)

Варіант 2. Вибір автомобіля для корпоративного використання
Альтернативи:
- Авто A: Економічний, середній комфорт, середня надійність
- Авто B: Висока ціна, високий комфорт, висока надійність
- Авто C: Середня ціна, низький комфорт, висока надійність
- Авто D: Низька ціна, середній комфорт, низька надійність

Критерії оцінки:
- Вартість обслуговування (менше – краще)
- Комфорт (більше – краще)
- Надійність (більше – краще)

Варіант 3. Вибір ноутбука для роботи
Альтернативи:
- Ноутбук A: Дорогий, потужний, середній час автономної роботи
- Ноутбук B: Середня ціна, середня продуктивність, довгий час роботи
- Ноутбук C: Дешевий, слабка продуктивність, середній час автономії
- Ноутбук D: Висока ціна, висока продуктивність, короткий час автономії

Критерії оцінки:
- Вартість (менше – краще)
- Продуктивність (більше – краще)
- Автономність (більше – краще)

Варіант 4. Вибір місця для проведення конференції
Альтернативи:
- Зала A: Велика місткість, середня ціна, хороший транспорт
- Зала B: Середня місткість, висока ціна, високий рівень обслуговування
- Зала C: Невелика місткість, низька ціна, середній рівень комфорту
- Зала D: Висока місткість, висока ціна, зручне розташування

Критерії оцінки:
- Місткість (більше – краще)
- Вартість оренди (менше – краще)
- Рівень обслуговування (більше – краще)

Варіант 5. Вибір мобільного тарифного плану
Альтернативи:
- Тариф A: Низька ціна, мало інтернету, безлімітні дзвінки
- Тариф B: Висока ціна, багато інтернету, безлімітні дзвінки
- Тариф C: Середня ціна, обмежений інтернет, дешеві дзвінки
- Тариф D: Дуже низька ціна, мало інтернету, платні дзвінки

Критерії оцінки:
- Вартість (менше – краще)
- Кількість мобільного інтернету (більше – краще)
- Вартість дзвінків (менше – краще)

Варіант 6. Вибір системи управління проектами
Альтернативи:
- Система A: Безкоштовна, базові функції, проста у використанні
- Система B: Платна, багато функцій, складна у використанні

- Система C: Середня ціна, середня кількість функцій, середня складність
- Система D: Безкоштовна, мінімальні можливості, дуже проста

Критерії оцінки:

- Вартість (менше – краще)
- Функціональність (більше – краще)
- Зручність у використанні (більше – краще)

Варіант 7. Вибір стратегії маркетингу для стартапу

Альтернативи:

- Стратегія A: Соцмережі, низькі витрати, довгий термін окупності
- Стратегія B: Контекстна реклама, високі витрати, швидка віддача
- Стратегія C: Таргетована реклама, середні витрати, середня ефективність
- Стратегія D: Реклама на YouTube, високі витрати, довгий термін ефективності

Критерії оцінки:

- Вартість (менше – краще)
- Охоплення аудиторії (більше – краще)
- Термін окупності (менше – краще)

Лабораторна робота №2. Експертні системи: аналіз даних на основі логічних правил

Мета роботи: Ознайомитися з основними принципами роботи експертних систем, що засновані на логічних правилах (логічні вирази, умови, комбінації правил).

Короткі теоретичні положення

Експертна система – це програмний комплекс, що використовує накопичені знання та логічні правила для вирішення завдань у певній предметній галузі. В основі експертних систем лежить база знань (правил) та механізм логічного виводу, який обирає потрібні правила та формує результат.

Ключові елементи експертної системи:

- **База знань** (набір правил і фактів).
- **Механізм виводу** (алгоритми, що аналізують факти та застосовують правила).
- **Пояснювальний модуль** (не завжди обов'язково, але часто використовується для пояснення логіки прийняття рішень).

У контексті Excel/Google Sheets базою знань можуть бути правила, сформульовані у вигляді формул (наприклад, IF, IFS, комбінації IF з AND, OR, NOT тощо), а механізм виводу – це обчислювальна модель, що застосовує ці формули до вхідних даних та повертає результат.

2.2 Основні логічні функції в Excel/Google Sheets

Функція	Опис функції та синтаксис	Приклад використання та результат
IF	Повертає одне значення, якщо умова істинна, та інше, якщо умова хибна. IF(умова; якщо_істина; якщо_хиба)	=IF(A1>=50;"Здано";"Не здано")якщо A1=55 → "Здано"
IFS	Перевіряє кілька умов і повертає значення для першої істинної умови.IFS(умова1; значення1; [умова2; значення2];...)	=IFS(A1>=90;"відмінно";A1>=75;"добре";TRUE;"незадовільно") якщо A1=85 → "добре"
AND	Повертає TRUE, якщо всі умови істинні.AND(умова1; [умова2];...)	=AND(A1>=50;B1>=50) → TRUE тільки якщо обидві умови виконано
OR	Повертає TRUE, якщо принаймні одна умова істинна.OR(умова1; [умова2];...)	=OR(A1>=90;B1>=90) → TRUE, якщо хоча б одна умова істинна
NOT	Інвертує логічне значення умови.NOT(умова)	=NOT(A1>=50) якщо A1=55 → FALSE (оскільки умова істинна, то NOT поверне FALSE)
XOR	Повертає TRUE, якщо істинна тільки одна умова з усіх.XOR(умова1; [умова2];...)	=XOR(A1>=50;B1>=50) поверне TRUE тільки якщо одна умова виконується (інша – ні). Наприклад, якщо A1=55, B1=40 → TRUE
COUNTIF	Підраховує кількість комірок, що відповідають заданій	=COUNTIF(A1:A10;">=90") –

Функція	Опис функції та синтаксис	Приклад використання та результат
	умові.COUNTIF(діапазон; критерій)	підрахує кількість комірок у діапазоні з балом ≥90
SUMPRODUCT	Повертає суму добутків елементів масивів, використовується для підрахунку кількості виконаних умов.SUMPRODUCT(масив1; [масив2];...)	=SUMPRODUCT(--(A1:A5>=90)) – підраховує кількість комірок у діапазоні, значення яких ≥90

2.3 Приклади експертних правил у формі логічних виразів

Нижче наведено кілька прикладів формул, які можуть використовуватися як експертні правила в електронних таблицях Excel або Google Sheets.

1. **Класифікація за діапазоном значень**

```
=IF(A2 > 100, "Понад норму", "У межах норми")
```

Якщо значення в комірці A2 перевищує 100, виводиться повідомлення «Понад норму», інакше – «У межах норми».

2. **Логічне правило з двома умовами (AND)**

```
=IF(AND(A2 >= 50, B2 <= 100), "Пройшов", "Не пройшов")
```

Якщо A2 ≥ 50 та B2 ≤ 100, результат – «Пройшов», інакше – «Не пройшов».

3. **Логічне правило з альтернативними умовами (OR)**

```
=IF(OR(A2 = "Критичний", B2 > 10), "Увага!", "Звичайний режим")
```

Якщо у стовпці A2 міститься значення «Критичний» *або* B2 більше 10, виводиться «Увага!», інакше – «Звичайний режим».

4. **Комбінований приклад з умовами AND та OR**

```
=IF(AND(A2 > 0, OR(B2 = "Терміновий", C2 > 1000)), "Виконати негайно", "Може зачекати")
```

Логіка така: щоб отримати «Виконати негайно», потрібно, аби A2 було більше 0 (виконується перша умова) та (друга умова) B2 дорівнювало «Терміновий» *або* C2 перевищувало 1000.

5. **Використання NOT**

```
=IF(NOT(A2 = "Активний"), "Неактивний або відсутній", "Активний")
```

Якщо у стовпці A2 *не* «Активний», то виводиться «Неактивний або відсутній», інакше «Активний». (NOT інвертує логічне значення умови.)

6. **Визначення текстового висновку з кількома категоріями (IFS)**

```
=IFS(
   A2 < 0, "Помилка вхідних даних",
   A2 < 50, "Низький рівень",
   A2 < 80, "Середній рівень",
   A2 <= 100, "Високий рівень",
   A2 > 100, "Поза визначеними межами"
)
```

Тут послідовно перевіряються кілька умов: якщо A2 менше 0 – «Помилка...», якщо менше 50 – «Низький рівень» і т.д.

7. **Складене правило для перевірки допустимих значень**

```
=IF(AND(A2>=1, A2<=10), "Допустимий діапазон", "Недопустиме значення")
```

Якщо A2 знаходиться між 1 та 10 (включно), виводиться «Допустимий діапазон», інакше – «Недопустиме значення».

8. **Обчислення рівня терміновості з урахуванням двох змінних**

```
=IF(OR(A2 > 5, B2 = "Критичний"), "Висока терміновість", "Звичайна терміновість")
```

Якщо A2 перевищує 5 або B2 має значення «Критичний», виводиться «Висока терміновість», інакше – «Звичайна терміновість».

9. **Рекомендація на основі доступності ресурсів**

```
=IF(AND(C2 > 100, D2 < 50), "Виділити додаткові ресурси", "Надати базові ресурси")
```

Якщо у стовпці C2 ресурси перевищують 100, але в стовпці D2 показник (наприклад, загальна завантаженість) менший за 50, то «Виділити додаткові ресурси», інакше – «Надати базові ресурси».

10. **Комплексне правило з IFS і логічними операціями**

```
=IFS(
   A2 < 0, "Помилка вхідних даних",
```

```
  A2 < 50, "Низька продуктивність",
  AND(A2 >= 50, A2 < 80), "Середня продуктивність",
  AND(A2 >= 80, A2 <= 100), "Висока продуктивність",
  A2 > 100, "Вийшло за межі"
)
```

Це приклад, де для кожного заданого діапазону чи умови передбачене окреме текстове повідомлення.

Завдання до роботи

1. Підготовка даних
    - Створити (або відкрити) електронну таблицю в Excel/Google Sheets.
    - У першому рядку (заголовок) визначити назви стовпців, які будуть відображати вхідні параметри для експертної системи (наприклад, «Рівень продажу», «Витрати», «Прибуток», «Рейтинг» тощо).
    - Заповнити кілька рядків прикладними даними (мінімум 10 рядків).
2. Формулювання логічних правил
    - Визначити принаймні 3–5 різних правил. Наприклад, якщо «Витрати» > певного значення та «Прибуток» < певного значення, тоді «Ризик банкрутства» = високий; або якщо «Рівень продажу» > 1000 одиниць, призначати знижку клієнтам і т.д.
    - Записати ці правила у вигляді логічних виразів (IF, AND, OR, IFS).
3. Створення автоматизованих висновків
    - У новому стовпці (назвемо його «Висновок» або «Рекомендація») впровадити логічні правила.
    - Заповнити формули, щоб автоматично з’являвся потрібний висновок чи рекомендація.
    - Перевірити правильність роботи формул (чи дають вони логічний результат при зміні вхідних даних).
4. Додатково
    - Умовне форматування: Зробити форматування рядків (або стовпців) залежно від значення «Висновку». Наприклад, якщо «Висновок» = «Негативний», то виділити весь рядок червоним фоном, якщо «Позитивний» – зеленим тощо.
    - Перевірка даних (Data Validation): Заборонити введення некоректних значень або виводити попередження, якщо користувач вводить значення, які виходять за межі визначеного діапазону.

Опис способу обробки результатів, хід експерименту

У межах лабораторної роботи експертна система реалізується у вигляді набору логічних правил, закодованих за допомогою функцій Excel або Google Sheets. Обробка результатів полягає у послідовному застосуванні цих правил до вхідних даних з метою автоматизованого формування висновків.

Хід експерименту включає такі етапи:

1. Формування вхідних даних

Створюється таблиця з числовими або текстовими параметрами, що описують об’єкти аналізу (наприклад, результати тестування, показники діяльності, рівні ризику тощо). Дані вводяться вручну або генеруються тестовим способом.

2. Формалізація знань у вигляді логічних правил

Експертні знання предметної області трансформуються у систему умов типу IF–THEN, які реалізуються за допомогою функцій IF, IFS, AND, OR, NOT, XOR, COUNTIF, SUMPRODUCT.

3. Реалізація механізму логічного виводу

Логічні правила застосовуються до кожного рядка таблиці, формуючи автоматизований висновок або рекомендацію. Обчислення виконуються динамічно — при зміні вхідних даних результат оновлюється автоматично.

4. Перевірка коректності роботи правил

Проводиться тестування системи шляхом:

- навмисної зміни окремих значень;
- перевірки граничних випадків (мінімальні та максимальні значення);
- аналізу суперечливих або некоректних даних.

5. Аналіз отриманих результатів

Результати роботи експертної системи аналізуються з точки зору:

- логічної узгодженості;
- повноти правил;
- відсутності неоднозначних або взаємовиключних висновків.

Приклад виконання

Умова задачі:

Є таблиця, що містить інформацію про студентів і результати складання ними чотирьох тестів з дисципліни. Потрібно створити систему автоматизованого прийняття рішення щодо допуску студентів до іспиту на основі заданих критеріїв (логічних правил).

Критерії для допуску:

- Студент допускається, якщо він здав **усі чотири** тести (бал за кожний тест не менше 50 балів).
- Якщо студент має хоча б один незданий тест (менше 50 балів), він не допускається.
- Якщо студент має хоча б два тести з високими результатами (≥ 90), він допускається з відзнакою («Допущений з відзнакою»).
- Якщо студент має рівно **один** незданий тест, він отримує можливість додаткової перездачі («Допущений до перездачі»).
- Якщо студент має чотири однакові результати (усі 100), він отримує спеціальну відзнаку («Відмінник»).

Крок 1: Вхідні дані

Студент	Тест 1	Тест 2	Тест 3	Тест 4
Іваненко А.	90	100	95	92
Петренко Б.	45	88	90	93
Сидоренко В.	100	100	100	100
Мельниченко Г.	50	52	48	60
Савченко Д.	60	49	51	54

Крок 2: Реалізація логічних правил у формулах Excel/Google Sheets:

Додамо в таблицю стовпець «Статус», у якому визначатиметься рішення щодо студента на основі заданих критеріїв.

Формула для комірки F2 («Статус» для студента Іваненко А.):

```
=IFS(
   AND(COUNTIF(B2:E2, "=100")=4), "Відмінник",
   SUMPRODUCT(--(B2:E2>=90))>=2, "Допущений з відзнакою",
   COUNTIF(B2:E2,"<50")=1, "Допущений до перездачі",
   COUNTIF(B2:E2,"<50")>1, "Не допущений",
   AND(B2>=50, C2>=50, D2>=50, E2>=50), "Допущений",
   TRUE, "Не допущений"
)
```

Пояснення формули:

- COUNTIF(B2:E2, "=100")=4 – перевіряє, чи всі чотири тести мають 100 балів.
- SUMPRODUCT(--(B2:E2>=90))>=2 – підраховує кількість тестів з балами ≥ 90, якщо таких два чи більше, студент отримує «відзнаку».
- COUNTIF(B2:E2,"<50")=1 – якщо рівно один тест незданий, студент отримує «перездачу».
- COUNTIF(B2:E2,"<50")>1 – якщо два чи більше тестів нездані, студент не допускається.
- AND(B2>=50, C2>=50, D2>=50, E2>=50) – перевіряє, чи студент здав усі тести.
- TRUE – якщо не спрацювала жодна умова, студент не допускається.

Скопіювати формулу вниз для інших студентів (F3:F6).

Крок 3: Додаткове застосування функцій XOR та NOT:

Приклад використання XOR і NOT для перевірки особливих умов:

Припустимо, що студент допускається до конкурсу, якщо він **рівно один раз** набрав менше 60 балів.

```
=IF(XOR(B2<60, C2<60, D2<60, E2<60), "Учасник конкурсу", "Не учасник конкурсу")
```

Така формула буде повертати "Учасник конкурсу" тільки якщо рівно одна умова є істиною.

Крок 4: Підсумкова таблиця результатів:

Студент	Тест 1	Тест 2	Тест 3	Тест 4	Статус	Конкурс
Іваненко А.	90	100	95	92	Допущений з відзнакою	Не учасник конкурсу
Петренко Б.	45	88	90	93	Допущений до перездачі	Учасник конкурсу
Сидоренко В.	100	100	100	100	Відмінник	Не учасник конкурсу
Мельниченко Г.	50	52	48	60	Допущений до перездачі	Учасник конкурсу
Савченко Д.	60	49	51	54	Допущений до перездачі	Не учасник конкурсу

Домашнє завдання

1. Обрати іншу предметну область для побудови експертної системи (наприклад: оцінювання кредитоспроможності клієнтів, визначення рівня ризику проєкту, аналіз стану обладнання, оцінювання ефективності співробітників).
2. Сформувати таблицю з не менше ніж 4 вхідними параметрами та 10–15 записами.
3. Розробити щонайменше 5 логічних правил, які:
   - використовують різні логічні функції;
   - містять складені умови (AND, OR, NOT, XOR).
4. Реалізувати автоматизований висновок у окремому стовпці таблиці.
5. Додати:
   - умовне форматування результатів;
   - перевірку введення даних (Data Validation).
6. Підготувати короткий висновок (5–7 речень) щодо ефективності використаних правил.

Рекомендації щодо оформлення звіту

Звіт з лабораторної роботи повинен містити такі структурні елементи:

1. Титульна сторінка

(назва закладу освіти, дисципліна, номер і тема лабораторної роботи, ПІБ, група, рік).

2. Мета роботи.
3. Короткі теоретичні положення

(основні поняття експертних систем, логічні правила, роль механізму виводу).

4. Постановка задачі

(опис предметної області, критерії, очікувані результати).

5. Хід виконання роботи:
   - структура таблиці;
   - приклади логічних правил;
   - використані формули.
6. Результати експерименту:
   - підсумкова таблиця;
   - приклади автоматизованих висновків.
7. Аналіз результатів та висновки.
8. Список використаних джерел.

Вимоги до оформлення:

- шрифт Times New Roman, 14 pt;
- міжрядковий інтервал — 1.5;
- формули та таблиці повинні бути читабельними та підписаними.

Контрольні питання

1. Що таке експертна система та які її основні складові?
2. Як реалізувати базу знань та механізм виводу в Excel/Google Sheets?
3. Яка різниця між логічними функціями IF, IFS, AND, OR?
4. Як налаштувати умовне форматування для візуального відображення результатів логічних перевірок?
5. У чому переваги використання формул порівняно з ручним аналізом даних?
6. Як працює функція NOT і коли вона може бути корисною?
7. Як перевірити коректність логічних формул у таблиці?

Література

1. Сааті Т. Л. Прийняття рішень. Метод аналізу ієрархій : пер. з англ. – Москва : Радіо і зв’язок, 1993. – 278 с.

2. Turban E., Sharda R., Delen D. Decision Support and Business Intelligence Systems. – 9th ed. – Upper Saddle River, NJ : Pearson Education, 2011. – 720 p.
3. Jackson P. Introduction to Expert Systems. – 3rd ed. – Harlow : Addison-Wesley, 1999. – 542 p.
4. Giarratano J., Riley G. Expert Systems: Principles and Programming. – 4th ed. – Boston : PWS Publishing, 2005. – 656 p.
5. ISO/IEC 25010:2011. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. – Geneva : ISO, 2011.

Варіанти

Варіант 1. Оцінка ризиків для проектів

Вхідні дані:

Проект	Поточний бюджет	Витрати	Термін виконання (факт, днів)	Термін плановий (днів)
Проект А	100000	110000	35	30
Проект Б	250000	240000	60	60
Проект В	50000	30000	20	25
Проект Г	75000	80000	45	40
Проект Д	150000	150000	28	30

Умови для логічних правил:

1. Якщо **Поточний бюджет** < **Витрат** або **фактичний термін виконання** > **плановий термін**, то позначити проект як «**Високий ризик**».
2. Якщо **Поточний бюджет** ≥ **Витрат**, але **фактичний термін** наближається до критичного (наприклад, перевищено на 10% планового терміну або є інші показники ризику), позначити «**Середній ризик**».
3. В іншому разі – «**Низький ризик**».

Варіант 2. Рейтинг кредитоспроможності клієнтів

Вхідні дані:

Клієнт	Доходи (тис. грн)	Коефіцієнт заборгованості*	Кількість прострочених платежів
Іванов	70	0.2	0
Петров	40	0.35	2
Сидоренко	20	0.6	1
Коваленко	80	0.1	0
Гриценко	15	0.4	3

Умови для логічних правил (спрощений варіант):

1. Якщо «Доходи» > 50 та «Коефіцієнт заборгованості» < 0.3, тоді «**Висока кредитоспроможність**».
2. Якщо «Доходи» від 20 до 50 або «Коефіцієнт заборгованості» між 0.3 та 0.5 і прострочених платежів ≤ 2, тоді «**Середня**».
3. Інакше – «**Низька**».

Варіант 3. Сортування товарів за рівнем пріоритету постачання

Вхідні дані:

Товар	Попит (шт./тиждень)	Залишок (шт.)	Категорія товару
Товар А	500	10	Продовольчі
Товар Б	150	80	Будматеріали
Товар В	1000	5	Техніка
Товар Г	300	100	Продовольчі
Товар Д	700	30	Побутова хімія

Умови для логічних правил (приклад):

1. Якщо «Попит» > 800 (дуже високий) та «Залишок» < 10 (критично низький), позначити «**Негайний**» (пріоритет доставки).

2. Якщо «Попит» від 300 до 800 та «Залишок» середній (наприклад, 10–50), позначити «**Стандартний**».
3. Якщо «Попит» низький (<300) або «Залишок» достатній (>50), позначити «**Низький**».

Варіант 4. Аналіз продуктивності працівників

Вхідні дані:

Працівник	Кількість виконаних завдань	Середній час виконання (хв)	Якість результату (оцінка 1-10)
Іваненко	35	15	9
Петренко	20	25	8
Мельник	40	20	6
Шевченко	25	18	9
Коваль	30	30	7

Приклад умов:

1. Якщо **кількість завдань** > 30 і **якість результату** ≥ 8, то «**Висока ефективність**».
2. Якщо **кількість завдань** від 20 до 30 і **якість** ≥ 8, то «**Достатня ефективність**» (із потенціалом зростання).
3. Якщо **кількість завдань** < 20 або **якість** < 5, то «**Низька ефективність**».
4. Якщо залишаються інші комбінації – «**Середня ефективність**».

Варіант 5. Фільтрація заявок до технічної підтримки

Вхідні дані:

Номер заявки	Тип проблеми	Терміновість	Критичні часові обмеження (так/ні)
1001	Збій сервера	висока	так
1002	Налаштування ПО	середня	ні
1003	Апаратна проблема	низька	ні
1004	Критичний збій систем	висока	так
1005	Доопрацювання звітів	середня	так

Умови для логічних правил:

1. Якщо «Терміновість» = висока та «Критичні часові обмеження» = так, позначити як «**Термінова обробка**».
2. Інакше – «**Звичайна обробка**» (або розподіл на «Середню» чи «Низьку» пріоритетність залежно від типу проблеми і терміновості).

Варіант 6. Визначення пріоритету маршруту доставки (логістика)

Вхідні дані:

Маршрут	Відстань (км)	Тип вантажу	Терміновість (висока/середня/низька)	Можливі затримки (так/ні)
A	800	Небезпечний вантаж	висока	так
B	150	Швидкопсувний	середня	ні
C	2000	Стандартний	висока	так
D	500	Цінний	середня	ні
E	1200	Швидкопсувний	низька	так

Приклад умов для визначення пріоритету доставки:

1. Якщо терміновість висока та є ризик затримок, пріоритет **«Терміновий»**.
2. Якщо вантаж небезпечний або цінний та відстань перевищує певний поріг, можливо, пріоритет підвищується.
3. Якщо терміновість низька і затримки не очікуються, пріоритет знижується до **«Стандартного»** чи **«Низького»**.
4. Та інші логічні комбінації залежно від конкретних вимог.

Варіант 7. Визначення відповідності кандидатів на вакансію (рекрутинг)

Вхідні дані:

Кандидат	Досвід (роки)	Спеціалізація	Рівень володіння іноземною мовою (бали 1-5)	Очікувана заробітна плата (тис. грн)
Коваль	5	Розробник ПЗ	4	40
Іваненко	2	Тестувальник	3	25
Петренко	7	Менеджер проектів	5	60
Сидоренко	3	Розробник ПЗ	2	35
Мельник	10	Системний аналітик	4	55

Приклад умов відбору:

1. Необхідний мінімальний досвід (наприклад, ≥3 роки) і певна спеціалізація.
2. Високий рівень володіння іноземною мовою (≥4) може підвищувати пріоритет кандидата.
3. Якщо очікувана зарплата перевищує бюджет, кандидат не розглядається або переходить у «резерв».
4. Можуть бути додаткові фактори, такі як наявність сертифікацій, додаткові навички тощо.

Лабораторна робота №3. Застосування методів кластеризації та класифікації для аналізу даних

Мета роботи: Набути практичних навичок застосування основних методів кластеризації (на прикладі k-means) та класифікації (на прикладі дерева рішень або методу опорних векторів) з області інтелектуального аналізу даних (Data Mining).

Короткі теоретичні положення

- **Інтелектуальний аналіз даних (Data Mining):** Процес виявлення прихованих, нетривіальних, практично корисних та доступних для інтерпретації знань (шаблонів, закономірностей) у великих обсягах даних.
- **Навчання без учителя (Unsupervised Learning):** Тип машинного навчання, де модель навчається на даних без міток класів.
    - **Кластеризація:** Завдання розбиття множини об'єктів на групи (кластери) таким чином, щоб об'єкти в межах одного кластера були більш подібними між собою, ніж з об'єктами з інших кластерів.
        - *Метод k-means:* Ітеративний алгоритм, що мінімізує суму квадратів відстаней між об'єктами та центрами їх кластерів (центроїдами). Вимагає попереднього задання кількості кластерів 'k'.
        - *Метрики подібності/відстані:* Евклідова відстань, манхеттенська відстань тощо.
        - *Оцінка якості кластеризації:* Метод "ліктя" (Elbow method) для вибору 'k', силуетний коефіцієнт (Silhouette Score).
- **Навчання з учителем (Supervised Learning):** Тип машинного навчання, де модель навчається на даних, що мають мітки класів (цільову змінну).
    - **Класифікація:** Завдання віднесення об'єкта до одного з попередньо визначених класів на основі його характеристик (ознак).
        - *Алгоритми:* Дерева рішень (Decision Trees), Метод опорних векторів (Support Vector Machines - SVM), Логістична регресія, k-Найближчих сусідів (k-NN) тощо.
        - *Навчальна та тестова вибірки:* Дані розділяються для навчання моделі та незалежної перевірки її ефективності.
        - *Оцінка якості класифікації:* Матриця помилок (Confusion Matrix), Точність (Accuracy), Повнота (Recall), Точність передбачення (Precision), F1-міра (F1-Score).
- **Попередня обробка даних (Data Preprocessing):** Етап підготовки даних перед моделюванням, що може включати: очищення даних (обробка пропущених значень, викидів), трансформацію даних (нормалізація, стандартизація), вибір ознак.

Завдання до роботи

1. **Вибір та завантаження даних:** Обрати набір даних (dataset) для аналізу. Рекомендовані варіанти:
    - Класичні набори даних: Iris, Wine (доступні в бібліотеці Scikit-learn).
2. **Попередній аналіз та підготовка даних:**
    - Ознайомитися зі структурою даних (кількість записів, ознак, типи даних).
    - Перевірити наявність пропущених значень та обрати стратегію їх обробки (видалення, заповнення середнім/медіаною/модою).

- Провести візуальний аналіз розподілу ознак (гістограми, діаграми розсіювання).
- Якщо необхідно, виконати стандартизацію або нормалізацію числових ознак (наприклад, за допомогою StandardScaler). Визначити, які ознаки будуть використані для кластеризації та класифікації.

3. **Кластеризація:**
   - Реалізувати алгоритм кластеризації k-means.
   - Обґрунтувати вибір кількості кластерів 'k', використовуючи метод "ліктя" (побудувати графік залежності суми квадратів відстаней до центроїдів WCSS від 'k').
   - Провести кластеризацію з обраним 'k'.
   - Візуалізувати результати кластеризації (наприклад, діаграму розсіювання для двох основних ознак, забарвлену за кластерами).
   - Оцінити якість кластеризації (наприклад, обчислити силуетний коефіцієнт).
4. **Класифікація:**
   - Розділити дані на навчальну (training) та тестову (testing) вибірки (зазвичай у співвідношенні 70/30 або 80/20).
   - Обрати та реалізувати щонайменше один алгоритм класифікації (рекомендовано: Дерево рішень).
   - Навчити модель класифікації на навчальній вибірці.
   - Зробити прогнози для тестової вибірки.
   - Оцінити якість класифікації: побудувати матрицю помилок, розрахувати Accuracy, Precision, Recall, F1-Score (наприклад, за допомогою classification_report з Scikit-learn).
   - Візуалізувати модель (наприклад, побудувати графічне представлення дерева рішень).
5. **Інтерпретація результатів:**
   - Описати характеристики об'єктів, що потрапили до кожного кластера.
   - Проаналізувати результати роботи класифікатора (наскільки добре модель розрізняє класи, які помилки робить).
   - Сформулювати висновки про те, як отримані результати (кластери, модель класифікації) можуть бути використані для підтримки прийняття рішень у предметній області обраного набору даних (наприклад, сегментація клієнтів для маркетингу, попередня діагностика, виявлення аномалій).

Опис способу обробки результатів, хід експерименту

Результати лабораторної роботи отримуються шляхом послідовного виконання етапів підготовки даних, побудови моделей кластеризації та класифікації, а також оцінювання їх якості за стандартними метриками.

1. Підготовка даних: завантаження набору даних, перевірка структури, обробка пропусків (за наявності), стандартизація/нормалізація числових ознак.
2. Кластеризація (k-means): виконання кластеризації з різними значеннями k, вибір оптимального k (метод “ліктя”), отримання міток кластерів, оцінювання якості (силуетний

коефіцієнт) та візуалізація результатів.

3. Класифікація (дерево рішень або SVM): поділ даних на навчальну і тестову вибірки, навчання моделі, прогнозування та оцінювання якості (матриця помилок, Accuracy, Precision, Recall, F1-score), візуалізація моделі (за потреби).
4. Інтерпретація результатів: аналіз характеристик кластерів і типових помилок класифікації, формулювання висновків щодо можливого використання отриманих моделей для підтримки прийняття рішень.

Цифрові значення метрик та графічні матеріали (діаграми, матриця помилок, візуалізація кластерів/моделі) використовуються як основа для підсумкового аналізу та висновків у звіті.

Приклад виконання

1. **Створення середовища та імпорт бібліотек:**
    - Створити новий Python скрипт (.py) або Jupyter Notebook (.ipynb).
    - Імпортувати необхідні бібліотеки.

```
# Імпорт бібліотек для роботи з даними
import pandas as pd
import numpy as np

# Імпорт для кластеризації
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler # Для масштабування даних
from sklearn.metrics import silhouette_score

# Імпорт для класифікації
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier, plot_tree # Дерево рішень
# Альтернативно: from sklearn.svm import SVC # Метод опорних векторів
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Імпорт для візуалізації
import matplotlib.pyplot as plt
import seaborn as sns

# Налаштування візуалізацій (опціонально)
sns.set(style="whitegrid")
plt.rcParams['figure.figsize'] = (10, 6)
```

2. **Завантаження даних:**
    - Завантажити обраний набір даних. У прикладі використаємо вбудований датасет `Iris` з `sklearn`. Для завантаження з CSV-файлу використовуйте `pd.read_csv('шлях/до/вашого/файлу.csv')`.

Python

```
# Завантаження датасету Iris з sklearn
from sklearn.datasets import load_iris
iris = load_iris()
# Створення Pandas DataFrame для зручності
# iris.data містить ознаки, iris.feature_names - їх назви
iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
# iris.target містить мітки класів (0, 1, 2), iris.target_names - їх назви
iris_df['target'] = iris.target
```

```
iris_df['species'] = iris_df['target'].map({0: iris.target_names[0], 1: iris.target_names[1], 2:
iris.target_names[2]})

print("Перші 5 рядків даних:")
print(iris_df.head())
print("\nІнформація про датасет:")
iris_df.info()
print("\nОписова статистика:")
print(iris_df.describe())
```

3. **Попередній аналіз та підготовка даних:**

    - Перевірити пропуски, типи даних (зроблено вище за допомогою .info(), .describe()).
    - Для багатьох алгоритмів (включаючи k-means та SVM) важливо масштабувати дані. Використаємо StandardScaler. Будемо масштабувати лише числові ознаки, які використовуються для моделювання.

```
# Вибір ознак для кластеризації/класифікації (всі числові ознаки)
features = iris.feature_names
X = iris_df[features]
# Цільова змінна для класифікації
y = iris_df['target']

# Перевірка на пропуски
print("\nКількість пропущених значень по стовпцях:")
print(X.isnull().sum())
# У датасеті Iris немає пропусків. Якщо є, їх треба обробити:
# Наприклад, заповнення середнім:
# from sklearn.impute import SimpleImputer
# imputer = SimpleImputer(strategy='mean')
# X = imputer.fit_transform(X)
# Або видалення рядків з пропусками: X.dropna(inplace=True)

# Масштабування ознак
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# X_scaled - це масив NumPy, для зручності можна повернути в DataFrame
X_scaled_df = pd.DataFrame(X_scaled, columns=features)
print("\nМасштабовані ознаки (перші 5 рядків):")
print(X_scaled_df.head())
```

4. **Кластеризація (k-means):**

    - Визначити оптимальну кількість кластерів 'k' за допомогою методу "ліктя".
    - Навчити модель k-means з обраним 'k'.
    - Оцінити та візуалізувати результати.

```
# --- Кластеризація ---
print("\n--- Кластеризація (k-means) ---")
# Визначення оптимального k за методом "ліктя"
wcss = [] # Within-cluster sum of squares
k_range = range(1, 11) # Перевіряємо k від 1 до 10
```

```
for k in k_range:
    kmeans_model = KMeans(n_clusters=k, init='k-means++', n_init=10, random_state=42)
    kmeans_model.fit(X_scaled)
    wcss.append(kmeans_model.inertia_) # inertia_ = WCSS

# Побудова графіка методу "ліктя"
plt.figure(figsize=(10, 6))
plt.plot(k_range, wcss, marker='o', linestyle='--')
plt.title('Метод "ліктя" для визначення оптимального k')
plt.xlabel('Кількість кластерів (k)')
plt.ylabel('WCSS (Сума квадратів відстаней)')
plt.xticks(k_range)
plt.grid(True)
plt.show()

# За графіком, "лікоть" знаходиться приблизно на k=3. Обираємо k=3
optimal_k = 3
print(f"\nОбрана оптимальна кількість кластерів: {optimal_k}")

# Застосування k-means з оптимальним k
kmeans = KMeans(n_clusters=optimal_k, init='k-means++', n_init=10, random_state=42)
cluster_labels = kmeans.fit_predict(X_scaled)

# Додавання міток кластерів до основного DataFrame
iris_df['cluster'] = cluster_labels

# Оцінка якості кластеризації (силуетний коефіцієнт)
silhouette_avg = silhouette_score(X_scaled, cluster_labels)
print(f"Силуетний коефіцієнт для k={optimal_k}: {silhouette_avg:.4f}")

# Візуалізація кластерів (на прикладі перших двох ознак)
plt.figure(figsize=(10, 6))
sns.scatterplot(x=X_scaled[:, 0], y=X_scaled[:, 1], hue=cluster_labels, palette='viridis', s=100, alpha=0.8)
# Додаємо центроїди
centers = kmeans.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], marker='X', s=200, c='red', label='Центроїди')
plt.title('Результати кластеризації K-Means (k=3)')
# Використовуємо оригінальні назви ознак для підписів осей
plt.xlabel(f'Масштабована {features[0]}')
plt.ylabel(f'Масштабована {features[1]}')
plt.legend()
plt.grid(True)
plt.show()

# Аналіз розміру кластерів
print("\nРозподіл об'єктів за кластерами:")
print(iris_df['cluster'].value_counts())
```

5. **Класифікація (Дерево рішень):**

    - Розділити дані на навчальну та тестову вибірки.
    - Навчити модель класифікації.

- Оцінити її ефективність.

```
# --- Класифікація (Дерево рішень) ---
# Використовуємо масштабовані дані X_scaled та оригінальні мітки класів y
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42, stratify=y)

print(f"Розмір навчальної вибірки: {X_train.shape[0]} об'єктів")
print(f"Розмір тестової вибірки: {X_test.shape[0]} об'єктів")

# Ініціалізація та навчання моделі дерева рішень
# max_depth обмежує глибину дерева для уникнення перенавчання
dt_classifier = DecisionTreeClassifier(criterion='gini', max_depth=4, random_state=42)
dt_classifier.fit(X_train, y_train)

# Прогнозування на тестовій вибірці
y_pred = dt_classifier.predict(X_test)

# Оцінка якості класифікації
accuracy = accuracy_score(y_test, y_pred)
print(f"\nТочність (Accuracy) моделі: {accuracy:.4f}")

print("\nМатриця помилок (Confusion Matrix):")
cm = confusion_matrix(y_test, y_pred)
# Візуалізація матриці помилок
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=iris.target_names,
yticklabels=iris.target_names)
plt.title('Матриця помилок')
plt.ylabel('Справжній клас')
plt.xlabel('Передбачений клас')
plt.show()


print("\nЗвіт про класифікацію (Classification Report):")
report = classification_report(y_test, y_pred, target_names=iris.target_names)
print(report)

# Візуалізація дерева рішень (може бути великою для глибоких дерев)
plt.figure(figsize=(20, 10)) # Збільшений розмір для кращої читабельності
plot_tree(dt_classifier,
          filled=True,
          rounded=True,
          class_names=iris.target_names,
          feature_names=features)
plt.title("Візуалізація Дерева Рішень")
plt.show()
```

6. **Інтерпретація та документування:**

   - Проаналізувати отримані графіки, метрики, структуру дерева.
   - Сформулювати висновки щодо характеристик кластерів та ефективності класифікатора.

- Задокументувати результати та висновки у звіті.

Пояснення до коду:

1. **Імпорт бібліотек:** Завантажуються всі необхідні інструменти для роботи з даними (pandas, numpy), візуалізації (matplotlib, seaborn) та машинного навчання (sklearn).
2. **Завантаження даних:** Використовується вбудований датасет Iris. Код показує, як створити DataFrame, додати назви стовпців і класів, вивести базову інформацію (head, info, describe) та розподіл за класами (value_counts). Також додано закоментовані приклади візуального аналізу (pairplot, boxplot).
3. **Підготовка даних:** Виділяються ознаки (X) та цільова змінна (y). Здійснюється перевірка на пропуски. Найважливіший крок тут - **масштабування** ознак за допомогою StandardScaler, що приводить їх до нульового середнього та одиничної дисперсії, що є важливим для k-means та SVM.
4. **Кластеризація:**
   - Спочатку реалізується **Метод "ліктя"**, щоб знайти оптимальну кількість кластерів (k). Будується графік залежності WCSS (суми квадратів відстаней від точок до центрів їх кластерів) від k. Оптимальне k зазвичай знаходиться в точці "зламу" графіка.
   - Потім застосовується KMeans з обраним k. Результат (cluster_labels) додається до DataFrame.
   - Обчислюється **Силуетний коефіцієнт** для оцінки якості кластеризації (значення від -1 до 1, чим ближче до 1, тим краще).
   - Будується **візуалізація** результатів кластеризації (діаграма розсіювання) для перших двох (масштабованих) ознак.
   - Проводиться аналіз відповідності знайдених кластерів реальним класам за допомогою pd.crosstab.
5. **Класифікація:**
   - Дані (масштабовані ознаки X_scaled та оригінальні мітки y) **розділяються** на навчальну та тестову вибірки за допомогою train_test_split. Параметр stratify=y важливий для збереження пропорцій класів в обох вибірках.
   - **Ініціалізується та навчається** модель DecisionTreeClassifier. Параметр max_depth допомагає уникнути перенавчання моделі.
   - Робляться **прогнози** на тестовій вибірці (predict).
   - Проводиться **оцінка якості**: обчислюється accuracy_score, будується та візуалізується confusion_matrix (матриця помилок), виводиться детальний classification_report (з Precision, Recall, F1-score).
   - **Візуалізується** саме дерево рішень за допомогою plot_tree.
6. **Висновки:** Надається приклад текстового опису отриманих результатів та їх можливого застосування.

Домашнє завдання

1. Обрати датасет (один із варіантів):
   - Wine або Breast Cancer (sklearn),

- або власний CSV (не менше 200 записів, 4+ числових ознак).

2. Для кластеризації:
   - виконати k-means;
   - обґрунтувати k методом “ліктя” та силуетним коефіцієнтом (показати обидві візуалізації/значення);
   - описати кластери (розмір + середні значення ознак).
3. Для класифікації:
   - реалізувати 2 моделі (наприклад, Decision Tree та SVM або Decision Tree та k-NN);
   - порівняти їх за Accuracy і F1-score (macro або weighted);
   - зробити висновок, яка модель краща і чому.
4. Додатково (за бажанням, +бонус):
   - виконати PCA до 2D та порівняти візуально класи/кластери у просторі компонент.

Рекомендації щодо оформлення звіту

1. Титульний аркуш (за зразком кафедри).
2. Тема і мета лабораторної роботи.
3. Короткі теоретичні відомості (кластеризація, класифікація, метрики якості, нормалізація).
4. Вхідні дані:
   - джерело датасету;
   - опис ознак і цільової змінної;
   - кількість записів, наявність/відсутність пропусків.
5. Попередня обробка:
   - які операції виконано (імпутація/видалення пропусків, масштабування);
   - обґрунтування необхідності стандартизації.
6. Хід роботи та результати:
   - кластеризація: графік “ліктя”, вибір k, silhouette score, візуалізація кластерів, аналіз кластерів;
   - класифікація: train/test split, модель, confusion matrix, classification report, візуалізація (дерево або інше).
7. Аналіз результатів:
   - інтерпретація кластерів;
   - аналіз помилок класифікації;
   - приклад використання в СППР.
8. Висновки.
9. Відповіді на контрольні питання.
10. Список літератури (ДСТУ 8302:2015).
11. Додатки (за потреби): повний лістинг коду, додаткові графіки.

Контрольні питання

1. Дайте визначення кластеризації та наведіть приклад її застосування.
2. Дайте визначення класифікації та наведіть приклад її застосування.
3. У чому полягає ключова відмінність між завданнями кластеризації та класифікації?
4. Поясніть принцип роботи методу k-means. Як обирається кількість кластерів 'k'?
5. Що таке матриця помилок (confusion matrix) і які метрики якості класифікації можна з неї отримати?
6. Поясніть суть метрик Accuracy, Precision, Recall та F1-Score. Коли варто надавати перевагу тій чи іншій метриці?
7. Навіщо потрібна попередня обробка даних (зокрема, стандартизація/нормалізація)?
8. Що таке силуетний коефіцієнт і для чого він використовується?
9. Наведіть приклад, як результати кластеризації даних про клієнтів можуть допомогти у прийнятті маркетингових рішень.

10. Наведіть приклад, як модель класифікації може використовуватися в системі підтримки прийняття медичних рішень.

Література

1. Han J., Kamber M., Pei J. Data Mining: Concepts and Techniques. – 3rd ed. – Waltham : Morgan Kaufmann, 2012. – 703 p.
2. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. – 2nd ed. – New York : Springer, 2009. – 745 p.
3. James G., Witten D., Hastie T., Tibshirani R. An Introduction to Statistical Learning: with Applications in R. – New York : Springer, 2013. – 426 p.
4. Bishop C. M. Pattern Recognition and Machine Learning. – New York : Springer, 2006. – 738 p.
5. Aggarwal C. C. Data Mining: The Textbook. – Cham : Springer, 2015. – 734 p.
6. Pedregosa F. та ін. Scikit-learn: Machine Learning in Python // Journal of Machine Learning Research. – 2011. – Vol. 12. – P. 2825–2830.

Варіанти

Варіант 1. Breast Cancer Wisconsin (Diagnostic)

```
from sklearn.datasets import load_breast_cancer
data = load_breast_cancer()
X = data.data
y = data.target
```

- **X** містить 30 числових ознак;
- **y** містить бінарні мітки (0 або 1) – доброякісна чи злоякісна пухлина.

Варіант 2. Digits

```
from sklearn.datasets import load_digits
data = load_digits()
X = data.data
y = data.target
```

- **X** містить 64 числові ознаки для кожного зображення (розміром 8×8);
- **y** містить цілі від 0 до 9, що відповідають рукописним цифрам.

Варіант 3. Wine

```
from sklearn.datasets import load_wine
data = load_wine()
X = data.data
y = data.target
```

- **X** містить 13 числових ознак (хімічні показники вина);
- **y** містить 3 класи (три сорти вина).

Варіант 4. Iris

```
from sklearn.datasets import load_iris
data = load_iris()
X = data.data
y = data.target
```

- **X** містить 4 числові ознаки (довжина/ширина чашолистків і пелюсток);
- **y** містить 3 класи (три види ірисів).

Варіант 5. make_blobs

```
from sklearn.datasets import make_blobs
# Приклад: 300 точок із 3 центрами, кожна точка має 2 ознаки
X, y = make_blobs(n_samples=300, centers=3, n_features=2, random_state=42)
```

Варіант 6. make_moons

```
from sklearn.datasets import make_moons
# Приклад: 200 точок, шум 0.05, 2D-простір
X, y = make_moons(n_samples=200, noise=0.05, random_state=42)
```

Варіант 7. make_classification

```
from sklearn.datasets import make_classification
# Приклад: 200 об'єктів, 5 ознак, з яких 3 інформативні, 1 дубльована і 1 "шумова"
X, y = make_classification(n_samples=200,
                           n_features=5,
                           n_informative=3,
                           n_redundant=1,
                           n_classes=2,
                           random_state=42)
```

Лабораторна робота №4. Прогнозування у система підтримки прийняття рішень на основі часових рядів

Мета роботи: побудова прогнозів для аналізу попиту, оцінки ризику або визначення витрат за допомогою таких методів, як просте прогнозування, ковзне середнє та експоненціальне згладжування.

Короткі теоретичні положення

Основні поняття

- **Часовий ряд** – послідовність вимірювань, зроблених через рівні проміжки часу. До основних компонентів відносять:
  - **Тренд:** загальна напрямність ряду (зростання або спадання).
  - **Сезонність:** періодичність у змінності даних.
  - **Випадковість (шум):** нерегулярні коливання, що не описуються систематичними законами.

Методи прогнозування

- **Просте прогнозування:**
  - Використання останнього спостереження як прогнозу для наступного періоду.
  - Застосовується як базова модель порівняння точності при використанні більш складних методів.
- **Метод ковзного середнього:**
  - Розрахунок прогнозу як середнього арифметичного значень за фіксованим «вікном» останніх n спостережень.
  - **Вплив параметра вікна:** збільшення розміру вікна робить прогноз більш гладким, однак може сповільнити реакцію на зміни.
- **Експоненціальне згладжування:**
  - Призначене для більшої ваги останнім спостереженням, які оцінюються за допомогою коефіцієнта згладжування (α).
  - Формула для простого експоненціального згладжування:

$$S_t = \alpha \cdot Y_t + (1 - \alpha) \cdot S_{t-1}$$

де S_t – згладжене значення на момент t, Y_t – фактичне спостереження, а α – коефіцієнт згладжування ($0 < \alpha \leq 1$).

Завдання до роботи

Необхідно виконати наступні завдання:

- **Підготовка даних:**
  - Завантажити та попередньо обробити дані часових рядів (очищення, заповнення пропусків, перетворення даних при необхідності).
  - Провести візуальний аналіз даних (побудова графіку початкового часового ряду).
- **Реалізація простого прогнозування:**
  - Використання останнього значення ряду як прогноз для майбутнього періоду.
  - Розрахунок базових статистичних показників та порівняння з фактичними даними.
- **Прогнозування методом ковзного середнього:**
  - Реалізувати обчислення ковзного середнього для вибраних розмірів вікна (наприклад, 3, 5, 7 періодів).
  - Побудувати графік, на якому відображено як фактичні дані, так і отримані прогнозні значення.
- **Реалізація експоненціального згладжування:**
  - Застосувати функціонал пакету **statsmodels** (наприклад, клас SimpleExpSmoothing)

для побудови моделі.
- Визначити оптимальне значення параметра згладжування α.
- Побудувати графік прогнозу та порівняти його з фактичними даними.

- **Оцінка точності прогнозування:**
  - Розбити дані на тренувальну та тестову вибірки.
  - Обчислити метричні показники якості прогнозу: MAE (середня абсолютна помилка), MSE (середньоквадратична помилка), RMSE (корінь квадратного середнього відхилення).
  - Проаналізувати залишки (різниця між прогнозованими і фактичними значеннями).
- **Візуалізація даних:**
  - Побудувати графіки для порівняння методів прогнозування (наприклад, з використанням бібліотеки matplotlib).
  - Відобразити результати аналізу залишків (для перевірки наявності автокореляції).

Опис способу обробки результатів, хід експерименту

Обчислення метрік якості прогнозування

- **Розбиття даних:**
  - Дані слід поділити на тренувальну вибірку (для побудови моделі) та тестову вибірку (для оцінки точності прогнозу).
- **Обчислення показників:**
  - **MAE (Mean Absolute Error):**

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|Y_i - \hat{Y}_i|$$

  - **MSE (Mean Squared Error):**

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

  - **RMSE (Root Mean Squared Error):**

$$RMSE = \sqrt{MSE}$$

Візуалізація

2. **Графічне порівняння:**
   1. Побудова одного або декількох графіків, де представлені фактичні значення та значення, отримані прогнозними моделями.
   2. Побудова графіків залишків для аналізу відхилень та виявлення можливих закономірностей.

Аналіз залишків

4. Перевірка наявності автокореляції залишків (за допомогою графіків ACF і PACF) для оцінки якості моделі.
5. Аналіз розподілу залишків для виявлення систематичних похибок.

Оптимізація параметрів

2. Для методу експоненціального згладжування важливо підібрати оптимальне значення α. Це можна зробити:
   1. За допомогою перебору значень параметра та вибору того, що мінімізує обрану метрику.
   2. За допомогою вбудованих методів оптимізації, наприклад, функцій з пакету statsmodels.

Приклад виконання

Крок 1. Підготовка даних

Завантаження стандартного датасету та попередня обробка

5. Імпортуємо необхідні бібліотеки та завантажуємо дані. У цьому прикладі використовується датасет «AirPassengers» з пакету **statsmodels.tsa.datasets**.
6. Перетворюємо стовпець з датами у формат datetime і встановлюємо його як індекс, що дозволить легко працювати з часовим рядом.
7. Виводимо перші кілька рядків для перевірки коректності завантаження даних.

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
import statsmodels.api as sm

# Завантаження датасету "AirPassengers" з R-датасетів
data = sm.datasets.get_rdataset("AirPassengers", "datasets").data

# Переглянемо, які стовпці містить датасет
print("Стовпці датасету:", data.columns.tolist())

# Вибираємо потрібний стовпець та створюємо копію
data = data[['value']].copy()

# Перейменовуємо стовпець для зручності
data.columns = ['#Passengers']

# Створення datetime-індексу.
# Використовуємо частоту 'ME' (кінець місяця)
dates = pd.date_range(start='1949-01-01', periods=len(data), freq='ME')
data.index = dates

# Переглядаємо перші рядки датасету для перевірки
print(data.head())
```

Попередній аналіз:

6. Перевірте, чи є пропущені значення або аномалії. Наприклад, можна використовувати:

```
print(data.info())
print(data.describe())
```

7. Побудуйте графік початкового часового ряду:

```
plt.figure(figsize=(10, 5))
plt.plot(data['#Passengers'], marker='o')
plt.title("Часовий ряд: кількість пасажирів (AirPassengers)")
plt.xlabel("Дата")
plt.ylabel("Кількість пасажирів")
plt.grid(True)
plt.show()
```

Крок 2. Розбиття даних на тренувальну та тестову вибірки

Для оцінки моделей прогнозування дані розбивають таким чином, що останні 12 місяців використовуються як тестова вибірка:

```
# Розбиття даних: тренувальна вибірка без останнього року, тест – останні 12 місяців
train = data.iloc[:-12]
test = data.iloc[-12:]
print("Кількість спостережень тренувальної вибірки:", len(train))
print("Кількість спостережень тестової вибірки:", len(test))
```

Крок 3. Просте прогнозування

Проста модель прогнозування:

Прогноз за кожним кроком робиться за останнім значенням тренувальних даних. Для прогнозу на тестовий період просто повторюємо останнє спостереження тренувальної вибірки.

```
# Отримуємо останнє спостереження тренувальної вибірки
last_value = train['#Passengers'].iloc[-1]
# Прогноз на весь тестовий період - повторюємо останнє значення
simple_forecast = [last_value] * len(test)

print("Останнє значення тренувальної вибірки:", last_value)
```

Крок 4. Прогнозування методом ковзного середнього

Розрахунок ковзного середнього:

5. Обчислюємо ковзне середнє для всього датасету із вибраним розміром вікна. Тут використаємо вікно у 12 періодів.
6. Для прогнозування тестового періоду, використаємо середнє останнього вікна тренувальної вибірки (це базова реалізація).

```
window_size = 12

# Обчислення ковзного середнього для всього датасету
data['moving_avg'] = data['#Passengers'].rolling(window=window_size).mean()

# Візуалізація фактичних даних та ковзного середнього
plt.figure(figsize=(10, 5))
plt.plot(data['#Passengers'], label='Фактичні дані', marker='o')
plt.plot(data['moving_avg'], label='Ковзне середнє (window=12)', color='red')
plt.title("Метод ковзного середнього")
plt.xlabel("Дата")
plt.ylabel("Кількість пасажирів")
plt.legend()
plt.grid(True)
plt.show()

# Прогноз для тестового періоду: використання середнього останніх 12 спостережень з тренувальної вибірки
last_window_avg = train['#Passengers'].tail(window_size).mean()
moving_avg_forecast = [last_window_avg] * len(test)
print("Середнє значення останнього вікна тренувальної вибірки:", last_window_avg)
```

Крок 5. Прогнозування методом експоненціального згладжування

Застосування експоненціального згладжування:

- Створюємо модель експоненціального згладжування для тренувальної вибірки за допомогою класу SimpleExpSmoothing з пакету **statsmodels**.
- Оптимізуємо параметр згладжування (α) або встановлюємо його вручну (тут для прикладу використовується початкове значення 0.2).
- Прогнозуємо значення для тестового періоду.

```
from statsmodels.tsa.holtwinters import SimpleExpSmoothing

# Побудова моделі експоненціального згладжування на тренувальній вибірці
ses_model = SimpleExpSmoothing(train['#Passengers']).fit(smoothing_level=0.2, optimized=True)
# Прогнозування для тестового періоду (кількість періодів = довжина тестової вибірки)
ses_forecast = ses_model.forecast(len(test))

# Додавання прогнозованих значень до тестової вибірки для порівняння
test = test.copy()  # щоб уникнути попереджень про копіювання
test['ses_forecast'] = ses_forecast

# Візуалізація прогнозу експоненціального згладжування
plt.figure(figsize=(10, 5))
plt.plot(train['#Passengers'], label='Тренувальні дані', marker='o')
plt.plot(test['#Passengers'], label='Фактичні дані (тест)', marker='o')
plt.plot(test['ses_forecast'], label='Прогноз (SES)', marker='o', color='green')
plt.title("Метод експоненціального згладжування")
plt.xlabel("Дата")
plt.ylabel("Кількість пасажирів")
plt.legend()
plt.grid(True)
plt.show()
```

Крок 6. Оцінка точності прогнозування

Обчислення метрик прогнозування: MAE, MSE, RMSE

Для обчислення помилок використовуємо бібліотеку **sklearn.metrics**.

```
from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np

# Обчислення показників для простого прогнозування
mae_simple = mean_absolute_error(test['#Passengers'], simple_forecast)
mse_simple = mean_squared_error(test['#Passengers'], simple_forecast)
rmse_simple = np.sqrt(mse_simple)
print("Просте прогнозування:")
print("MAE =", mae_simple)
print("MSE =", mse_simple)
print("RMSE =", rmse_simple)

# Обчислення показників для методу ковзного середнього
mae_moving = mean_absolute_error(test['#Passengers'], moving_avg_forecast)
mse_moving = mean_squared_error(test['#Passengers'], moving_avg_forecast)
rmse_moving = np.sqrt(mse_moving)
print("\nПрогнозування методом ковзного середнього:")
print("MAE =", mae_moving)
print("MSE =", mse_moving)
print("RMSE =", rmse_moving)

# Обчислення показників для експоненціального згладжування
mae_ses = mean_absolute_error(test['#Passengers'], ses_forecast)
mse_ses = mean_squared_error(test['#Passengers'], ses_forecast)
rmse_ses = np.sqrt(mse_ses)
print("\nПрогнозування методом експоненціального згладжування:")
print("MAE =", mae_ses)
print("MSE =", mse_ses)
print("RMSE =", rmse_ses)
```

Крок 7. Аналіз та візуалізація залишків

Аналіз залишків:

Обчислюємо залишки (різниця між фактичними та прогнозованими значеннями) для моделі експоненціального згладжування та будуємо графік для візуального аналізу.

```
# Залишки для методу експоненціального згладжування
residuals = test['#Passengers'] - ses_forecast

plt.figure(figsize=(10, 5))
plt.plot(residuals, marker='o')
plt.title("Залишки (фактичні значення - прогноз SES)")
plt.xlabel("Дата")
plt.ylabel("Залишки")
plt.axhline(0, color='black', linestyle='--')
plt.grid(True)
plt.show()

# Для більш глибокого аналізу можна побудувати автокореляційну функцію (ACF)
from statsmodels.graphics.tsaplots import plot_acf
plot_acf(residuals, lags=len(residuals)-1)
plt.title("ACF залишків")
plt.show()
```

Домашнє завдання

1. Обрати інший часовий ряд, відмінний від прикладу AirPassengers (можливі варіанти):
    - щомісячні продажі товару;
    - щотижневе споживання електроенергії;
    - динаміка витрат або доходів;
    - курси валют або біржові індекси.
2. Підготувати часовий ряд:
    - перевірити наявність пропусків;
    - виконати попередню обробку (за потреби).
3. Реалізувати три методи прогнозування:
    - просте прогнозування;
    - метод ковзного середнього (мінімум два різні розміри вікна);
    - експоненціальне згладжування.
4. Розбити дані на тренувальну і тестову вибірки.
5. Обчислити метрики якості прогнозу: MAE, MSE, RMSE для кожного методу.
6. Порівняти результати та:
    - визначити метод з найменшою похибкою;
    - пояснити отримані відмінності.
7. Сформулювати короткий висновок щодо можливості використання отриманого прогнозу в системі підтримки прийняття рішень.

Рекомендації щодо оформлення звіту

Звіт з лабораторної роботи оформлюється на аркушах формату А4 та повинен містити такі розділи:

1. Титульний аркуш (згідно з вимогами кафедри / університету).
2. Тема та мета лабораторної роботи.
3. Короткі теоретичні положення:
    - поняття часового ряду;
    - основні методи прогнозування;
    - показники якості прогнозу.
4. Вхідні дані:
    - джерело часового ряду;
    - кількість спостережень;
    - періодичність (день, місяць, рік тощо).
5. Хід виконання роботи:
    - побудова початкового графіка ряду;
    - реалізація методів прогнозування;
    - розбиття на тренувальну і тестову вибірки;
    - обчислення метрик точності.
6. Результати та аналіз:
    - порівняльна таблиця метрик MAE, MSE, RMSE;
    - графічне порівняння прогнозів;
    - аналіз залишків.
7. Висновки:
    - оцінка якості прогнозування;
    - практична цінність прогнозу для СППР.
8. Список використаної літератури.

Вимоги до оформлення:

- шрифт Times New Roman, 14 pt;
- міжрядковий інтервал — 1.5;
- усі графіки та таблиці мають бути підписані.

Контрольні питання

1. Що таке часовий ряд та які його основні компоненти?
   (Відповідь повинна містити опис тренду, сезонності, циклічності та випадковості.)
2. Які відмінності між простим прогнозуванням, методом ковзного середнього та експоненціальним згладжуванням?
   (Поясніть, як кожен метод використовує попередні значення для формування прогнозу.)

3. Яким чином впливає розмір вікна в методі ковзного середнього?
   (Які переваги та недоліки збільшення або зменшення розміру вікна?)
4. Як здійснюється оптимізація параметра згладжування α у методі експоненціального згладжування?
   (Наведіть приклади методів або алгоритмів для добору оптимального α.)
5. Які метрики використовуються для оцінки якості прогнозування (наприклад, MAE, MSE, RMSE) і як їх інтерпретувати?
6. Що представляють собою графіки залишків і яку інформацію вони надають щодо точності моделі?
7. Які обмеження мають застосовані методи прогнозування в контексті СППР?

Література

1. Hyndman R. J., Athanasopoulos G. Forecasting: Principles and Practice. – 2nd ed. – Melbourne : OTexts, 2018. – 382 p.
2. Box G. E. P., Jenkins G. M., Reinsel G. C., Ljung G. M. Time Series Analysis: Forecasting and Control. – 5th ed. – Hoboken : Wiley, 2015. – 712 p.
3. Makridakis S., Wheelwright S. C., Hyndman R. J. Forecasting: Methods and Applications. – 3rd ed. – New York : Wiley, 1998. – 656 p.
4. Chatfield C. The Analysis of Time Series: An Introduction. – 6th ed. – Boca Raton : CRC Press, 2004. – 333 p.
5. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. – 2nd ed. – New York : Springer, 2009. – 745 p.
6. Hyndman R. J. Forecasting with Exponential Smoothing: The State Space Approach. – Berlin : Springer, 2008. – 372 p.

Варіанти

Варіант 1: Аналіз сонячних плям ("sunspots")

Завдання:

4. Проведіть аналіз часових рядів даних про кількість сонячних плям.
5. Побудуйте графік часового ряду, визначте циклічність та зробіть прогноз майбутніх значень за допомогою методів експоненціального згладжування або ARIMA.

Приклад завантаження даних:

```
import pandas as pd
import statsmodels.api as sm

# Завантаження
data = sm.datasets.get_rdataset("sunspots", "datasets").data
series = data['value'].copy()

# Створення datetime-індексу:
# дані — щомісячні спостереження починаючи з 1749-01
dates = pd.date_range(start='1749-01', periods=len(series), freq='M')
series.index = dates
series.name = 'Sunspots'

print(series.head())
```

Варіант 2: Аналіз даних про вилов лисиць ("lynx")

Завдання:

- Проаналізуйте щорічні дані про вилов лисиць у Канаді.
- Визначте тренди та можливу циклічність, побудуйте модель прогнозування (наприклад, з використанням простого ковзного середнього або експоненціального згладжування).

Приклад завантаження даних:

```
import pandas as pd
import statsmodels.api as sm

# Завантаження
```

```
data = sm.datasets.get_rdataset("lynx", "datasets").data
series = data['value'].copy()

# Щорічні дані з 1821 року (кінець року)
dates = pd.date_range(start='1821-01', periods=len(series), freq='A')
series.index = dates
series.name = 'Lynx Trappings'

print(series.head())
```

Варіант 3: Аналіз температури повітря в Ноттінгемі ("nottem")

Завдання:

3. Проведіть дослідження часових рядів середньої температури повітря в Ноттінгемі (місячні дані).
4. Проаналізуйте сезонність та визначте моделі прогнозування (наприклад, експоненціальне згладжування або Holt-Winters).

Приклад завантаження даних:

```
import pandas as pd
import statsmodels.api as sm

# Завантаження
data = sm.datasets.get_rdataset("nottem", "datasets").data
series = data['value'].copy()

# Щомісячні дані починаючи з 1920-01
dates = pd.date_range(start='1920-01', periods=len(series), freq='M')
series.index = dates
series.name = 'Nottingham Air Temp'

print(series.head())
```

Варіант 4: Аналіз рівня CO_2 (набор даних "CO2")

Завдання:

- Проаналізуйте часовий ряд даних про концентрацію вуглекислого газу в атмосфері.
- Визначте тренди та сезонні коливання, побудуйте модель прогнозування.

Приклад завантаження даних:

```
import pandas as pd
import statsmodels.api as sm

# Завантаження
data = sm.datasets.get_rdataset("CO2", "datasets").data
series = data['value'].copy()

# Щомісячні дані починаючи з 1959-01
dates = pd.date_range(start='1959-01', periods=len(series), freq='M')
series.index = dates
series.name = 'CO2 (ppm)'

print(series.head())
```

Примітка: Зверніть увагу, що у цьому наборі даних може бути кілька змінних; необхідно вибрати відповідну колонку для часової моделі (наприклад 'uptake').

Варіант 5: Аналіз течії Нілу ("Nile")

Завдання:

- Дослідіть щорічні дані про течію річки Ніл.
- Побудуйте модель прогнозування (наприклад, ARIMA або експоненціальне згладжування) та проаналізуйте наявність тренду.

Приклад завантаження даних:

```
import pandas as pd
import statsmodels.api as sm

# Завантаження
data = sm.datasets.get_rdataset("Nile", "datasets").data
series = data['value'].copy()

# Щорічні дані з 1871 року
dates = pd.date_range(start='1871-01', periods=len(series), freq='A')
series.index = dates
series.name = 'Nile Flow'

print(series.head())
```

Варіант 6: Аналіз фінансових даних компанії ("JohnsonJohnson")

Завдання:

- Проаналізуйте квартальні дані про прибутки компанії Johnson & Johnson.
- Визначте сезонні коливання та тренди, розробіть модель прогнозування.

Приклад завантаження даних:

```
import pandas as pd
import statsmodels.api as sm

# Завантаження
data = sm.datasets.get_rdataset("JohnsonJohnson", "datasets").data
series = data['value'].copy()

# Щоквартальні дані з 1960 Q1
dates = pd.date_range(start='1960-01', periods=len(series), freq='Q')
series.index = dates
series.name = 'J&J Quarterly Earnings'

print(series.head())
```

Варіант 7: Аналіз продажів (BJsales)

Завдання:

- Проаналізуйте місячні дані про продажі компанії, що представлені у наборі даних BJsales.
- Побудуйте графік часового ряду, визначте сезонні патерни та створіть модель прогнозування із застосуванням одного чи кількох методів.

Приклад завантаження даних:

```
import pandas as pd
import statsmodels.api as sm

# Завантаження
data = sm.datasets.get_rdataset("BJsales", "datasets").data
series = data['value'].copy()

# Щомісячні дані з 1971-01
dates = pd.date_range(start='1971-01', periods=len(series), freq='M')
series.index = dates
series.name = 'BJ Sales'

print(series.head())
```

Лабораторна робота №5. Розробка модельно-орієнтованої системи підтримки прийняття рішень

Мета роботи: Освоїти методику створення модельно-орієнтованої системи підтримки прийняття рішень. Навчитися використовувати математичні моделі для оптимізації процесу прийняття рішень.

Короткі теоретичні положення

Системи підтримки прийняття рішень (СППР) — це інтерактивні інформаційні системи, які допомагають особам або організаціям приймати обґрунтовані рішення у складних або невизначених ситуаціях. СППР поєднують в собі бази даних, аналітичні моделі, інтерфейси користувача і спеціальні методи обробки даних.
Одним із підходів до побудови СППР є **модельно-орієнтовані СППР** — це така система, в основі якої лежить **математична або імітаційна модель**, яка дозволяє аналізувати можливі рішення на основі формалізованого опису реальної ситуації.
Основні особливості модельно-орієнтованих СППР:

1. Фокус на використанні **моделей** (оптимізаційних, статистичних, економічних, прогнозних).
2. Динамічне оновлення моделі при зміні вхідних даних.
3. Проведення **експериментів** із різними сценаріями для вибору найкращого варіанту рішення.

Приклади моделей:

1. Лінійне програмування (лінійна оптимізація витрат або прибутку)
2. Цілочисельне програмування (задачі вибору або маршрутизації)
3. Квадратичне програмування (оптимізація портфеля інвестицій)
4. Статистичні моделі прогнозування попиту чи ризиків
5. Імітаційне моделювання процесів обслуговування або виробництва

Основні компоненти моделі у СППР

1. **Змінні рішення** — Параметри, значення яких ми можемо змінювати для досягнення мети (наприклад, кількість продукції, обсяг інвестицій, кількість ресурсів).
2. **Цільова функція** — Формалізоване у вигляді математичного виразу правило оптимізації (мінімізації або максимізації деякого показника: витрат, часу, ризику, прибутку).
3. **Обмеження** — Умови, які обмежують простір можливих рішень (наприклад, обмежені ресурси, мінімальні або максимальні обсяги виробництва, вимоги до якості або термінів).
4. **Вхідні дані** — Фактичні значення параметрів: ціни, запаси ресурсів, вимоги клієнтів, характеристики процесів тощо.
5. Загальний процес роботи модельно-орієнтованої СППР
6. **Формулювання задачі**: визначення мети, змінних, обмежень та цільової функції.
7. **Побудова математичної моделі**: запис задачі у вигляді рівнянь і нерівностей.
8. **Розв'язання моделі**: застосування методів оптимізації для знаходження оптимального рішення.
9. **Аналіз результатів**: перевірка рішень на доцільність, чутливість до зміни параметрів.
10. **Прийняття рішення**: вибір найкращої альтернативи на основі аналізу моделі.

Типи задач, що розв’язуються за допомогою модельно-орієнтованих СППР

Тип задачі	Приклад
Оптимізаційні задачі	Мінімізація витрат виробництва при обмежених ресурсах
Задачі вибору	Вибір найкращого постачальника за ціною і якістю
Задачі розподілу	Розподіл бюджету між різними каналами реклами
Прогнозування	Оцінка попиту на товар у майбутньому
Планування	Складання маршруту доставки з мінімальними витратами часу або

Тип задачі	Приклад
	пального
Імітаційні задачі	Моделювання роботи складу чи сервісного центру

Переваги використання моделей у СППР

1. **Швидкість обробки** великої кількості альтернатив.
2. **Об'єктивність**: мінімізація впливу людського фактора.
3. **Гнучкість**: можливість легко змінювати параметри моделі і отримувати нові результати.
4. **Економічність**: виявлення рішень, які мінімізують витрати або ризики.
5. **Сценарний аналіз**: дослідження «що буде, якщо...» при зміні початкових умов.

Інструменти для реалізації модельно-орієнтованої СППР

1. **Python**:
    1. scipy.optimize.linprog — для задач лінійного програмування.
    2. PuLP, OR-Tools — для задач цілочисельного програмування.
    3. cvxopt, quadprog — для задач квадратичного програмування.
    4. matplotlib, seaborn — для побудови графіків результатів.
2. **Інші системи**:
    1. MATLAB Optimization Toolbox
    2. Excel Solver
    3. OpenSolver (доповнення до Excel)
    4. IBM CPLEX Optimization Studio
    5. Google OR-Tools

У цій лабораторній роботі ми побудуємо спрощену модель оптимізації витрат виробництва за умов обмеження ресурсів і попиту, реалізуємо її у Python, проведемо експериментальний аналіз і візуалізуємо результати.

Завдання до роботи

- Ознайомитись з поняттям модельно-орієнтованої СППР.
- Створити в Google Colab власну спрощену модель прийняття рішень.
- Використовуючи Python, побудувати та проаналізувати модель:
    - Модель оптимізації (напр., задача мінімізації витрат або максимізації прибутку).
    - Провести експеримент з варіацією вхідних даних.
- Візуалізувати результати роботи моделі (графіки, таблиці).
- Підготувати звіт за результатами роботи.

Опис способу обробки результатів, хід експерименту

Результати лабораторної роботи отримуються шляхом розв’язання оптимізаційної моделі для заданого набору вхідних параметрів та подальшого аналізу отриманого рішення.

У ході експерименту:

1. Обчислюється оптимальне значення змінних рішення для базових вхідних даних.
2. Аналізується значення цільової функції (витрати або прибуток) в оптимальній точці.
3. Проводиться аналіз чутливості шляхом варіації окремих параметрів моделі (наприклад, мінімального попиту або ресурсних обмежень).
4. Результати подаються у вигляді числових значень та графічних ілюстрацій (область допустимих рішень, положення оптимуму).
5. На основі аналізу формулюються рекомендації щодо прийняття управлінського рішення.

Приклад виконання

Крок 1. Формулювання задачі

6. **Проблема:** потрібно мінімізувати витрати на виробництво двох продуктів A і B за умови, що сумарний обсяг випуску не менше 50 одиниць.
7. **Змінні рішення:**

1. x — кількість одиниць продукту A
2. y — кількість одиниць продукту B

8. **Цільова функція:**

$$\min\ 40x + 30y$$

9. **Обмеження:**

$$\begin{cases} 2x + y \le 100, \\ x + 2y \le 80, \\ x + y \ge 50, \\ x \ge 0,\ y \ge 0. \end{cases}$$

Крок 2. Математичне формулювання в Python

```
from scipy.optimize import linprog

# 1) коефіцієнти цільової функції
c = [40, 30]

# 2) нерівності у вигляді A_ub x <= b_ub:
A_ub = [
   [ 2,  1],   # 2x +  y <= 100
   [ 1,  2],   #  x + 2y <=  80
   [-1, -1]    # -x - y <= -50  →  x + y >= 50
]
b_ub = [100, 80, -50]

# 3) межі змінних (0 ≤ x, y < ∞)
bounds = [(0, None), (0, None)]
```

Крок 3. Розв’язання задачі оптимізації

```
result = linprog(c, A_ub=A_ub, b_ub=b_ub, bounds=bounds, method='highs')

if result.success:
   x_opt, y_opt = result.x
   print(f"Оптимальне рішення: x = {x_opt:.2f}, y = {y_opt:.2f}")
   print(f"Мінімальні витрати = {result.fun:.2f}")
else:
   print("Рішення не знайдено:", result.message)
```

Крок 4. Аналіз чутливості (варіація попиту)

Перевіримо, як зміниться розв’язок, якщо мінімальний попит зросте до Dmin=60D_{\min}=60:

```
# Оновимо обмеження попиту
b_ub_mod = [100, 80, -60]  # -x - y <= -60 → x + y >= 60

res_mod = linprog(c, A_ub=A_ub, b_ub=b_ub_mod, bounds=bounds, method='highs')

if res_mod.success:
   print("При Dmin=60 → x =", res_mod.x[0], "y =", res_mod.x[1],
         "витрати =", res_mod.fun)
else:
   print("Рішення не знайдено при Dmin=60.")
```

Крок 5. Візуалізація області та оптимуму

```
import numpy as np
import matplotlib.pyplot as plt

# Границі графіка
x = np.linspace(0, 60, 300)
```

```
y1 = (100 - 2*x)      # 2x + y = 100
y2 = (80 - x)/2       # x + 2y = 80
y3 = 50 - x           # x + y = 50 (Dmin)

# Обчислимо обмеження на y
y_upper = np.minimum(y1, y2)
y_lower = y3

plt.figure(figsize=(7,7))
plt.plot(x, y1, label='2x + y = 100')
plt.plot(x, y2, label='x + 2y = 80')
plt.plot(x, y3, label='x + y = 50 (Dmin)')

# Реальне визначення допустимої області
feasible_region = (y_lower <= y_upper) & (y_upper >= 0) & (y_lower >= 0)

# Заповнення між y_lower і y_upper лише там, де область допустима
plt.fill_between(x, y_lower, y_upper, where=feasible_region,
                 color='lightblue', alpha=0.5)

# Нанесемо оптимум
plt.scatter(x_opt, y_opt, color='red', zorder=5)
plt.text(x_opt+1, y_opt+1, f'Оптимум ({x_opt:.1f},{y_opt:.1f})')

plt.xlim(0, 60)
plt.ylim(0, 60)
plt.xlabel('x (A)')
plt.ylabel('y (B)')
plt.title('Область допустимих рішень та оптимум')
plt.legend()
plt.grid(True)
plt.show()
```

Крок 6. Формулювання висновків

3. **Яке оптимальне рішення** задовольняє мінімальний попит та мінімізує витрати.
4. **Як зміна попиту** (D□ᵢ□) впливає на обсяги виробництва та загальні витрати.
5. **Рекомендації:** якщо бюджет обмежений, варто коригувати попит або шукати способи знизити витрати на одиницю продукції.

Домашнє завдання

1. Сформулювати власну задачу оптимізації для модельно-орієнтованої СППР (один із варіантів):
   - максимізація прибутку від виробництва;
   - мінімізація витрат при обмежених ресурсах;
   - оптимальний розподіл бюджету між напрямами діяльності;
   - вибір оптимального набору проєктів.
2. Визначити:
   - змінні рішення;
   - цільову функцію;
   - систему обмежень.
3. Реалізувати модель у Python (scipy.optimize, PuLP або OR-Tools).
4. Знайти оптимальне рішення для базового набору параметрів.
5. Провести аналіз чутливості (зміна одного або двох параметрів).
6. Побудувати хоча б один графік, що ілюструє зміну цільової функції або області допустимих рішень.
7. Сформулювати короткі висновки щодо практичного використання моделі в СППР.

Рекомендації щодо оформлення звіту

Звіт з лабораторної роботи оформлюється на аркушах формату А4 та повинен містити:

1. Титульний аркуш (за вимогами кафедри).
2. Тему та мету роботи.
3. Короткі теоретичні положення:
    - поняття модельно-орієнтованої СППР;
    - типи математичних моделей.
4. Постановку задачі:
    - опис проблеми;
    - змінні рішення;
    - цільову функцію;
    - обмеження.
5. Хід виконання роботи:
    - реалізація моделі в Python;
    - отримання оптимального рішення.
6. Результати та аналіз:
    - значення змінних та цільової функції;
    - результати аналізу чутливості;
    - графічні матеріали.
7. Висновки та рекомендації.
8. Список використаної літератури.

Контрольні питання

1. Що таке модельно-орієнтована СППР?
2. Які типи моделей найчастіше використовуються в СППР?
3. Як формулюється задача оптимізації?
4. Для чого проводиться аналіз чутливості в СППР?
5. Які бібліотеки Python дозволяють розв'язувати задачі оптимізації?
6. Які переваги має використання моделей у прийнятті рішень?
7. Які етапи включає розробка модельно-орієнтованої СППР?

Література

1. Hillier F. S., Lieberman G. J. Introduction to Operations Research. – 9th ed. – New York : McGraw-Hill, 2010. – 1216 p.
2. Winston W. L. Operations Research: Applications and Algorithms. – 4th ed. – Belmont : Thomson Brooks/Cole, 2004. – 1418 p.
3. Taha H. A. Operations Research: An Introduction. – 10th ed. – Boston : Pearson Education, 2017. – 832 p.
4. Turban E., Sharda R., Delen D. Decision Support and Business Intelligence Systems. – 9th ed. – Upper Saddle River, NJ : Pearson Education, 2011. – 720 p.
5. Bertsimas D., Tsitsiklis J. Introduction to Linear Optimization. – Belmont : Athena Scientific, 1997. – 608 p.
6. ISO/IEC 25010:2011. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. – Geneva : ISO, 2011.

Варіанти

Варіант 1. Оптимальне планування виробництва напоїв

Проблема:

Підприємство виготовляє два види напоїв: Лимонад (x) і Апельсиновий сік (y). Потрібно мінімізувати витрати на виробництво, враховуючи обмеження на доступність інгредієнтів: воду, цукор та апельсиновий концентрат.

Змінні рішення:

8. x — літри лимонаду
9. y — літри апельсинового соку

Цільова функція:

$\min\ 0.5x + 0.7y$

(вартість виготовлення одиниці напоїв)

Обмеження:

7. Вода: $2x + 3y \leq$ 500 літрів
8. Цукор: $1x + 2y \leq$ 200 кг
9. Концентрат: $x + 1y \leq$ 150 літрів
10. Мінімальний випуск продукції: $x + y \geq 100$

Варіант 2. Оптимізація асортименту магазинів

Проблема:

Магазин вирішує, скільки продавати Товару А (x) і Товару В (y) для мінімізації витрат на закупку при обмеженні складських площ і бюджету.

Змінні рішення:

- x — кількість одиниць товару А
- y — кількість одиниць товару В

Цільова функція:

$\min\ 20x + 15y$

Обмеження:

6. Площа складу: $2x + 3y \leq 300$ м²
7. Бюджет: $20x + 15y \leq 4000$
8. Мінімальні продажі: $x + y \geq 150$

Варіант 3. Мінімізація витрат на освітлення парку

Проблема:

Необхідно розташувати два типи ліхтарів (тип А і тип В) для освітлення парку з мінімальними витратами.

Змінні рішення:

- x — кількість ліхтарів типу А
- y — кількість ліхтарів типу В

Цільова функція:

$\min\ 300x + 500y$

Обмеження:

5. Мінімальне освітлення зони 1: $x + 2y \geq 30$
6. Мінімальне освітлення зони 2: $2x + 5y \geq 40$
7. Межі кількості: $x, y \geq 0$

Варіант 4. Розподіл ресурсів на рекламні кампанії

Проблема:

Компанія хоче мінімізувати витрати на рекламу в Інтернеті та на телебаченні, досягаючи мінімального охоплення аудиторії.

Змінні рішення:

- x — тисячі гривень на рекламу в Інтернеті
- y — тисячі гривень на рекламу на ТБ

Цільова функція:

$\min\ 5x + 8y$

Обмеження:

- Охоплення: $10x + 15y \geq 150$ тисяч осіб
- Бюджет: $5x + 8y \leq 200$
- Мінімальна присутність в Інтернеті: $x \geq 10$

Варіант 5. Оптимізація виробництва меблів

Проблема:

Фабрика виготовляє столи (x) і стільці (y). Потрібно мінімізувати витрати на виробництво при обмеженні за кількістю робочих годин та матеріалів.

Змінні рішення:

- x — кількість столів

- y — кількість стільців

Цільова функція:

$\min\ 50x + 20y$

Обмеження:

- Робочі години: $3x + 2y \leq 240$
- Матеріали: $4x + 1y \leq 300$
- Мінімальний випуск: $x + y \geq 50$

Варіант 6. Мінімізація витрат на транспорт товарів

Проблема:

Компанія має доставити товари із складу у два магазини з мінімальними витратами на транспортування.

Змінні рішення:

- x — кількість одиниць товару до магазину 1
- y — кількість одиниць товару до магазину 2

Цільова функція:

$\min\ 10x + 15y$

Обмеження:

- Потреба магазину 1: $x \geq 80$
- Потреба магазину 2: $y \geq 60$
- Обмеження складу: $x + y \leq 200$

Варіант 7. Оптимізація використання сировини для хімічного виробництва

Проблема:

Компанія виробляє два продукти X і Y з однієї сировини, потрібно мінімізувати витрати при обмеженні на використання сировини і забезпеченні мінімального виробництва.

Змінні рішення:

- x — кількість продукту X
- y — кількість продукту Y

Цільова функція:

$\min\ 12x + 18y$

Обмеження:

- Використання сировини: $5x + 7y \leq 500$
- Мінімальне виробництво: $x + y \geq 50$
- Мінімальна кількість продукту Y: $y \geq 20$

1

Лабораторна робота №6. Аналіз та візуалізація великих даних у системах підтримки прийняття рішень

Мета роботи: Ознайомитися з інструментами та підходами до роботи з Big Data, навчитися використовувати сучасні аналітичні платформи для прийняття обґрунтованих управлінських рішень на основі даних.

Короткі теоретичні положення

Великі дані (Big Data) — це обсяги даних, які перевищують можливості традиційних інструментів обробки інформації щодо їх збору, зберігання, обробки та аналізу. Вони характеризуються п’ятьма основними властивостями (5V):

- **Volume** (обсяг) — величезні обсяги даних (від терабайтів до петабайтів);
- **Velocity** (швидкість) — висока швидкість надходження та обробки;
- **Variety** (різноманіття) — структуровані, напівструктуровані та неструктуровані дані;
- **Veracity** (достовірність) — мінливість і неоднорідність якості даних;
- **Value** (цінність) — здатність отримувати з даних користь.

У контексті **систем підтримки прийняття рішень (СППР)**, великі дані дозволяють:

- формувати глибший аналіз поведінки користувачів, ринкових тенденцій, ризиків;
- автоматизувати процеси ухвалення рішень;
- підвищувати точність прогнозів за рахунок використання широкого набору даних;
- виявляти приховані закономірності в потоках інформації.

Архітектура обробки великих даних

Основні компоненти інфраструктури роботи з Big Data:

- **Сховище даних**: Google BigQuery, Hadoop Distributed File System (HDFS), Amazon S3.
- **Інструменти обробки**: Apache Spark, Dask, SQL-like інтерфейси (BigQuery SQL).
- **Мови програмування**: Python, Scala, Java.
- **Середовища візуалізації**: Google Data Studio, Tableau, Python-бібліотеки (Plotly, Seaborn, Matplotlib).

Google BigQuery — це аналітична платформа для обробки великих обсягів даних у хмарному середовищі. Вона дозволяє виконувати SQL-запити до таблиць, які можуть містити мільйони або мільярди рядків, за лічені секунди.

Переваги BigQuery у СППР:

- Підтримка SQL-подібної мови для аналізу.
- Висока масштабованість.
- Підтримка інтеграції з Python, R, Tableau.
- Збереження даних у структурованій формі з можливістю шифрування та контролю доступу.

Взаємодія BigQuery та Python

Python через бібліотеку google-cloud-bigquery надає можливості:

1. автоматизованого запуску SQL-запитів;
2. зчитування результатів запитів у формі DataFrame;
3. подальшого аналізу та візуалізації даних за допомогою Pandas, Seaborn, Plotly.

Це дозволяє:

1. будувати аналітичні пайплайни;
2. інтегрувати дані з іншими джерелами;
3. оперативно формувати графіки, таблиці, теплові карти тощо.

Візуалізація результатів

У СППР візуалізація виконує ключову роль, адже вона:

1. сприяє швидкому розумінню даних і трендів;
2. допомагає формувати звіти;

3. полегшує виявлення критичних точок і аномалій.

Найпоширеніші типи візуалізацій:

1. стовпчикові та лінійні діаграми (trend analysis);
2. теплові карти (heatmaps);
3. кругові діаграми (розподіл категорій);
4. boxplot та histogram (розподіли значень);
5. кореляційні матриці (виявлення взаємозв'язків між змінними).

Завдання до роботи

1. Ознайомитися з можливостями платформи **Google BigQuery** для аналізу великих даних.
2. Підключити Python до BigQuery через бібліотеку google-cloud-bigquery.
3. Вибрати одну з відкритих публічних таблиць (наприклад, bigquery-public-data) для аналізу.
4. Сформувати SQL-запит до обраної таблиці з метою:
5. фільтрації, сортування або агрегації даних;
6. вибірки за умовами (наприклад, по роках, регіонах, категоріях).
7. Отримати результат запиту у форматі Pandas DataFrame.
8. Провести базовий статистичний аналіз даних (середнє, медіана, стандартне відхилення, кількість пропущених значень).
9. Побудувати не менше 3 різних типів візуалізацій (лінійна діаграма, гістограма, кореляційна матриця тощо).
10. Зробити аналітичний висновок на основі побудованих графіків щодо можливостей використання цих даних у СППР.

Опис способу обробки результатів, хід експерименту

Обробка результатів лабораторної роботи здійснюється шляхом виконання аналітичного запиту до великого набору даних, подальшої обробки отриманих результатів у середовищі Python та їх візуального аналізу.

У ході експерименту:

1. Результати SQL-запиту зберігаються у вигляді таблиці (DataFrame) для подальшого аналізу.
2. Проводиться базовий статистичний аналіз та очищення даних (обробка пропусків, перетворення типів).
3. Будуються візуалізації для виявлення трендів, розподілів та взаємозв'язків між змінними.
4. На основі отриманих графіків формулюються аналітичні висновки щодо можливостей використання даних у системах підтримки прийняття рішень.

Приклад виконання

Крок 1. Реєстрація в Google Cloud Platform (GCP) та створення проєкту
Перейдіть на сайт: https://console.cloud.google.com/
Якщо ви вперше — підтвердьте обліковий запис та прийміть умови.
Натисніть **"Select project" → "New project"**.
Введіть назву (наприклад, spp-bq-analysis) → натисніть **Create**.
Переконайтесь, що обрали правильний проєкт (вказаний у верхньому меню).

Крок 2. Увімкнення API для BigQuery
В меню ліворуч: **APIs & Services → Library**.
У пошуку введіть **BigQuery API**.
Натисніть **Enable**.

Крок 3. Створення сервісного облікового запису та ключа доступу
В меню: IAM & Admin → Service Accounts.
Натисніть + Create Service Account.
Введіть назву (наприклад, colab-access) → натисніть Create and Continue.
Додайте ролі:
BigQuery > BigQuery Data Viewer

BigQuery > BigQuery Job User
BigQuery > BigQuery Read Session User
Завершіть створення → перейдіть на вкладку Keys.
Натисніть Add Key → Create new key → JSON.
Збережіть файл ключа (.json) на комп'ютер.

Крок 4. Завантаження ключа у Google Colab
Відкрийте новий файл у Google Colab.
Завантажте .json-файл ключа:

```
from google.colab import files
uploaded = files.upload()
```

Нотатка: після завантаження буде виведено ім'я файлу. Надалі використовуйте його в key_path.

Крок 5. Підключення до Google BigQuery у Python

```
from google.cloud import bigquery
from google.oauth2 import service_account

key_path = "your-key.json"  # Замінити на фактичну назву файлу

credentials = service_account.Credentials.from_service_account_file(key_path)
client = bigquery.Client(credentials=credentials, project=credentials.project_id)
```

Крок 6. Отримання даних із BigQuery
Обираємо публічний датасет, наприклад: bigquery-public-data.covid19_open_data.
SQL-запит для отримання COVID-даних по Україні за 2021 рік:

```
query = """
SELECT
  country_name,
  date,
  cumulative_confirmed AS total_confirmed,
  cumulative_deceased AS total_deceased,
  population
FROM `bigquery-public-data.covid19_open_data.covid19_open_data`
WHERE date BETWEEN '2021-01-01' AND '2021-12-31'
  AND country_name = 'Ukraine'
ORDER BY date
"""

df = client.query(query).to_dataframe()
```

Крок 7. Попередній аналіз та обробка даних

```
import pandas as pd

df.info()
df.describe()
df.isna().sum()
df['date'] = pd.to_datetime(df['date'])
df.fillna(0, inplace=True)
df['mortality_rate'] = df.apply(
    lambda row: row['total_deceased'] / row['total_confirmed']
    if row['total_confirmed'] > 0 else 0, axis=1)
```

Крок 8. Візуалізація результатів
1. Лінійна діаграма захворювань і смертей:

```
import matplotlib.pyplot as plt

plt.figure(figsize=(12,6))
plt.plot(df['date'], df['total_confirmed'], label="Захворювання")
plt.plot(df['date'], df['total_deceased'], label="Смерті")
```

```
plt.title("Динаміка COVID-19 в Україні (2021)")
plt.xlabel("Дата")
plt.ylabel("Кількість")
plt.legend()
plt.grid(True)
plt.show()
```
2. Гістограма летальності:
```
plt.hist(df['mortality_rate'], bins=30, color='orange')
plt.title("Розподіл летальності")
plt.xlabel("Летальність")
plt.ylabel("Кількість днів")
plt.show()
```
3. Теплова карта кореляцій:
```
import seaborn as sns

corr = df[['total_confirmed', 'total_deceased', 'population', 'mortality_rate']].corr()
sns.heatmap(corr, annot=True, cmap="coolwarm")
plt.title("Кореляційна матриця")
plt.show()
```
Крок 9. Інтерпретація результатів

У звіті вказати:

- які змінні найбільш взаємопов’язані;
- які періоди мали піки заражень або летальності;
- які висновки можна зробити для прийняття рішень.

Домашнє завдання

1. Обрати інший публічний набір великих даних (наприклад, транспортні, фінансові, кліматичні або демографічні дані).
2. Сформувати та виконати власний SQL-запит, який містить:
    - фільтрацію за часовим періодом або категорією;
    - агрегування (SUM, AVG, COUNT тощо).
3. Завантажити результати запиту у Python та виконати:
    - перевірку структури даних;
    - базовий статистичний аналіз.
4. Побудувати не менше трьох різних типів візуалізацій.
5. Сформулювати короткий аналітичний висновок (5–7 речень) щодо того, які управлінські рішення можуть бути підтримані на основі виконаного аналізу.

Рекомендації щодо оформлення звіту

Звіт з лабораторної роботи оформлюється на аркушах формату А4 та повинен містити такі розділи:

1. Титульний аркуш (за вимогами кафедри).
2. Тема та мета роботи.
3. Короткі теоретичні положення:
    - поняття великих даних;
    - роль аналітики та візуалізації у СППР.
4. Опис даних:
    - джерело набору даних;
    - коротка характеристика змінних;
    - обсяг вибірки.
5. Хід виконання роботи:
    - опис SQL-запиту;
    - етапи обробки даних у Python.
6. Результати та візуалізація:
    - побудовані графіки;
    - таблиці з агрегованими показниками.
7. Аналіз результатів та висновки:
    - інтерпретація виявлених трендів;
    - можливості використання результатів у СППР.

8. Список використаної літератури.

Вимоги до оформлення:

- шрифт Times New Roman, 14 pt;
- міжрядковий інтервал — 1.5;
- усі рисунки та таблиці повинні мати назву та номер.

Контрольні питання

1. Що таке великі дані? Які основні характеристики Big Data?
2. Яке призначення Google BigQuery у СППР?
3. Як виконується підключення до BigQuery з Python?
4. У чому різниця між SQL-запитами в BigQuery та традиційними SQL?
5. Які типи візуалізацій найбільш інформативні для аналізу великих даних?
6. Як визначити кореляцію між змінними в pandas/Seaborn?
7. Які труднощі можуть виникнути при роботі з великими даними в хмарі?

Література

1. Han J., Kamber M., Pei J. Data Mining: Concepts and Techniques. – 3rd ed. – Waltham : Morgan Kaufmann, 2012. – 703 p.
2. Provost F., Fawcett T. Data Science for Business. – Sebastopol : O'Reilly Media, 2013. – 414 p.
3. Aggarwal C. C. Big Data Analytics. – Cham : Springer, 2014. – 340 p.
4. Marr B. Big Data in Practice. – Chichester : Wiley, 2016. – 256 p.
5. Few S. Information Dashboard Design. – 2nd ed. – Burlingame : Analytics Press, 2013. – 223 p.
6. Turban E., Sharda R., Delen D. Decision Support and Business Intelligence Systems. – 9th ed. – Upper Saddle River, NJ : Pearson Education, 2011. – 720 p.

Варіанти

Варіант 1

Тема: Аналіз найпопулярніших імен у США

Джерело: bigquery-public-data.usa_names.usa_1910_2013

Завдання:

- Здійснити вибірку топ-10 імен у вказаному штаті.
- Побудувати графік популярності імені по роках.

Приклад коду:

```
query = """
SELECT name, year, SUM(number) AS total
FROM `bigquery-public-data.usa_names.usa_1910_2013`
WHERE state = 'CA'
GROUP BY name, year
ORDER BY year
"""

df = client.query(query).to_dataframe()
```

Варіант 2

Тема: Динаміка споживчих цін (CPI) у світі

Джерело: bigquery-public-data.imf_world_economic_outlook.country_series_latest

Завдання:

- Витягти дані CPI для вибраних країн.
- Побудувати графік динаміки та знайти кореляцію з ВВП.

Приклад коду:

```
query = """
SELECT country_name, year, cpi, gdp
FROM `bigquery-public-data.imf_world_economic_outlook.country_series_latest`
WHERE year BETWEEN 2010 AND 2021
AND country_name IN ('Germany', 'France', 'Italy')
"""
```

```
df = client.query(query).to_dataframe()
```

Варіант 3
Тема: COVID-19 у різних країнах
Джерело: bigquery-public-data.covid19_open_data.covid19_open_data
Завдання:

- Порівняти динаміку захворюваності в 3 країнах.
- Побудувати гістограму летальності.

Приклад коду:

```
query = """
SELECT date, country_name, total_confirmed, total_deceased
FROM `bigquery-public-data.covid19_open_data.covid19_open_data`
WHERE country_name IN ('Ukraine', 'Poland', 'Germany')
AND date BETWEEN '2021-01-01' AND '2021-12-31'
"""

df = client.query(query).to_dataframe()
```

Варіант 4
Тема: Статистика народжень у США
Джерело: bigquery-public-data.natality.us_births
Завдання:

- Побудувати річний та місячний розподіл народжень.

Приклад коду:

```
query = """
SELECT year, month, state, mother_age, COUNT(*) AS births
FROM `bigquery-public-data.natality.us_births`
WHERE state = 'California'
GROUP BY year, month, state, mother_age
"""

df = client.query(query).to_dataframe()
```

Варіант 5
Тема: Енергоспоживання в США
Джерело: bigquery-public-data.eia_co2.us_electricity_generation
Завдання:

- Побудувати діаграму часток генерації енергії з різних джерел.

Приклад коду:

```
query = """
SELECT year, energy_source, SUM(generation_mwh) AS total_gen
FROM `bigquery-public-data.eia_co2.us_electricity_generation`
WHERE year >= 2015
GROUP BY year, energy_source
ORDER BY year
"""

df = client.query(query).to_dataframe()
```

Варіант 6
Тема: Якість повітря у США
Джерело: bigquery-public-data.epa_historical_air_quality.pm25_frm_daily_summary
Завдання:

- Візуалізувати рівень PM2.5 для заданого міста.

Приклад коду:

```
query = """
SELECT date_local, county_name, arithmetic_mean AS pm25
```

```
FROM `bigquery-public-data.epa_historical_air_quality.pm25_frm_daily_summary`
WHERE state_code = '06' AND county_name = 'Los Angeles'
AND date_local BETWEEN '2021-01-01' AND '2021-12-31'
"""

df = client.query(query).to_dataframe()
```

Варіант 7
Тема: Міграція населення за даними ООН
Джерело: bigquery-public-data.un_migration.migration_data
Завдання:

- Побудувати зміну міграції між країнами у часі.

Приклад коду:

```
query = """
SELECT year, country_name, destination_name, value
FROM `bigquery-public-data.un_migration.migration_data`
WHERE country_name IN ('Ukraine', 'Poland', 'Germany')
AND destination_name IN ('Germany', 'Poland')
"""

df = client.query(query).to_dataframe()
```

Лабораторна робота №7. Аналіз ризиків і вразливостей у системах підтримки прийняття рішень

Мета роботи: Набути практичних навичок виявлення, аналізу та пріоритетизації інформаційних ризиків у системах підтримки прийняття рішень.

Короткі теоретичні положення

У сучасних СППР інформаційні ризики є ключовим фактором, що впливає на надійність, безпечність і стабільність прийняття рішень.

Основні поняття:

1 Ризик — функція ймовірності виникнення загрози (P) та потенційної шкоди (S):

$$R = P \times S$$

2 Матриця ризиків — двовимірна таблиця, яка класифікує загрози за шкалою "Ймовірність × Шкода".

3 Категорії ризику: допустимий, помірний, критичний, катастрофічний (визначаються порогами).

4 Взаємозалежність загроз: деякі загрози підвищують або знижують ризик інших. Модель представлена орієнтованим графом.

5 Метод TOPSIS — один із методів багатокритеріального вибору, який дозволяє оцінити загрози не лише за ризиком, а й за іншими критеріями (вартість ліквідації, іміджевий вплив, складність виявлення).

Завдання до роботи

- Завантажити таблицю загроз (ID, опис, ймовірність, шкода, вартість усунення, іміджеві наслідки, залежності).
- Розрахувати індивідуальні значення ризику для кожної загрози.
- Побудувати матрицю ризиків та класифікувати загрози за зонами небезпеки.
- Побудувати граф залежностей між загрозами (networkx), ідентифікувати критичні вузли.
- Виконати сценарний аналіз: змінити ймовірності/шкоду в залежності від умов (песимістичний, базовий, оптимістичний сценарії).
- Провести багатокритеріальний аналіз пріоритетності загроз за методом TOPSIS.
- Побудувати графік ризиків (heatmap) та граф впливів.
- Зробити висновки щодо найпріоритетніших загроз.

Опис способу обробки результатів, хід експерименту

Обробка результатів лабораторної роботи здійснюється шляхом кількісної оцінки інформаційних ризиків та їх подальшої аналітичної інтерпретації.

У ході експерименту:

1. Для кожної загрози обчислюється базовий рівень ризику на основі ймовірності та очікуваної шкоди.
2. Загрози класифікуються за рівнями ризику та відображаються у матриці ризиків.
3. Аналізується взаємозалежність загроз шляхом побудови орієнтованого графа впливів.
4. Проводиться сценарний аналіз (оптимістичний, базовий, песимістичний) для оцінки стійкості результатів.
5. Виконується багатокритеріальна пріоритетизація загроз методом TOPSIS з урахуванням ризику, вартості нейтралізації та іміджевих наслідків.
6. На основі візуалізацій і розрахунків формулюються висновки щодо найкритичніших загроз для СППР.

Приклад виконання

Крок 1: Завантаження початкових даних загроз
Дані містять список загроз для СППР з атрибутами:

- ID — унікальний ідентифікатор загрози.
- Description — опис загрози.
- Probability — ймовірність реалізації загрози (від 0 до 1).
- Impact — очікувана шкода (від 0 до 1).
- MitigationCost — вартість запобігання (умовні одиниці).
- ReputationImpact — оцінка іміджевої шкоди (1–10).

ID	Description	Probability	Impact	MitigationCost	ReputationImpact
T1	Несанкціонований доступ	0.8	0.9	8	9
T2	Витік даних через API	0.6	0.7	5	7
T3	Помилки адміністрування	0.4	0.3	3	2
T4	DDoS-атака	0.7	0.8	7	8
T5	Внутрішня загроза (інсайдер)	0.5	0.6	6	6

```
import pandas as pd

data = {
   "ID": ["T1", "T2", "T3", "T4", "T5"],
   "Description": [
      "Несанкціонований доступ",
      "Витік даних через API",
      "Помилки адміністрування",
      "DDoS-атака",
      "Внутрішня загроза (інсайдер)"
   ],
   "Probability": [0.8, 0.6, 0.4, 0.7, 0.5],
   "Impact": [0.9, 0.7, 0.3, 0.8, 0.6],
   "MitigationCost": [8, 5, 3, 7, 6],
   "ReputationImpact": [9, 7, 2, 8, 6]
}

df_threats = pd.DataFrame(data)
print(df_threats)
```

Крок 2: Обчислення базового ризику
Для кожної загрози обчислюється ризик як:

$$\text{Risk} = \text{Probability} \times \text{Impact}$$

У Python:

```
df_threats['Risk'] = df_threats['Probability'] * df_threats['Impact']
```

Крок 3: Класифікація ризику
Загрози розподіляються за рівнями:

- 0–0.2: Низький ризик
- 0.2–0.5: Помірний ризик
- 0.5–0.7: Високий ризик
- >0.7: Критичний ризик

```
def classify_risk(r):
   if r <= 0.2: return 'Низький'
   elif r <= 0.5: return 'Помірний'
   elif r <= 0.7: return 'Високий'
   else: return 'Критичний'
df_threats['RiskLevel'] = df_threats['Risk'].apply(classify_risk)
```

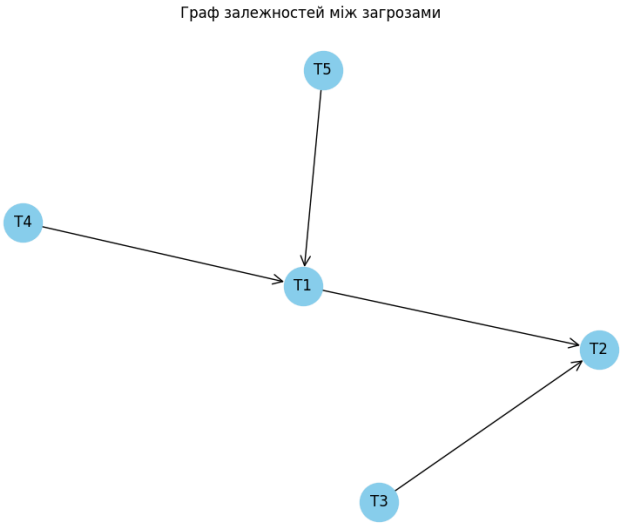
Крок 4: Візуалізація ризиків

```
import seaborn as sns
import matplotlib.pyplot as plt

pivot = df_threats.pivot(index='ID', columns='Description', values='Risk')
sns.heatmap(pivot, annot=True, cmap='Reds')
plt.title('Матриця ризиків')
plt.show()

plt.figure(figsize=(8, 5))
plt.bar(df_threats["ID"], df_threats["Risk"], color='tomato')
plt.title("Рівень ризику по загрозах")
plt.xlabel("Загроза")
plt.ylabel("Ризик")
plt.grid(True)
plt.show()
```

Крок 5: Побудова графа залежностей між загрозами

Для врахування впливів одних загроз на інші створюємо орієнтований граф.

```
import networkx as nx
G = nx.DiGraph()
G.add_nodes_from(df_threats["ID"])
edges = [("T3", "T2"), ("T1", "T2"), ("T5", "T1"), ("T4", "T1")]
G.add_edges_from(edges)
pos = nx.spring_layout(G, seed=42)
plt.figure(figsize=(8, 6))
nx.draw(G, pos, with_labels=True, node_color='skyblue', node_size=1000,
        arrows=True, arrowstyle='->', arrowsize=20)
plt.title("Граф залежностей між загрозами")
plt.show()
```



На діаграмі видно:

3. вузли — загрози,
4. стрілки — напрямки впливу

Крок 6: Сценарний аналіз

Перевірити, як змінюється рівень ризику для кожної загрози за трьома сценаріями:

Сценарій	Опис
Оптимістичний	Ймовірність і шкода зменшуються на 20%
Базовий	Поточні дані (без змін)
Песимістичний	Ймовірність і шкода збільшуються на 20%

Формула розрахунку ризику:

$$R_i = P_i \times S_i$$

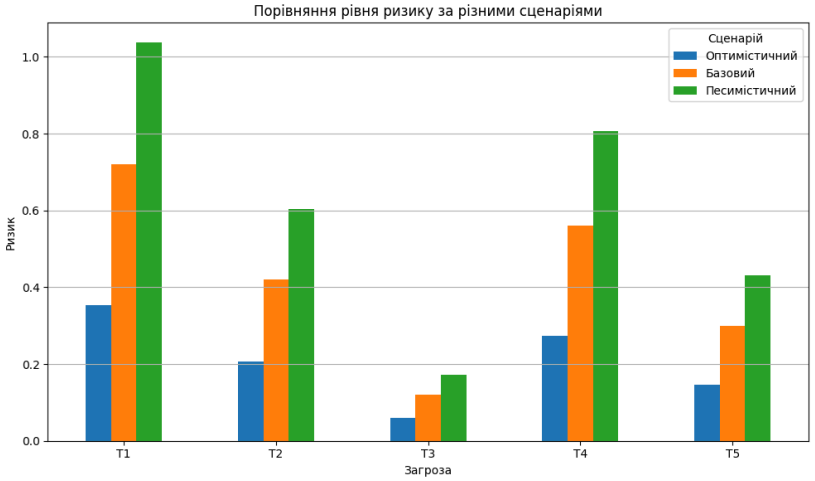
де:

5. P_i — ймовірність загрози в обраному сценарії,
6. S_i — шкода від її реалізації в обраному сценарії.

Розрахунок (приклад для T1 – Несанкціонований доступ):

6. **P = 0.8, S = 0.9**
7. **Базовий:**
   $R = 0.8 \times 0.9 = 0.72$
8. **Оптимістичний:**
   $P = 0.8 \times 0.8 = 0.64,\ S = 0.9 \times 0.8 = 0.72$
   $R = 0.64 \times 0.72 = 0.461$
9. **Песимістичний:**
   $P = 0.8 \times 1.2 = 0.96,\ S = 0.9 \times 1.2 = 1.08$
   $R = 0.96 \times 1.08 = 1.037$

Результати сценарного аналізу



ID	Оптимістичний	Базовий	Песимістичний	Рівень (Опт.)	Рівень (Базовий)	Рівень (Песим.)
T1	0.461	0.720	1.037	Помірний	Критичний	Критичний
T2	0.269	0.420	0.605	Помірний	Помірний	Високий
T3	0.077	0.120	0.173	Низький	Низький	Низький
T4	0.358	0.560	0.806	Помірний	Високий	Критичний
T5	0.192	0.300	0.432	Низький	Помірний	Помірний

Висновки зі сценарного аналізу:

4. T1 залишається критичною загрозою навіть у базовому та песимістичному сценарії.
5. T4 (DDoS-атака) стає критичною в песимістичному сценарії — потенційно недооцінена.
6. T3 стабільно має низький ризик — може мати низький пріоритет.
7. Сценарій дозволяє виявити найбільш нестійкі до змін загрози.

Код на Python:

```
scenarios = {
   "Оптимістичний": (0.7, 0.7),
   "Базовий": (1.0, 1.0),
   "Песимістичний": (1.2, 1.2)
}
for name, (p_mult, i_mult) in scenarios.items():
   df_threats[name] = (df_threats["Probability"] * p_mult) * (df_threats["Impact"] * i_mult)
```

```
    df_threats[f"{name}_Level"] = df_threats[name].apply(classify_risk)
scenarios_plot = df_threats[["ID", "Оптимістичний", "Базовий", "Песимістичний"]].set_index("ID")
scenarios_plot.plot(kind='bar', figsize=(10, 6))
plt.title("Порівняння рівня ризику за різними сценаріями")
plt.ylabel("Ризик")
plt.xlabel("Загроза")
plt.xticks(rotation=0)
plt.grid(axis='y')
plt.legend(title="Сценарій")
plt.tight_layout()
plt.show()
```

Крок 7: Пріоритетизація загроз методом TOPSIS

Провести **багатокритеріальну оцінку загроз** у системі підтримки прийняття рішень з урахуванням не лише ризику, але й додаткових факторів: вартості нейтралізації та іміджевих наслідків.

Вхідні дані

Для кожної загрози враховуються такі критерії:

10. Risk – розрахований ризик: $R_i = P_i \times S_i$
11. MitigationCost – оцінка вартості ліквідації загрози (умовні одиниці)
12. ReputationImpact – оцінка негативного іміджевого впливу (1–10)

Критерії необхідно мінімізувати, тобто що менше значення — то краще.

Ваги критеріїв (за експертним методом):

6. Ризик — 0.5
7. Вартість нейтралізації — 0.3
8. Іміджеві наслідки — 0.2

Реалізація в Python використовуючи бібліотеку pyDecision для алгоритму TOPSIS:

```
!pip install pyDecision

from pyDecision.algorithm import topsis_method
import numpy as np

# Матриця рішень (альтернативи x критерії)
X = np.array([
    [0.72, 8, 9],  # T1
    [0.42, 5, 7],  # T2
    [0.12, 3, 2],  # T3
    [0.56, 7, 8],  # T4
    [0.30, 6, 6]   # T5
])

criteria_type = ['max', 'min', 'max']
weights = [0.5, 0.3, 0.2]
scores = topsis_method(X, weights, criteria_type, graph=False, verbose=False)

threats = ["T1", "T2", "T3", "T4", "T5"]
for i, score in enumerate(scores):
    print(f"{threats[i]}: {score:.2f}")

plt.figure(figsize=(8,5))
plt.bar(threats, scores, color="skyblue")
plt.xlabel("Загрози")
plt.ylabel("TOPSIS оцінка")
plt.title("Пріоритет загроз за методом TOPSIS")
plt.grid(True, axis='y')
plt.show()
```

Приклад результату:

T1: 0.73
T2: 0.53

T3: 0.27
T4: 0.66
T5: 0.34

Інтерпретація:

3. Найвищий бал отримала загроза T1, отже вона є найкритичнішою за сукупністю факторів.
4. T3 — найменш небезпечна й пріоритетна загроза.
5. Такий підхід дозволяє балансувати між ризиком, вартістю та репутаційними збитками, а не зосереджуватись лише на одному критерії.

Домашнє завдання

1. Сформувати власний перелік загроз для обраної системи підтримки прийняття рішень (не менше 6 загроз).
2. Задати для кожної загрози:
    - ймовірність реалізації;
    - величину шкоди;
    - вартість усунення;
    - іміджеві наслідки.
3. Розрахувати базовий рівень ризику та виконати його класифікацію.
4. Побудувати матрицю ризиків та граф залежностей між загрозами.
5. Провести сценарний аналіз (мінімум два альтернативні сценарії).
6. Виконати пріоритетизацію загроз методом TOPSIS з обґрунтуванням вибору ваг критеріїв.
7. Сформулювати короткі рекомендації щодо першочергових заходів зниження ризиків.

Рекомендації щодо оформлення звіту

Звіт з лабораторної роботи оформлюється на аркушах формату A4 та повинен містити:

1. Титульний аркуш (за вимогами кафедри).
2. Тему та мету лабораторної роботи.
3. Короткі теоретичні положення:
    - поняття ризику та вразливості;
    - методи аналізу ризиків у СППР.
4. Вхідні дані:
    - опис загроз;
    - обґрунтування вибраних параметрів.
5. Хід виконання роботи:
    - розрахунок ризиків;
    - побудова матриці ризиків;
    - сценарний аналіз;
    - TOPSIS-аналіз.
6. Результати та візуалізація:
    - heatmap ризиків;
    - граф залежностей;
    - діаграма пріоритетів.
7. Аналіз результатів і висновки:
    - визначення критичних загроз;
    - рекомендації щодо управління ризиками.
8. Список використаної літератури.

Вимоги до оформлення:

- шрифт Times New Roman, 14 pt;
- міжрядковий інтервал — 1.5;
- усі таблиці та рисунки повинні мати назви та нумерацію.

Контрольні питання

1. Що таке ризик у контексті СППР і як він розраховується?
2. Яку мету має побудова матриці ризиків?
3. У чому полягає ідея сценарного аналізу ризиків?
4. Як враховується взаємозалежність загроз?
5. Який принцип роботи методу TOPSIS?
6. Які додаткові критерії доцільно враховувати при пріоритетизації загроз?
7. У чому переваги використання Python для аналізу ризиків?

Література

1. ISO/IEC 27005:2018 Information security risk management. – Geneva : ISO, 2018.
2. NIST SP 800-30 Guide for Conducting Risk Assessments. – Gaithersburg : National Institute of Standards and Technology, 2012.
3. Stoneburner G., Goguen A., Feringa A. Risk Management Guide for Information Technology Systems. – Gaithersburg : NIST, 2002. – 85 p.
4. Alberts C., Dorofee A. Managing Information Security Risks. – Boston : Addison-Wesley, 2002. – 432 p.
5. Turban E., Sharda R., Delen D. Decision Support and Business Intelligence Systems. – 9th ed. – Upper Saddle River, NJ : Pearson Education, 2011. – 720 p.
6. Saaty T. L. Decision Making for Leaders. – Pittsburgh : RWS Publications, 1990. – 315 p.

Варіанти

Варіант 1

ID	Загроза	P	S	Вартість	Імідж
T1	Злам облікових записів	0.6	0.8	7	9
T2	DDoS-атака на вебпортал	0.5	0.7	6	8
T3	Витік через співробітника	0.4	0.6	4	7
T4	Помилки в оновленнях ПЗ	0.3	0.5	3	6
T5	Незахищене API	0.5	0.8	5	8

Варіант 2

ID	Загроза	P	S	Вартість	Імідж
T1	Незашифроване зберігання паролів	0.7	0.6	6	8
T2	Вразливість SQL Injection	0.5	0.9	7	9
T3	Зловмисне ПЗ	0.4	0.4	4	5
T4	Людський фактор (помилки користувачів)	0.6	0.3	3	4
T5	Віддалене адміністрування без VPN	0.5	0.7	5	6

Варіант 3

ID	Загроза	P	S	Вартість	Імідж
T1	Несанкціоноване копіювання даних	0.6	0.9	7	9
T2	Недостатній контроль доступу	0.5	0.7	5	6
T3	Спам та фішинг	0.7	0.6	6	7
T4	Порушення резервного копіювання	0.3	0.5	4	5
T5	Підробка документів	0.4	0.8	6	8

Варіант 4

ID	Загроза	P	S	Вартість	Імідж
T1	Незахищені канали передачі	0.5	0.6	5	7
T2	Фізичний доступ до серверів	0.3	0.9	8	9
T3	Несанкціонований доступ ззовні	0.6	0.7	6	8
T4	Неактуальні антивіруси	0.4	0.6	3	5
T5	Вразливість у CMS	0.5	0.8	5	7

Варіант 5

ID	Загроза	P	S	Вартість	Імідж
T1	Використання застарілого ПЗ	0.5	0.6	4	6
T2	Несанкціонована модифікація	0.6	0.7	5	7
T3	Відсутність журналювання	0.4	0.5	3	5
T4	Порушення політики паролів	0.7	0.8	6	9

ID	Загроза	P	S	Вартість	Імідж
T5	Витік даних через флешки	0.3	0.6	4	6

Варіант 6

ID	Загроза	P	S	Вартість	Імідж
T1	Атака на мережу Wi-Fi	0.6	0.5	4	7
T2	Ненадійна система автентифікації	0.7	0.6	5	8
T3	Вразливість у мобільному додатку	0.5	0.8	6	9
T4	Атака типу Man-in-the-Middle	0.4	0.7	5	7
T5	Компрометація системних журналів	0.3	0.6	4	6

Варіант 7

ID	Загроза	P	S	Вартість	Імідж
T1	Небезпечне з'єднання з базами даних	0.6	0.7	5	8
T2	Витік через файлообмінники	0.4	0.6	3	6
T3	Відсутність логів авторизації	0.5	0.5	4	5
T4	Проблеми з SSL-сертифікатами	0.3	0.7	5	7
T5	Порушення цілісності даних	0.7	0.8	6	9