Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА

## Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Вилучення та аналіз даних в судочинстві
за допомогою OLAP та Data Mining

(тема)

Виконав:
студент II курсу, групи СПм-20-1
Полонець К.С.
(прізвище, ініціали)

Спеціальність
123 – Комп’ютерна інженерія
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма
Системне програмування
(повна назва освітньої програми)

Керівник: доц. Ільїна І.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри ЕОМ _____ Коваленко А.А.
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 – Комп’ютерна інженерія
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20___ р.

# ЗАВДАННЯ

## НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Полонцю Кирилу Сергійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Вилучення та аналіз даних в судочинстві за допомогою OLAP та Data Mining

затверджена наказом по університету від “ 05 ” листопада 2021 р. № 1657 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 14 грудня 2021 р.

3. Вхідні дані до роботи

1) організаційна структура підприємства та структурного підрозділу;
2) файли текстів судочинства;
3) методи засобів обробки даних.

4. Перелік питань, що потрібно опрацювати в роботі

1) аналіз вихідної інформації та вимог ТЗ;
2) аналіз предметної галузі судочинства;
3) проектування інструментів для обробки даних;
4) розробка інструментів для обробки даних;
5) аналіз даних за допомогою технології Data Mining;
6) висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів)

Слайд-презентація – 12 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1 )

| Найменування розділу | Консультант (посада, прізвище, ім'я, по батькові) | Позначка консультанта про виконання розділу | |
|---|---|---|---|
| | | підпис | дата |
| | | | |
| | | | |

## КАЛЕНДАРНИЙ ПЛАН

| № | Назва етапів роботи | Термін виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | Аналіз проблеми та огляд існуючих рішень | 09.11.21-12.11.21 | |
| 2 | Вибір технології розробки та | 13.11.21-18.11.21 | |
| | інструментальних засобів | | |
| 3 | Розробка проектних та технічних рішень | 19.11.21-03.12.21 | |
| 4 | Оформлення матеріалів кваліфікаційної роботи | 04.12.21-07.12.21 | |
| 5 | Подання кваліфікаційної роботи керівникові та | 08.12.21-09.12.21 | |
| | її попередній захист | | |
| 6 | Подання кваліфікаційної роботи на | 10.12.21-11.12.21 | |
| | рецензування | | |
| | | | |
| | | | |
| | | | |

Дата видачі завдання 08 листопада 2021 р.

Студент _____
(підпис)

Керівник роботи _____ Доц. Ільїна І.В.
(підпис) (посада, прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка кваліфкаційної роботи: 71 с., 26 рис., 3 дод., 13 джерел.

БАЗА ДАНИХ, OLAP, DATA MINING, BIG DATA, BUSINESS INTELLIGENCE, СУДИ.

Об’єктом дослідження є завдання знаходження прихованих закономірностей, а предметом дослідження - рішення у судових справах.

Мета роботи - аналіз записів, отриманих в результаті обробки документів і рішень судів за допомогою методів і засобів технології Business Intelligence.

В процесі дослідження проводився аналіз предметної області судочинства України, крім того був виконаний огляд існуючих аналогів.

У ході виконання кваліфікаційної роботи приведені результати проектування і реалізації бази даних, а також проектування і розробка інструментів для вилучення параметрів, необхідних для проведення аналізу, і наповнення бази даних судочинства.

Приведені результати OLAP аналізу цих судових рішень. У майбутньому, розроблені інструменти можуть дозволити зацікавленим людям самостійно отримувати дані і проводити їх аналіз.

# ABSTRACT

Bachelor’s thesis: 71 pages, 26 figures, 3 appendix, 13 sources.

DATABASE, OLAP, DATA MINING, BIG DATA, BUSINESS INTELLIGENCE, COURTS.

The object of research is the task of finding hidden patterns, and the subject of research - decisions in court cases.

The major goal of this tesis is to analyze the records obtained as a result of processing documents and court decisions using the methods and tools of Business Intelligence technology.

In the course of the research the analysis of the subject area of justice of Ukraine was carried out, besides the review of existing analogues was carried out.

In order to get the results of the design and implementation of the database, as well as the design and development of tools for extracting the parameters required for analysis, and filling the database of litigation.

The results of OLAP analysis of these court decisions are given. In the future, the developed tools can allow interested people to independently obtain data and analyze it.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ ..... 7
ВСТУП ..... 8
1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ СУДОЧИНСТВА ..... 10
1.1 Актуальність аналізу даних судочинства ..... 10
1.2 Аналіз структури даних судових рішень ..... 11
1.3 Огляд аналогів ..... 13
1.3.1 «ZakonOnline» ..... 13
1.3.2 «Pravosud» ..... 15
1.4 Постановка задачі ..... 17
2 ПРОЕКТУВАННЯ ІНСТРУМЕНТІВ ДЛЯ ОБРОБКИ ДАНИХ ..... 19
2.1 Вибір інструментів розробки ..... 19
2.2 Проектування бази даних судочинства ..... 20
2.3 Проектування інструмента для вилучення атрибутів судового рішення ..... 22
2.4 Проектування інструмента для наповнення бази даних ..... 30
3 РОЗРОБКА ІНСТРУМЕНТІВ ДЛЯ ОБРОБКИ ДАНИХ ..... 33
3.1 Розробка інструменту для вилучення атрибутів судової справи ..... 33
3.2 Розробка інструменту для наповнення бази даних ..... 34
4 АНАЛІЗ ДАНИХ ЗА ДОПОМОГОЮ ТЕХНОЛОГІЙ DATA MINING ..... 35
4.1 Основні типи завдань, які вирішуються технологіями Data Mining ..... 35
4.2 Постановка задач ..... 37
4.3 Вибір алгоритмів для вирішення поставлених задач ..... 38
4.4 Підбір алгоритмів для вирішення вибраних задач. Опис алгоритмів ..... 40
4.5 Основні етапи DataMining ..... 43

4.6 Вибір та перевірка оптимального алгоритму для майбутнього використання ..... 46
4.7 Алгоритм t-SNE ..... 53
4.8 Обробка даних ..... 54
4.9 Результат обробки та аналізу даних ..... 57
ВИСНОВКИ ..... 60
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ..... 61
ДОДАТОК А Фізична схема бази даних ..... 63
ДОДАТОК Б Схема куба даних судочинства ..... 64
ДОДАТОК В Графічний матеріал кваліфікаційної роботи ..... 65

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ПІБ – прізвище ім’я по батькові;

БД – база даних;

BI – Business Intelligence;

OLAP (Online Analytical Processing) – оперативний аналіз даних;

Data Mining – видобуток даних, інтелектуальний аналіз;

КУ – Кодекс України;

Microsoft SQL Server – система управління реляційними базами даних, розроблена корпорацією Microsoft.

SQL Server Express LocalDB – спрощена версія SQL Server, яка має всі функції програмування бази даних SQL Server.

# ВСТУП

У наш час відбувається інтенсивне накопичення величезних обсягів даних різного типу в різних предметних областях, що вимірюються в петабайтах, це в свою чергу дає можливість вирішувати завдання отримання нових фактів, залежностей і прихованих кореляцій, а також дозволяє вирішувати деякі аналітичні завдання, такі як прогнозування, перевірка статистичних гіпотез, розрахунок агрегатних показників тощо.

Об’єктом дослідження є завдання знаходження прихованих закономірностей, а предметом дослідження – рішення у судових справах.

Мета роботи – аналіз записів, отриманих в результаті обробки документів та рішень судів України за допомогою методів та засобів технологій OLAP та Data Mining.

Аналіз судочинства досить актуальний, оскільки результати аналізу будуть цікавими як великим організаціям, таким як правоохоронні органи та правозахисні організації, так і окремим особам, таким як незалежні юристи та люди, які перебувають під слідством. Для великих організацій буде більш цікавий глибокий аналіз для пошуку прихованих залежностей і відхилень, а окремим особам аналіз за конкретними суддями та адвокатами, який може допомогти передбачити, яке рішення може бути винесене певним суддею або вибрати найбільш ефективного адвоката.

Значимість даної роботи полягає в тому, що розроблені інструменти дозволять зацікавленим людям самостійно отримувати та аналізувати дані.

У першому розділі даної роботи проведено аналіз предметної галузі, обґрунтовано актуальність дослідження та викладено результати огляду аналогів.

У другому розділі представлені проектування та реалізація бази даних, а також наведено результати проектування інструментів для вилучення атрибутів із текстів судових справ та наповнення бази даних судочинства.

У третьому розділі представлені результати розробки інструментів для отримання атрибутів з текстів судових справ та наповнення бази даних судочинства.

У четвертому розділі наведено результати OLAP аналізу даних, вилучених із рішень у судових справах.

Вилучення даних виконано у середовищі PyCharm, мова програмування – Python. Запис даних у БД виконано у середовищі Visual Studio, мова програмування – C#. Для створення БД використовувалося середовище Microsoft SQL Server Management Studio. OLAP та Data Mining аналіз виконано у середовищі Visual Studio, з використанням Microsoft Excel.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ СУДОЧИНСТВА

## 1.1 Актуальність аналізу даних судочинства

Вперше термін "великі дані" використав Кліффорд Лінч у випуску журналу Nature від 3 вересня 2008 року, з темою "Як можуть вплинути на майбутнє науки технології, що відкривають можливості роботи з великими обсягами даних?". Після цього напрямок обробки великих даних почав стрімко розвиватися. Так, вже у 2010 році почали з’являтися перші програмні продукти та рішення для обробки великих даних.

22 грудня 2005 року набрав чинності Закон №3262 "Про доступ до судових рішень" [1], який передбачає розміщення рішень у судових справах на сайтах судів [2]. Завдяки цьому закону, з’явилася можливість відкритого доступу до судових рішень і, відповідно, можливість виконати аналіз справ судочинства.

Таким чином, є не тільки інструменти, необхідні для аналізу, але й дані судочинства знаходяться у відкритому доступі. Вже є кілька проектів, що займаються аналізом судових справ, однак ці проекти або не надають належної можливості аналізу, або набір функцій досить обмежений і доступ до них платний.

Аналіз судочинства досить актуальний, оскільки результати аналізу будуть цікавими як великим організаціям, таким як правоохоронні органи та правозахисні організації, так і окремим особам, таким як незалежні юристи та люди, які перебувають під слідством. Оскільки дозволять не лише зібрати статистичну інформацію, а й побачити більш глибокий аналіз щодо окремих суддів, адвокатів та інших атрибутів судових справ для визначення різних прихованих взаємозв’язків між атрибутами справи та для пошуку відхилень. Результат аналізу може допомогти передбачити, яке рішення може бути винесене певним суддею або допомогти вибрати найбільш ефективного

адвоката.

Завданням даної роботи є аналіз рішень у судових справах, за допомогою методів і засобів Business Intelligence, для виявлення прихованих закономірностей.

## 1.2 Аналіз структури даних судових рішень

Оскільки справи судочинства з’явилися у відкритому доступі порівняно недавно, структура офіційних веб-сайтів судів, де розміщуються судові справи, зазнавала змін. Так, на деякий час на сайтах судів з’являлися дані про рух справи, проте досить швидко ці дані було прибрано. Тому довелося орієнтуватися виключно на вміст судових актів, це пов’язано з крайньою роз’єднаністю судів щодо публікацій руху справ, а також з відсутністю такої інформації на деяких сайтах судів. На сайтах судів справи викладено після деякої обробки, яка виділила деякі атрибути: ім’я судді, дата суду, вид справи, підвид справи, інстанція, регіон, місто. Однак це лише частина необхідних атрибутів. Тому був проведений аналіз текстів судових справ для визначення списку атрибутів, які необхідно і можливо отримати. Основна складність у тому, що тексти судових справ досить структуровані і деякі дані, наприклад, імена деяких відповідачів, знеособлені. Текст судової справи можна поділити на чотири частини:

- у вступній частині рішення суду повинні бути зазначені дата та місце прийняття рішення суду, найменування суду, який прийняв рішення, склад суду, секретар судового засідання, сторони, інші особи, які беруть участь у справі, їхні представники, предмет спору чи заявлену вимогу;

- описова частина рішення суду повинна містити вказівку на вимогу позивача, заперечення відповідача та пояснення інших осіб, які беруть участь у справі;

- у мотивувальній частині рішення суду мають бути зазначені обставини справи, встановлені судом; докази, на яких ґрунтуються висновки

суду про ці обставини; докази, якими суд відкидає ті чи інші докази; закони, якими керувався суд;

- резолютивна частина рішення суду повинна містити висновки суду про задоволення позову або про відмову в задоволенні позову повністю або в частині, вказівку на розподіл судових витрат, строк та порядок оскарження рішення суду. Резолютивна частина рішення суду, прийнятого мировим суддею, також має містити вказівку на строк та порядок подання заяви про складання мотивованого рішення суду.

Таким чином, необхідні атрибути судової справи в основному містяться в основній та резолютивній частинах справи.

В результаті аналізу текстів судових справ та сайтів судів, було виявлено нуступне. За типом судові справи поділяються на:

- адміністративні;
- цивільні;
- кримінальні;
- за матеріалами.

Залежно від типу судової справи можна виділити типи судових рішень:

- постанови;
- рішення;
- визначення;
- вироки.

Кожне судове рішення має такі загальні атрибути:

- номер справи;
- місто;
- регіон;
- назву суду;
- суддя (ПІБ, стать);
- дата суду;
- нормативний акт (номер статті, назва кодексу);
- дата набуття чинності;

- рішення у справі;
- фізичні особи, які фігурують у справах.

Крім основних атрибутів, судове рішення може мати додаткові атрибути, залежно від типу справи та винесеного рішення:

- покарання;
- розмір покарання;
- міра покарання;
- відповідач;
- позивач;
- ПІБ адвоката;
- юридичні особи, що фігурують у справах.

В однії справі може бути кілька рішень, якщо у справі фігурує кілька відповідачів. У свою чергу, одне рішення може містити кілька покарань, у разі винесення декількох видів покарань (арешт, стягнення тощо). Крім того, кожне рішення має свій список статей, на підставі яких було винесено рішення.

## 1.3 Огляд аналогів

### 1.3.1 «ZakonOnline»

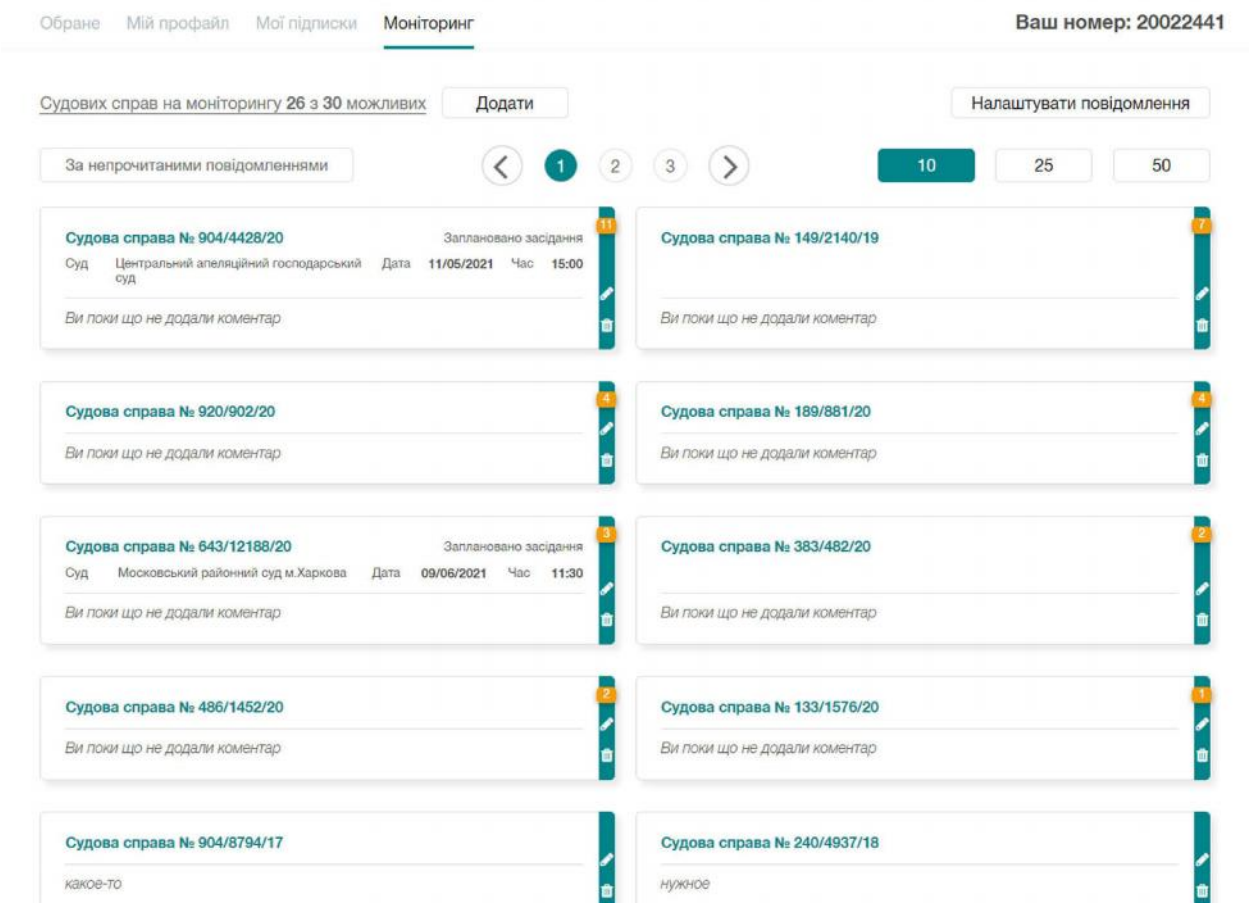


Рисунок 1.1 – Приклад інтерфейсу ZakonOnline

Сервіс «ЗаконОнлайн» позиціонує себе як універсальний робочий простір для юристів, адвокатів, суддів, прокурорів, нотаріусів та інших представників правничої професії. «ЗаконОнлайн» – це комерційний проект, який передбачає підписку починаючи від місяця до року [3].

Пошук законодавчих документів у розділі «Нормативні акти» можна здійснювати за такими категоріями:

- за типом (закон, кодекс, постанова тощо);
- за видавником (Міністерство освіти і науки України, Генеральна прокуратура України тощо);
- за датою.

Знайти потрібне судове рішення можна із застосуванням таких фільтрів:

- за інстанцією (перша, апеляційна, касаційна);
- за типом (рішення, постанова, ухвала тощо);
- за категорією;
- за судочинством;
- за регіоном;
- за судом;
- за суддею;
- за періодом.

Позитивні якості Сервісу:

- зручний інтерфейс користувача;
- тестовий період на 1 день;
- служба підтримки клієнтів.

Негативні якості Сервісу:

- висока ціна користування сервісу;
- багато потрібних функцій відсутні або недостатньо розвинені.

### 1.3.2 «Pravosud»

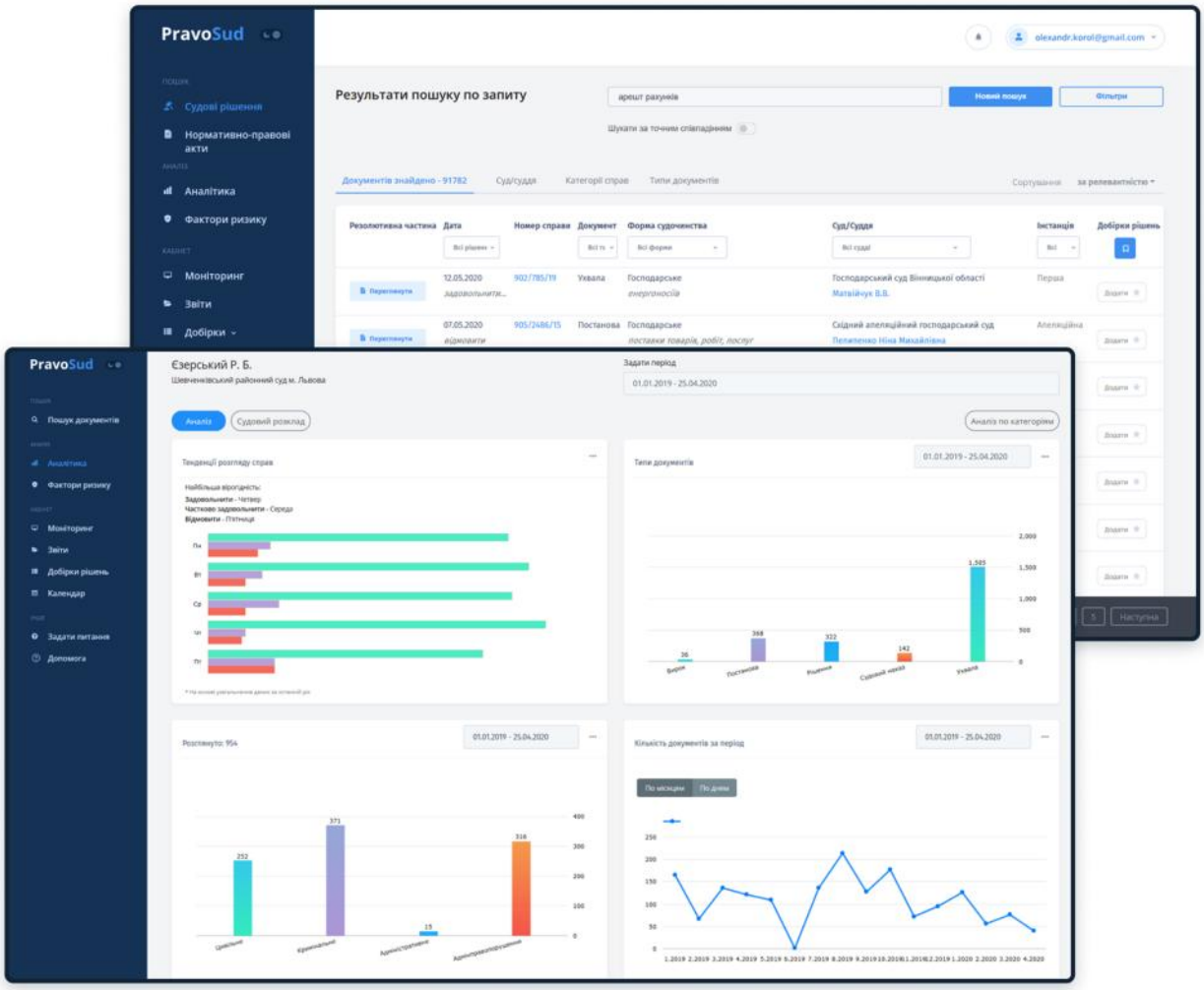


Рисунок 1.2 – Приклад інтерфейсу Pravosud

Сервіс «Pravosud» позиціонує себе як універсальна платформа для пошуку та аналізу документів з різних державних реєстрів. «Pravosud» – це комерційний проект, який передбачає підписку починаючи від місяця до року [4].

Сервіс дає змогу пошука судових рішень та практики ЄСПЛ, пошука законодавстсва, правових позицій Верховного Суду, аналізу практики суддів, аналізу контрагентів та моніторингу судової практики.

Позитивні якості Сервісу:

- швидкий пошук документів (3-6 секунд);
- понад 70 млн. документів в системі;
- тестовий період на 3 дні;

- високі аналітичні можливості.

Негативні якості Сервісу:

- досить незручний інтерфейс;
- висока ціна користування сервісу.

## 1.4 Постановка задачі

В результаті аналізу предметної області, що включає огляд аналогів, було виявлено, що бази даних судочинства немає у зручному та доступному вигляді. Тому виникла потреба самостійно розробити БД, витягти параметри судової справи та наповнити БД. На рисунку 1.3 представлено загальну схему підготовки даних для аналізу, яка демонструє етапи, через які необхідно пройти для проведення аналізу даних судочинства.

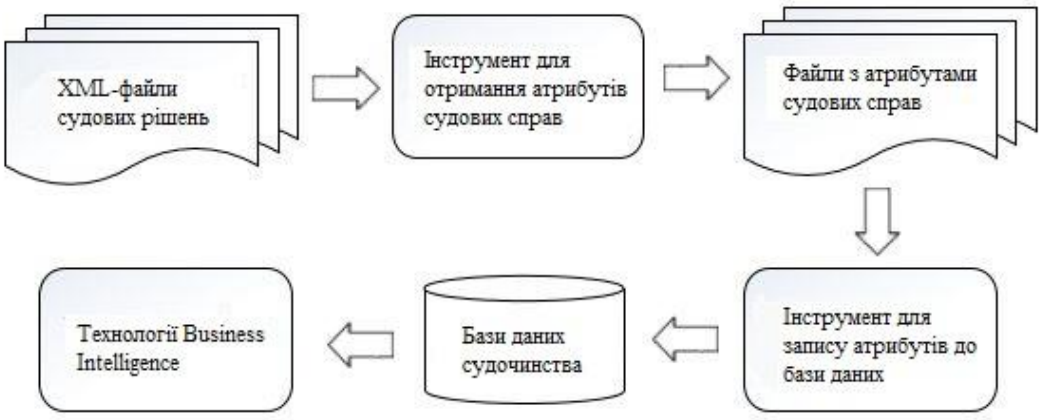


Рисунок 1.3 – Загальна схема підготовки даних до аналізу

Щоб виконати аналіз даних судочинства, необхідно виконати наступні задачі:

Задача 1. Вибір інструментів проектування, розробки та проведення аналізу.

Задача 2. Проектування та реалізація бази даних.

Задача 3. Проектування та розробка інструментів для вилучення параметрів та наповнення бази даних (додаток А).

Задача 4. Проведення OLAP аналізу даних (додаток Б).

# 2 ПРОЕКТУВАННЯ ІНСТРУМЕНТІВ ДЛЯ ОБРОБКИ ДАНИХ

## 2.1 Вибір інструментів розробки

Для вилучення параметрів судових справ використовувалася мова програмування Python серед розробки PyCharm. Основними перевагами обраного середовища та мови є:

- легкочитаний синтаксис;
- простий у навчанні;
- високорівнева мова;
- об’єктно-орієнтованість;
- великий набір модулів, що підключаються.

Для створення БД використовувалося середовище Microsoft SQL Server Management Studio за рахунок таких переваг:

- зручний графічний інтерфейс;
- багаті можливості;
- висока продуктивність;
- надійність.

Як середовище розробки програми для наповнення БД була використана Visual Studio. Вибір даного програмного забезпечення обумовлений такими можливостями:

- робота з об’єктами баз даних в браузері об’єктів SQL Server;
- визначення таблиць у новому конструкторі таблиць;
- розробка та тестування додатків баз даних у SQL Server Express LocalDB;
- запис асинхронного коду простим та інтуїтивно зрозумілим способом;
- отримання відомостей про об’єкт, що викликає, які допомагають з трасуванням та налагодженням.

Як мову програмування була обрана мова C#, так як вона має наступні

переваги:

- об’єктно-орієнтованість;
- орієнтація на безпеку коду;
- багатофункціональність;
- відносна простота.

Для побудови OLAP-кубів та проведення OLAP-аналізу використовувалося середовище Visual Studio, оскільки воно має такі переваги:

- засоби Business Intelligence;
- робота з базами даних SQL Server;
- зручний графічний інтерфейс.

## 2.2 Проектування бази даних судочинства

На основі проведеного аналізу структури судових рішень було спроектовано схему бази даних. База даних складається з 25 сутностей зі своїми наборами атрибутів:

- регіони (ідентифікатор регіону, назва регіону);
- міста (ідентифікатор міста, ідентифікатор регіону, назва міста);
- форми юридичних осіб (ідентифікатор форми юридичної особи, форма юридичної особи);
- юридичні особи (ідентифікатор юридичної особи, ім’я юридичної особи, ідентифікатор форми юридичної особи);
- юридичні особи справи (ідентифікатор юридичної особи, ідентифікатор справи);
- фізичні особи (ідентифікатор фізичної особи, ім’я фізичної особи);
- фізичні особи справи (ідентифікатор фізичної особи, ідентифікатор справи);
- позивачі (ідентифікатор позивача, тип позивача, ім’я позивача);
- суди (ідентифікатор суду, назва суду, ідентифікатор міста);

- види справ (ідентифікатор виду справ, назва виду справ);

- підвиди справ (ідентифікатор підвиду справ, назва підвиду справ);

- справи (ідентифікатор справи, номер справи, ідентифікатор суду, ідентифікатор судді, ідентифікатор адвоката, категорія, дата суду, дата набрання чинності, ідентифікатор позивача, ідентифікатор виду справи, ідентифікатор підвиду справи, ідентифікатор інстанції);

- судді (ідентифікатор судді, ПІБ, стать судді);

- рішення (ідентифікатор рішення, ідентифікатор справи, ідентифікатор відповідача, ідентифікатор типу рішення);

- рип рішення (ідентифікатор типу рішення, тип рішення); • Інстанції (ідентифікатор інстанції, назва інстанції);

- покарання (ідентифікатор покарання, ідентифікатор рішення, ідентифікатор типу покарання, розмір покарання, ідентифікатор типу міри покарання);

- статті рішення (ідентифікатор рішення, ідентифікатор статті); • Адвокати (ідентифікатор адвоката, ПІБ);

- типи покарань (ідентифікатор типу покарання, назва типу покарання);

- типи заходів покарань (ідентифікатор типу міри покарання, назва типу міри покарання);

- кодекси (ідентифікатор кодексу, назва кодексу);

- статті (ідентифікатор статті, статті, ідентифікатор кодексу, номер статті, ідентифікатор категорії статті);

- категорії статей (ідентифікатор категорії статей, назва категорії статей).

Кожна сутність має свій унікальний ідентифікатор, який дозволяє однозначно визначити елементи сутностей, а також допомагає запобігти дублювання даних. За винятком трьох сутностей (Статті рішення, Юридичні особи справ, Фізичні особи справ), які мають складовий ключ, що складається із двох відповідних ідентифікаторів.

Крім того, практично всі сутності пов’язані за допомогою бінарних

зв’язків, у даному випадку використовується зв’язок один до багатьох. Таким чином, Рішення може належати тільки одній Справі, але у Справи може бути безліч Рішень. Так само пов’язані й інші сутності: Юридичні особи та Форми юридичних осіб, Міста і Регіони, Справи і Позивачі, Суди і Міста, Справи і Суди, Справи та Види справ, Справи та Підвиди справ, Справи та Інстанції, Справи та Судді, Справи та Адвокати, Рішення та Тип рішення, Рішення та Відповідачі, Покарання та Рішення, Покарання та Типи покарань, Покарання та Типи заходів покарань, Статті та Кодекси, Статті та Категорії статей. Виняток становлять сутності: Рішення та Статті, Справи та Юридичні особи, Справи та Фізичні особи. У цих трьох випадках між сутностями використовується зв’язок багато до багатьох, сформований за рахунок відповідних проміжних таблиць. База даних створена у середовищі Microsoft SQL Server Management Studio відповідно до даних проектування. У додатку представлена фізична схема БД.

## 2.3 Проектування інструмента для вилучення атрибутів судового рішення

Вилучення даних являє собою складний процес отримання текстових даних з одного або різних джерел та приведення його до уніфікованого виду. Зазвичай об’єктами парсингу є веб-сайти, але можуть бути xml і pdf документи, і будь-які інші текстові файли та документи, що зберігають інформацію у текстовому вигляді.

У цій роботі як об’єкти для парсингу виступають xml документи. Xml файли суду є набором декількох параметрів в xml тегах (ім’я судді, дата суду, вид справи, підвид справи, інстанція, регіон, місто) і текст судової справи.

Для отримання параметрів текстів рішень був розроблений спеціальний інструмент. В якому застосовуються такі методи та алгоритми:

- алгоритм шинглів — алгоритм, розроблений для пошуку копій та дублікатів тексту, що розглядається, у веб-документі. Шингли - виділені з

тексту підпослідовності слів, що йдуть один за одним. Вибірка відбувається внахлест, тобто кожний новий шингл зсувається на одне слово [5];

- відстань Левенштейна між двома рядками – це мінімальна кількість операцій вставки одного символу, видалення одного символу та заміни одного символу на інший, необхідних для перетворення одного рядка на інший [6];

- регулярні вирази — формальна мова пошуку та здійснення маніпуляцій з підрядками в тексті, що базується на використанні метасимволів. По суті, це рядок-зразок («шаблон», «маска»), що складається з символів і метасимволів і задає правило пошуку [7].

Для кожного атрибуту судової справи була написана власна функція, з відповідними регулярними виразами та алгоритмом пошуку.

На рисунку 2.1 представлений алгоритм отримання атрибутів з текстів судових справ. Частина функцій (витяг номера справи, ПІБ адвоката, позивача, фізичних осіб, юридичних) не залежать від результатів інших функцій. Однак деяким функціям потрібні дані інших функцій. Так, для отримання статі судді потрібне ім’я судді, для рішень потрібен список відповідачів, для покарань список відповідачів та рішень, для статей список відповідачів та рішень.

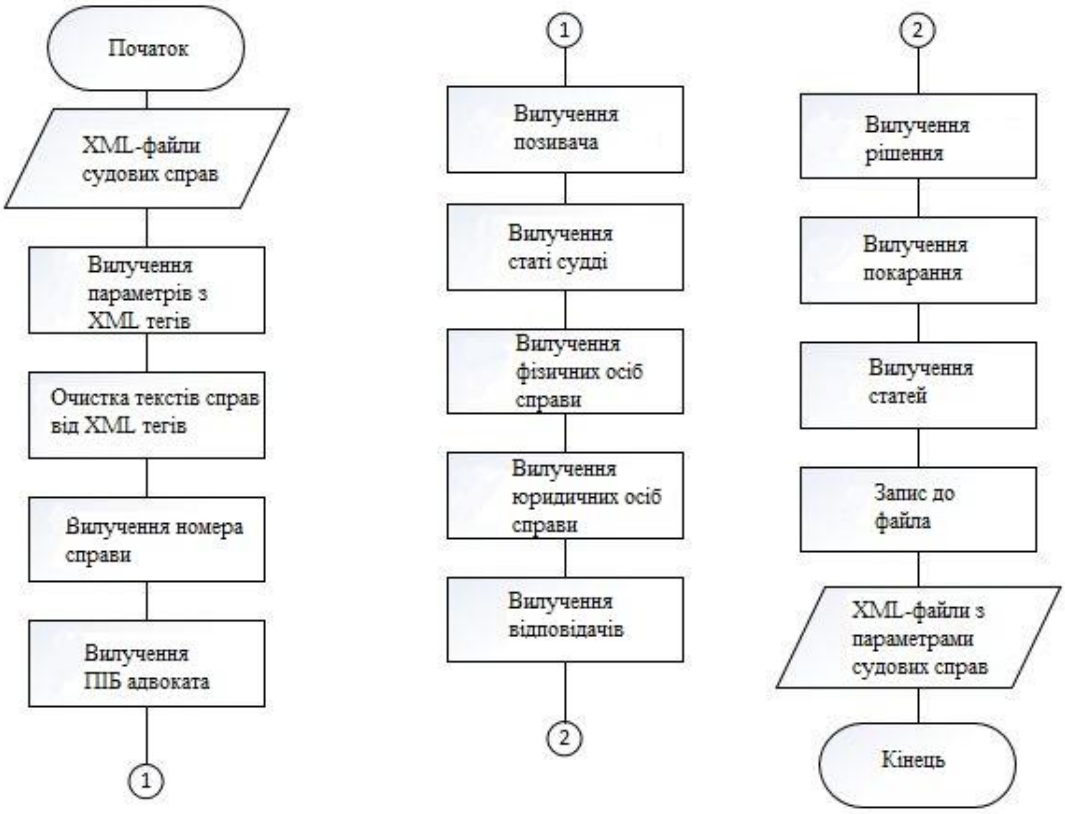


Рисунок 2.1 – Схема алгоритму вилучення атрибутів з текстів судових рішень

Вилучення параметрів з xml тегів полягає у вилученні тексту, укладеного в певних xml тегах. Виділення з тексту справи резолютивної частини здійснюється з використанням регулярних виразів. Так, у тексті виконується пошук заданих регулярних виразів (перед резолютивною частиною ставляться такі слова: постанова, рішення тощо) при знаходженні необхідного виразу виконується обрізка тексту від знайденого виразу і до кінця тексту.

Виділення із резолютивної частини справи фрагмента, що відноситься до відповідача, здійснюється з використанням методу шинглів та відстані Левенштейна. Оскільки ім’я відповідача може знаходитися в різних відмінках у списку відповідачів та в тексті рішення, текст розбивається на шингли (з довгою рівною кількості слів у імені відповідача) та використовуючи відстань Левенштейна порівнюються шингли рішення та шингл імені

відповідача.

Алгоритм для вилучення фізичних осіб знаходить у тексті усі висловлювання, які відповідають заданому набору регулярних выразів. Після цього, використовуючи відстань Левенштейна, зі списку видаляються всі дублікати. Для пошуку дублікатів використовується відстань Левенштейна, оскільки в тексті справи ті самі особи можуть згадуватися в різних відмінках. Такий самий алгоритм використовується при пошуку юридичних осіб (рисунок 2.2).

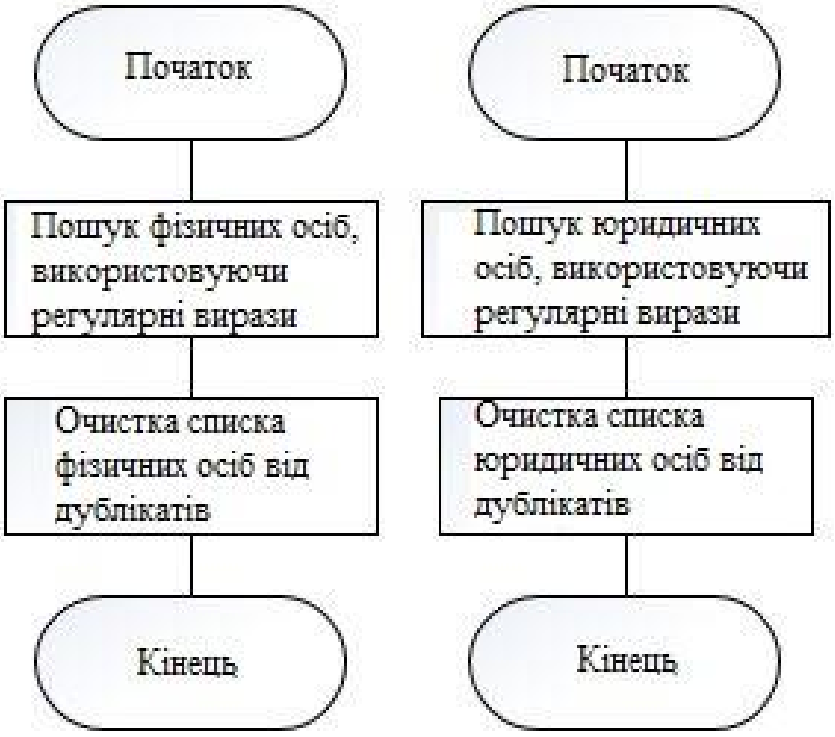


Рисунок 2.2 – Схема алгоритмів для вилучення фізичних та юридичних осіб

Алгоритм для вилучення номера справи знаходить у тексті перше вираз, яке задовольняє заданому набору регулярних висловів і відсікає непотрібну більш частину висловлювання, яка описує слова, після шуканим номером справи. Як результат береться перший знайдений вираз оскільки у

справі може бути лише один номер справи. Такий самий алгоритм використовується при пошуку позивача та адвоката. Алгоритм для вилучення статі судді відрізняється тим, що пошук регулярних виразів відбувається у ПІБ судді (рисунок 2.3).


Рисунок 2.3 – Схема алгоритмів для вилучення номера справи, позивача, адвоката та статі судді

Алгоритм для отримання відповідачів спочатку виділяє частину тексту з резолютивною частиною. Після цього йде за списком фізичних осіб та виділяє з рішення частину тексту, що відноситься до фізичної особи. Потім шукає в тексті задані регулярні вирази, якщо вираз знайдено, то до списку відповідачів додається ім’я поточної фізичної особи. Якщо вирази не знайдені, то алгоритм переходить до наступної фізичної особи. Якщо не було знайдено відповідачів серед фізичних осіб, то за такою самою схемою, як для фізичних осіб, виконується пошук відповідачів серед юридичних осіб (рисунок 2.4).

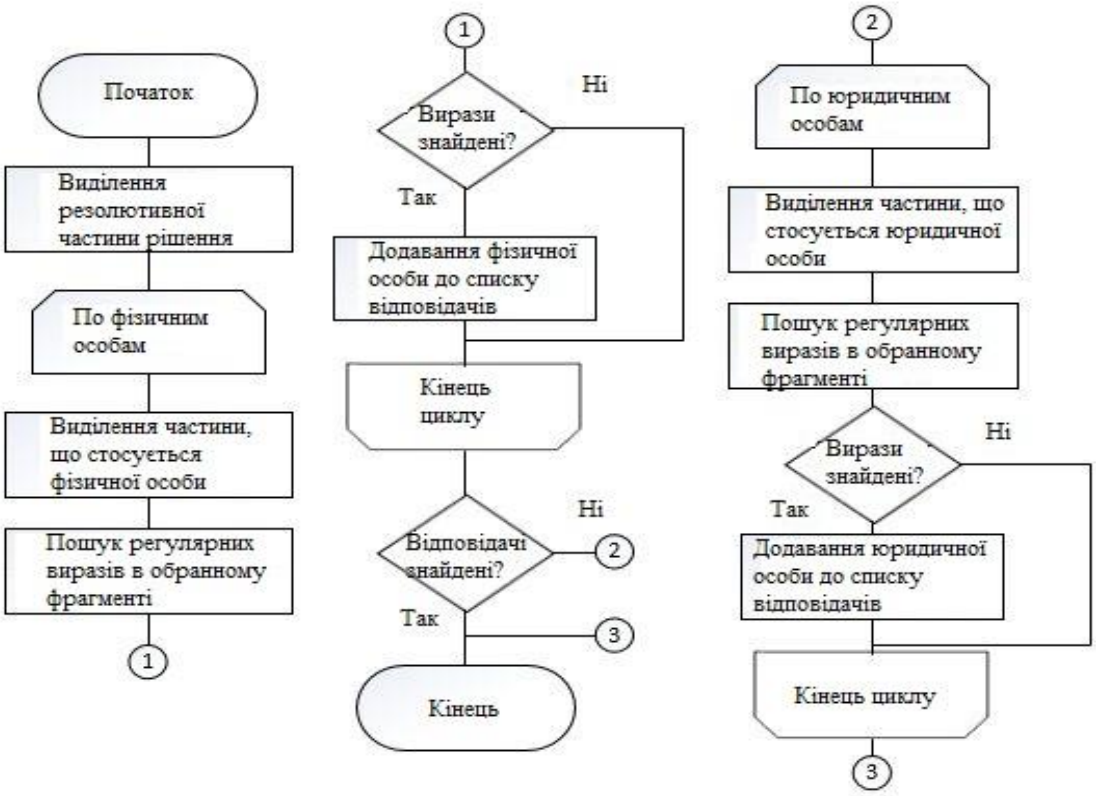


Рисунок 2.4 – Схема алгоритму для вилучення відповідачів

Алгоритм для отримання рішень спочатку виділяє частину тексту з резолютивною частиною. Якщо є відповідачі, алгоритм йде за списком відповідачів і виділяє з рішення частину тексту, що відноситься до відповідача. Потім шукає у тексті задані регулярні висловлювання, якщо вираз знайдено, то списку рішень додається рішення для поточного відповідача. Якщо вирази не знайдені, то алгоритм веде пошук у всьому рішенні у справі. Якщо ж відповідачів немає, то пошук регулярних виразів ведеться у всьому тексті рішення та результат записується до списку рішень (рисунок 2.5).

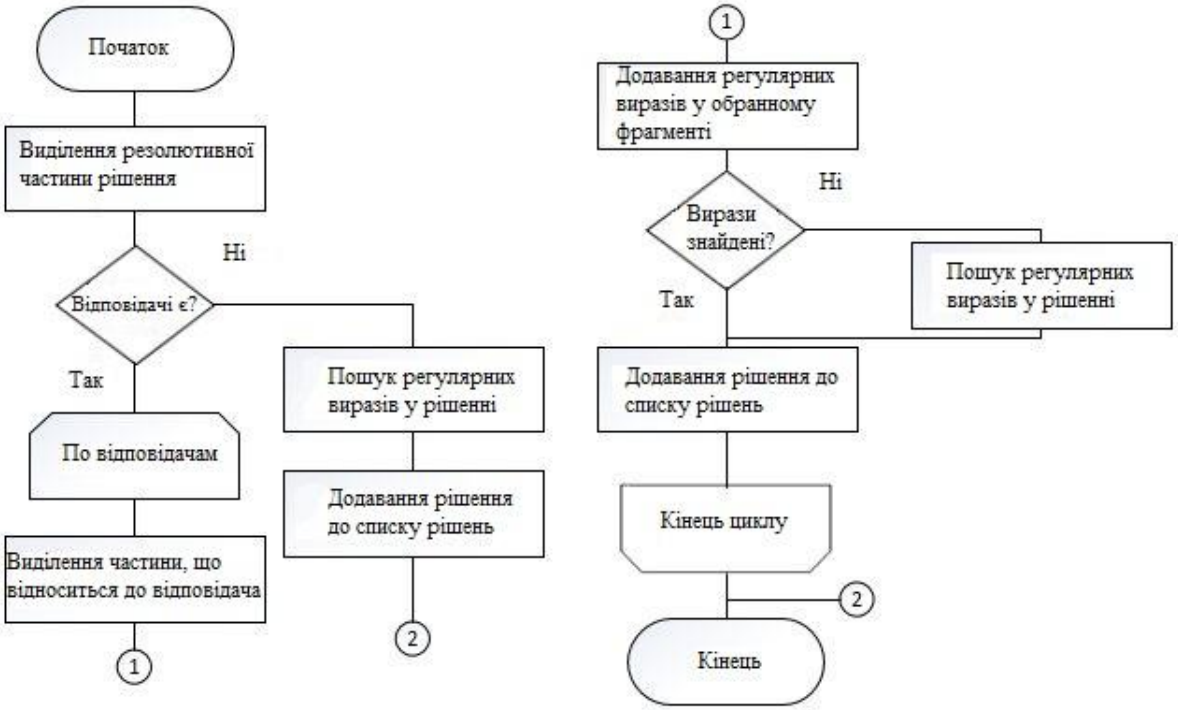


Рисунок 2.5 – Схема алгоритму для вилучення рішень

Алгоритм для отримання статей спочатку виділяє частину тексту з резолютивною частиною. Якщо є відповідачі, алгоритм йде за списком відповідачів і виділяє з рішення частину тексту, що відноситься до відповідача. Потім шукає в тексті регулярні вирази для номерів статей, якщо вирази знайдені, починається пошук назви кодексів поблизу номера статті, з використанням регулярних виразів. Якщо назва кодексу знайдена, то до списку статей додається номер статті та назва кодексу для відповідача. Якщо кодекс не знайдено, то алгоритм переходить до наступного відповідача. Якщо у виділеній частині рішення з відповідачем були знайдені номери статей, то пошук проводиться у всьому решении. Якщо номери було знайдено, далі йде пошук назви кодексу як було описано раніше. Якщо номерів статей немає у всьому рішенні, алгоритм завершує виконання. Якщо ж відповідачів немає, то пошук регулярних виразів проводиться у тексті рішення. І якщо номерів статей немає, алгоритм завершує виконання (рисунок 2.6).

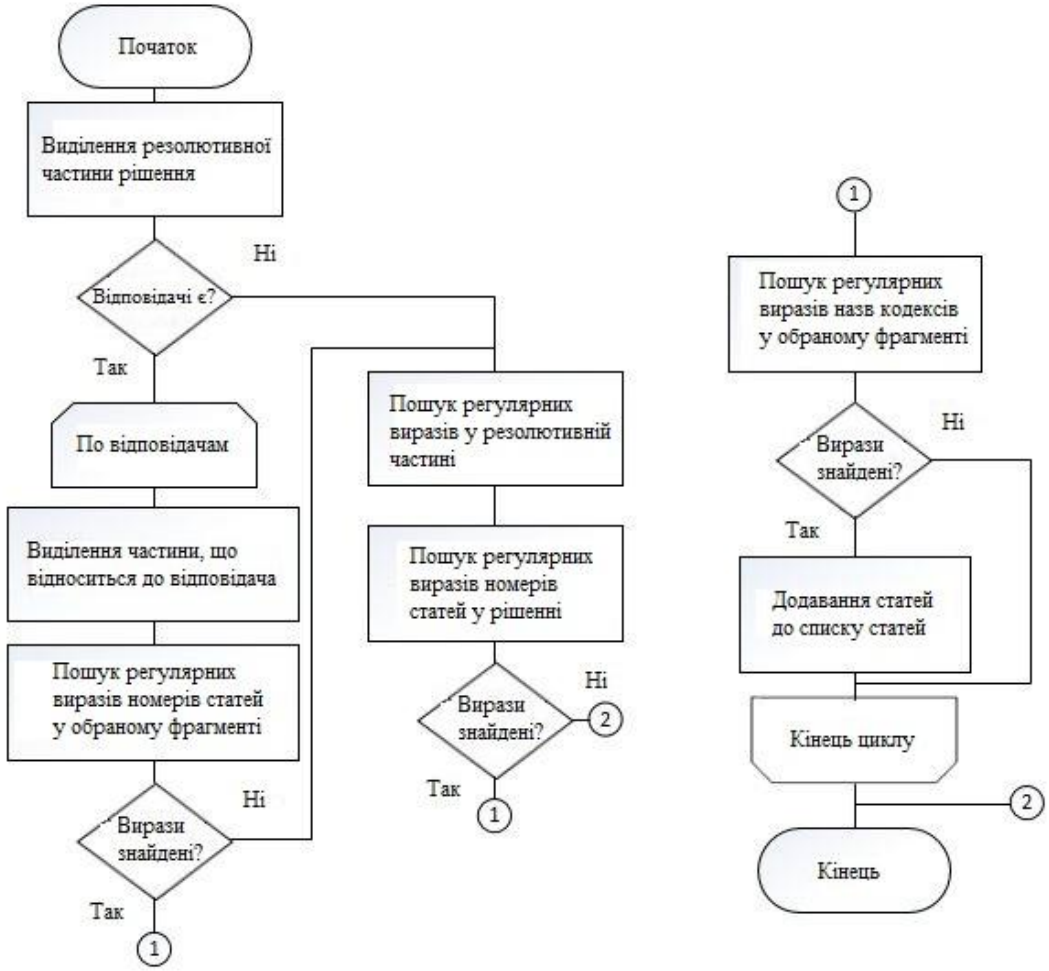


Рисунок 2.6 – Схема алгоритму для вилучення статей

Алгоритм для отримання покарань спочатку виділяє частину тексту з резолютивною частиною. Якщо є рішення, алгоритм йде за списком і виділяє з рішення частину тексту, що відноситься до поточного рішення. Потім шукає в тексті задані регулярні вирази покарань, якщо вираз знайдено, то список покарань додається покарання для поточного рішення. Якщо вирази не знайдені, то алгоритм переходить до наступного і пошук ведеться у всьому рішенні у справі. Якщо ж рішень немає, алгоритм завершує роботу (рисунок 2.7).

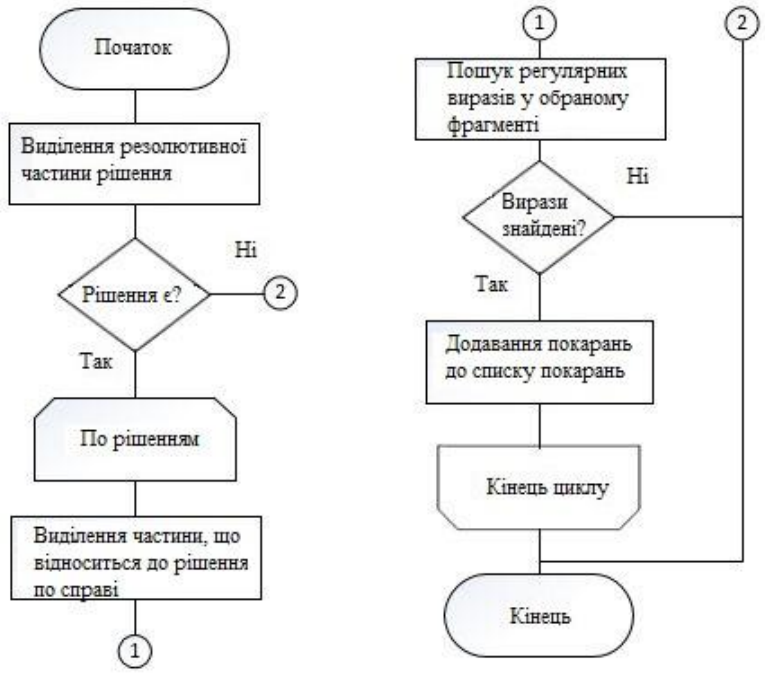


Рисунок 2.7 – Схема алгоритму для вилучення покарань

Інші атрибути (ім’я судді, вид та підвид справи, інстанція, дата суду, місто та регіон) знаходяться в XML тегах і дуже просто витягуються. Вилучені параметри записуються в XML файли, кожен атрибут записується на окремий рядок. Оскільки атрибутів одного типу може бути кілька, то у файл записується рядок наступної структури: «Назва атрибута: Атрибут 1; …; Атрибут N». Крім того, атрибут може мати кілька параметрів (рішення = ім’я відповідача + тип рішення). Тобто, «Атрибут 1» = «Параметр 1; …; Параметр N».

## 2.4 Проектування інструмента для наповнення бази даних

Наповнення БД є досить трудомістким заняттям, оскільки необхідно враховувати безліч факторів. Так, для кожної сутності необхідно виконувати перевірку на дублікати. Перевірка виконується з використанням відстані Левенштейна, причому для атрибутів з іменами потрібно порівнювати не ім’я повністю, а окреме прізвище та ім’я по батькові. Оскільки у справах можуть

зустрічатися однофамільці, тому в той час як прізвище може трохи відрізнятися, за рахунок різних відмінків, то ім’я, по-батькові повинні перевірятись на повний збіг. Також ім’я та по батькові може бути записано у вигляді ініціалів або повне ім’я та по батькові. Також необхідно враховувати зв’язок між різними сутностями. Так, наприклад, не можна спочатку створити рішення щодо суду якщо цієї справи ще немає, оскільки сутність «Рішення» посилається на ідентифікатор сутності «Діло». Крім того, необхідно враховувати тип атрибуту, так як БД вимагає точного їхнього дотримання. Кожна сутність має свій набір атрибутивної інформації, що включає первинний і зовнішні ключі. Кожен екземпляр сутності має свій унікальний ключ (він може складатися більш ніж з одного поля), який однозначно ідентифікує екземпляр. Зовнішні ключі, дозволяють однозначно визначити, на який екземпляр сутності 2 посилається екземпляр сутності 1. Спрощений алгоритм запису в БД атрибутів судової справи:

- першими заповнюються екземпляри сутностей, які не мають зовнішніх ключів (Адвокати, Види справ, Підвиди справ, Інстанції, Міста, Судді, Позивачі);

- запис міста за допомогою унікального ключа регіону;

- запис суду за допомогою унікального ключа міста;

- запис справи з усіма його атрибутами та зовнішніми посиланнями;

- записуються екземпляри сутності Юридичні особи та одночасно заповнюється проміжна таблиця юридичних осіб, які беруть участь у справі;

- записуються екземпляри сутності Фізичні особи та одночасно заповнюється проміжна таблиця фізичних осіб, які беруть участь у справі;

- після цього йде запис рішень, винесених у справі. Причому для кожного рішення спочатку записуються у відповідні сутності відповідачів та типів винесеного рішення;

- далі для кожного рішення записуються покарання, винесені за цим рішенням. Одночасно для кожного покарання записується тип покарань та тип міри покарання у відповідній сутності;

- також для кожного рішення записуються статті, на які воно посилається. Перед записом статті записується назва кодексу цієї статті у сутність, яка зберігає кодекси. Після запису статті у сутність додається запис до проміжної таблиці статей рішень. Цей алгоритм виконується для кожного файлу з параметрами судової справи.

# 3 РОЗРОБКА ІНСТРУМЕНТІВ ДЛЯ ОБРОБКИ ДАНИХ

## 3.1 Розробка інструменту для вилучення атрибутів судової справи

Інструмент для отримання атрибутів судової справи реалізований мовою Python з використанням його різних розширень, що дозволяють працювати над вилученням текстової інформації. Основною проблемою при реалізації інструменту було те, що у судових справ немає єдиної структури та імена відповідачів знеособлені. Тому для деяких параметрів доводилося становити досить великий список регулярних виразів. Причому ці списки доводилося постійно доповнювати, оскільки рішення у різних справах досить сильно відрізнялися. Отже, реалізація інструменту проводилася ітеративно. В результаті вдалося отримати такі дані:

- номер справи;
- область;
- місто;
- суд;
- вид справи;
- тип справи;
- інстанція;
- ПІБ Судді;
- стать судді;
- дата набуття чинності;
- дата суду;
- винесене рішення;
- назва нормативного акта;
- стаття;
- наказание;
- размер наказания;
- адвокат;

- відповідач;
- позивач;
- юридичні особи;
- фізичні особи.

Вилучені атрибути записувалися до XML файлів.

## 3.2 Розробка інструменту для наповнення бази даних

На основі результатів проведеного етапу проектування було реалізовано інструмент для заповнення бази даних судочинства мовою C# у середовищі Visual Studio. На вході інструмент отримує набір Xml файлів.

Основною проблемою розробки інструменту було правильно реалізувати взаємозв’язки між різними сутностями та передбачити всі можливі варіанти значень атрибутів. Так, імена можуть записуватися по-різному: прізвище ім’я по батькові, прізвище ініціали імені та по батькові, ініціали імені та по батькові потім прізвище. Тому спочатку ім’я розбивалося на окремі слова, потім визначалося де прізвище, а де ім’я та по батькові. Після цього проводиться порівняння прізвищ відстанню Левенштейна та окремо порівняння імені та по батькові на збіг.

У результаті роботи інструменту було отримано базу даних, наповнену атрибутами судових справ. Усі атрибути було записано у відповідні поля сутностей.

# 4 АНАЛІЗ ДАНИХ ЗА ДОПОМОГОЮ ТЕХНОЛОГІЙ DATAMINING

## 4.1 Основні типи завдань, які вирішуються технологіями Data Mining

На даний час немає єдиної думки, які саме завдання відносяться до Data Mining, при цьому багато авторитетних джерел перераховують такі основні задачі:

- класифікація;
- кластеризація;
- асоціація;
- послідовність;
- прогнозування
- визначення відхилень;
- оцінювання;
- аналіз зв’язків;
- візуалізація.

**Класифікація** (Classification)

Короткий опис. Найбільш просте та поширене завдання Data Mining. В результаті розв’язання задачі класифікації виявляються ознаки, що характеризують групи об’єктів досліджуваного набору даних – класи; за цими ознаками новий об’єкт можна зарахувати до того чи іншого класу.

Методи розв’язання. Для вирішення задач класифікації можуть використовуватися методи:

- найближчого сусіда (Nearest Neighbor);
- k-найближчого сусіда (k-Nearest Neighbor);
- байєсівські мережі (Bayesian Networks);
- індукція дерев рішень;
- нейронні мережі (neural networks).

**Кластеризація** (Clustering)

Кластеризація є логічним продовженням ідеї класифікації. Це завдання

складніше, особливість кластеризації у тому, що класи об’єктів спочатку не зумовлені. Результатом кластеризації є розбиття об’єктів на групи.

Приклад методу вирішення задачі кластеризації: навчання "без вчителя" особливого виду нейронних мереж – самоорганізованих карт Кохонена.

**Асоціація** (Associations)

Короткий опис. У результаті вирішення завдання пошуку асоціативних правил відшукуються закономірності між пов’язаними подіями у наборі даних. Відмінність асоціації від двох попередніх завдань Data Mining: пошук закономірностей здійснюється не на основі властивостей аналізованого об’єкта, а між кількома подіями, що відбуваються одночасно. Найбільш відомий алгоритм вирішення задачі пошуку асоціативних правил – алгоритм Apriori.

**Послідовність** (Sequence), або послідовна асоціація (sequential association)

Короткий опис. Послідовність дозволяє знайти часові закономірності між транзакціями. Завдання послідовності подібне до асоціації, але її метою є встановлення закономірностей не між одночасно наступаючими подіями, а між подіями, пов’язаними в часі (тобто, що відбуваються з певним інтервалом у часі). Іншими словами, послідовність визначається високою ймовірністю ланцюжка пов’язаних у часі подій. Фактично, асоціація є окремим випадком послідовності з тимчасовим кроком, рівним нулю. Це завдання Data Mining називають завданням знаходження послідовних шаблонів (sequential pattern). Правило послідовності: після події X за певний час відбудеться подія Y. Приклад. Після купівлі квартири мешканці у 60% випадків протягом двох тижнів купують холодильник, а протягом двох місяців у 50% випадків купується телевізор. Вирішення цієї задачі широко застосовується в маркетингу та менеджменті, наприклад, при управлінні циклом роботи з клієнтом (Customer Lifecycle Management).

**Прогнозування** (Forecasting)

Короткий опис. У результаті вирішення завдання прогнозування на основі особливостей історичних даних оцінюються пропущені або майбутні значення цільових чисельних показників. Для вирішення таких завдань широко застосовуються методи математичної статистики, нейронні мережі тощо.

**Визначення відхилень або викидів** (Deviation Detection), аналіз відхилень чи викидів

Короткий опис. Мета розв’язання даної задачі - виявлення та аналіз даних, що найбільш відрізняються від загальної множини даних, виявлення так званих нехарактерних шаблонів.

**Оцінювання** (Estimation)

Завдання оцінювання зводиться до передбачення безперервних значень ознаки.

**Аналіз зв’язків** (Link Analysis) – завдання знаходження залежностей у наборі даних.

**Візуалізація** (Visualization, Graph Mining)

Внаслідок візуалізації створюється графічний образ аналізованих даних. Для вирішення задачі візуалізації використовуються графічні методи, що показують наявність закономірностей даних. Приклад методів візуалізації - представлення даних у 2D та 3D вимірах.

**Підбиття підсумків** (Summarization) – завдання, мета якої – опис конкретних груп об’єктів із аналізованого набору даних.

## 4.2 Постановка задач

На етапі вивчення предметної галузі, проектування об’єктної моделі судових рішень було виявлено такі особливості тексту у рішеннях судів:

- у кожному рішенні присутні текстові маркери розділяючі рішення на частини: вступна частина, описова частина, мотивувальна частина, резолютивна частина;

- особисті дані у деяких рішеннях присутні, у деяких прихованих;

- дані суддів завжди присутні;

- текст містить досить детальну описову частину.

На підставі перерахованих пунктів, а також на підставі факту наявності великої кількості рішень можна зробити висновок про доцільність використання методів Data Mining стосовно даних рішень судів. Також на підставі вищезгаданого слід поставити основні напрямки завдань:

- за допомогою алгоритмів кластеризації провести пошук раніше невідомих залежностей та аномалій у судових рішеннях;

- створення механізму угруповання судових рішень із раніше неструктурованих даних;

- створення механізму аналізу згрупованих судових рішень на основі методів кластеризації;

- пошук відхилень у судових рішеннях для подальшого використання;

- пошук основних атрибутів, що впливають на угруповання в сукупності судових рішень;

- порівняння алгоритмів обробки багатовимірних об’єктів на прикладі судових рішень.

## 4.3 Вибір алгоритмів для вирішення поставлених задач

Для визначення можливості використання даних необхідно зрозуміти структуру даних і спробувати представити дані в числовому вигляді, придатному для використання в різних алгоритмах, а також для виконання поставлених завдань. Оскільки дослідження проводяться фактично рядку з текстом, ми можемо уявити рядок кожного рішення як набір окремих слів. У разі кожне слово виступає додатковою мірою для об’єкта рішення. Наприклад, при описі будь-якого об’єкта використовуються різні атрибути, і що більше атрибутів, то більш виразно буде розподіл об’єктів групи.

У практичному застосуванні дуже легко уявити на прикладі людей,

люди - це об’єкти, на початковому етапі за відсутності додаткових атрибутів (додаткових заходів) весь набір об’єктів відноситься до одного класу “людина”, якщо ввести міру “стать”, то всі об’єкти поділяться на два кластера відповідно “чоловіки” та “жінки”, кожна додаткова якість створюватиме нові групи, також різні комбінації атрибутів будуть давати різну кластеризацію об’єктів.

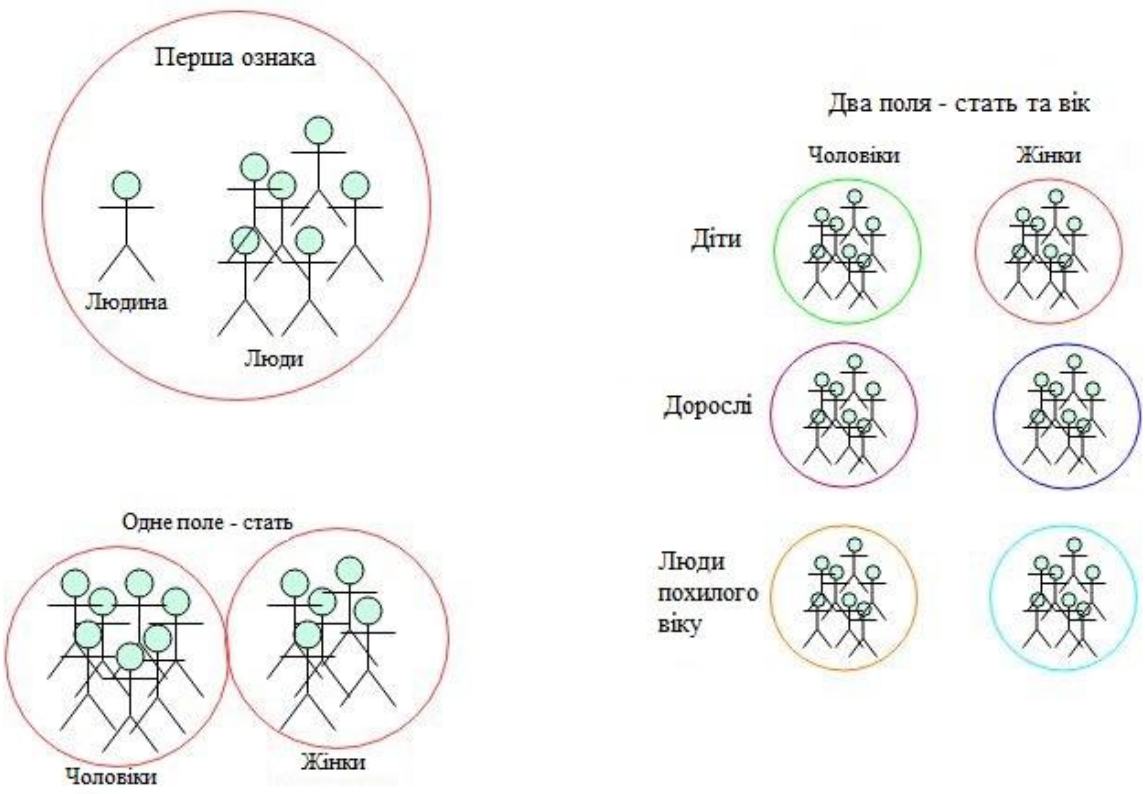


Рисунок 4.1 – Приклад группування одномірних та многомірних об’єктів

У нашому прикладі кожне слово впливає на положення об’єкта в багатовимірному просторі, що надалі визначає його становище у тому чи іншому кластері. Тобто кожне слово - ознака, що дозволяє дуже точно диференціювати кожен об’єкт у загальній картині, і водночас точно розподілять об’єкти за групами. Далі виникає необхідність представлення багатовимірних об’єктів у вигляді зрозумілих до сприйняття та застосування

даних, для цього необхідно кожен об’єкт з багатовимірного простору перетворити на зрозуміле та зручне для сприйняття трьох або двовимірне простір, для чого виникає необхідність вибору відповідного алгоритму, здатного обробити багатовимірні дані. Багатомірний об’єкт несе в собі велику кількість полів, кожне з яких є мірою, або в математичному сенсі кожне поле є координатою в просторі, таким чином наш об’єкт є багатовимірним у багатовимірному просторі. У нашому випадку, об’єкт є одномірним масивом з даними. При ретельнішому опрацюванні тексту судового рішення у цій роботі підготовлений текст суду є масив слів з індексами. Кожне слово може зустрічатися у тексті рішення від 1 та більше кількості разів, ця кількість і є індекс, приклад можна бачити на рисунку 4.2 нижче.

| | |
|---|---|
| суд | 15 |
| постановив | 14 |
| відповідність | 7 |
| викласти | 6 |
| відношення | 6 |
| належний | 6 |
| вищий | 5 |
| перебувати | 5 |
| шкода | 4 |
| обґрунтовано | 4 |
| ТОВ | 4 |
| банківський | 4 |

Рисунок 4.2 – Приклад індексованого об’єкта

## 4.4 Підбір алгоритмів для вирішення вибраних задач. Опис алгоритмів

При виборі алгоритмів для вирішення поставлених завдань необхідно керуватися типами поставлених задач, наявних типів вхідних даних, можливістю використання даних у тих чи інших алгоритмах (можливість математичного представлення даних).

Критеріями вибору алгоритму є:

- обробка багатовимірних об’єктів;

- швидкість виконання;

- наочність і точність результату.

Для вирішення завдань кластеризації з урахуванням перерахованих вище умов підійдуть алгоритми принцип роботи, яких укладено в зниженні розмірності об’єктів зі збереженням збіжності цих об’єктів, і можливістю наступного угруповання, а саме алгоритму Нелінійного Скорочення Розмірності (Nonlinear dimensionality reduction). Це алгоритми навчання. Алгоритм може дізнатися про внутрішню модель даних, яка може використовуватися, щоб відобразити пункти недоступні під час навчання з впровадженням у процес, який часто називають позавибірковим розширенням. До алгоритмів нелінійного скорочення розмірності відносяться наступні:

- зіставлення Семона (Sammon mapping, іноді Проекція Семона);

- картки, що самоорганізуються (Self-organizing map);

- автокодувальники (Autoencoders);

- дифузійні картки;

- численне вирівнювання (Manifold alignment);

- багатомірне шкалювання (MDS) – класичний МДС алгоритм;

- алгоритм Isomap;

- алгоритм LTSA (Local tangent space alignment – локальне вирівнювання до дотичного простору);

- алгоритм LLE (Locally-linear embedding) – Локальне лінійне вкладення;

- локально-лінійне вкладення Гессе (Hessian LLE);

- модифіковане локально-лінійне вкладення;

- нелінійний метод основних компонентів (Nonlinear Principal component analysis [PCA]);

- топологічно обмежене ізометричне вкладення;

- алгоритм t-SNE – стохастичне вкладення сусідів із розподілом Стьюдента.

Для роботи з нашими даними були обрані наступні алгоритми:

**Багатомірне шкалювання** (MDS) — метод аналізу та візуалізації даних за допомогою розташування точок, що відповідають об’єктам, що вивчаються (шкалюються), у просторі меншої розмірності ніж простір ознак об’єктів [8]. Точки розміщуються так, щоб попарні відстані між ними в новому просторі якнайменше відрізнялися від емпірично виміряних відстаней у просторі ознак об’єктів, що вивчаються. Якщо елементи матриці відстаней отримані за інтервальними шкалами, метод багатовимірного шкалювання називається метричним. Коли шкали порядкові, метод багатовимірного шкалювання називається неметричним. Міра відмінностей відстаней у вихідному та новому просторі називається функцією стресу.

Області застосування даного типу алгоритмів:

- пошук прихованих змінних, що пояснюють отриману з досвіду структуру попарних відстаней між явищами, що вивчаються;

- перевірка гіпотез про розташування досліджуваних явищ у просторі прихованих змінних;

- стиснення отриманого дослідним шляхом масиву даних шляхом використання невеликої кількості прихованих змінних;

- наочне представлення даних.

Алгоритм МДС прагне помістити кожен об’єкт у N-мірному просторі таким чином, що між об’єктами відстані зберігаються.

**Алгоритм LTSA** - локальне вирівнювання до дотичного простору (LTSa) є методом навчання різноманітності, яке може ефективно звести нелінійне вкладення з високорозмірних даних в низькорозмірні координати, а також може відновити багатовимірні координати з вкладених координат [9]. Даний алгоритм заснований на інтуїції, таким чином, коли мультирозмірні дані правильно розгорнуті, всі дотичні гіперплощини до різноманіття будуть побудовані. Алгоритм починає обчислення з K-найближчих сусідів кожної

точки. Він обчислює дотичний простір у кожній точці шляхом обчислення d-перших головних компонентів у кожній локальній околиці, оптимізує знаходження вкладення, що узгоджує з дотичними просторами, при цьому ігнорує інформаційні мітки, що переміщуються вибірок даних, і, таким чином, не може бути використаний для класифікації безпосередньо.

**Алгоритм LLE** (Locally-linear embedding) – локальне лінійне вкладення. Локально лінійне вкладення (LLE) шукає нижню межу одновимірної проекції даних, що зберігає відстані не більше найближчих сусідів. Його можна як серію локальних PCA з глобальним порівнянням знаходження найкращого нелінійного вкладення [10].

**Алгоритм t-SNE** (t-distributed stochastic neighbor embedding) – стохастичний алгоритм із розподілом Стьюдента та впровадженням сусіда [11].

Всі перераховані вище алгоритми здійснюють обробку багатовимірних даних з перетворенням їх у двовимірні і поданням на плоскій (двовимірній) діаграмі. Перераховані вище чотири алгоритми виявилися найбільш підходящими для роботи з тією конструкцією даних, що є в даній роботі, ці чотири алгоритми виявилися найбільш продуманими з точки зору результату, і найбільш ефективними для обробки даних. Варто відзначити, що список алгоритмів, що здійснюють нелінійне скорочення розмірності, не обмежується вищевказаним у даній роботі, однак саме перелічені в даній роботі алгоритми найбільш застосовні до існуючих у поточній роботі даних. Також варто відзначити, що багато існуючих алгоритмів є або комбінацією алгоритмів, або різними модифікаціями існуючих.

## 4.5 Основні етапи DataMining

Етап 1. Підготовка даних.

Етап 2. Індексація даних.

Етап 3. Нормалізація даних для використання у алгоритмах обробки.

Етап 4. Виконання алгоритмів та отримання результатів.

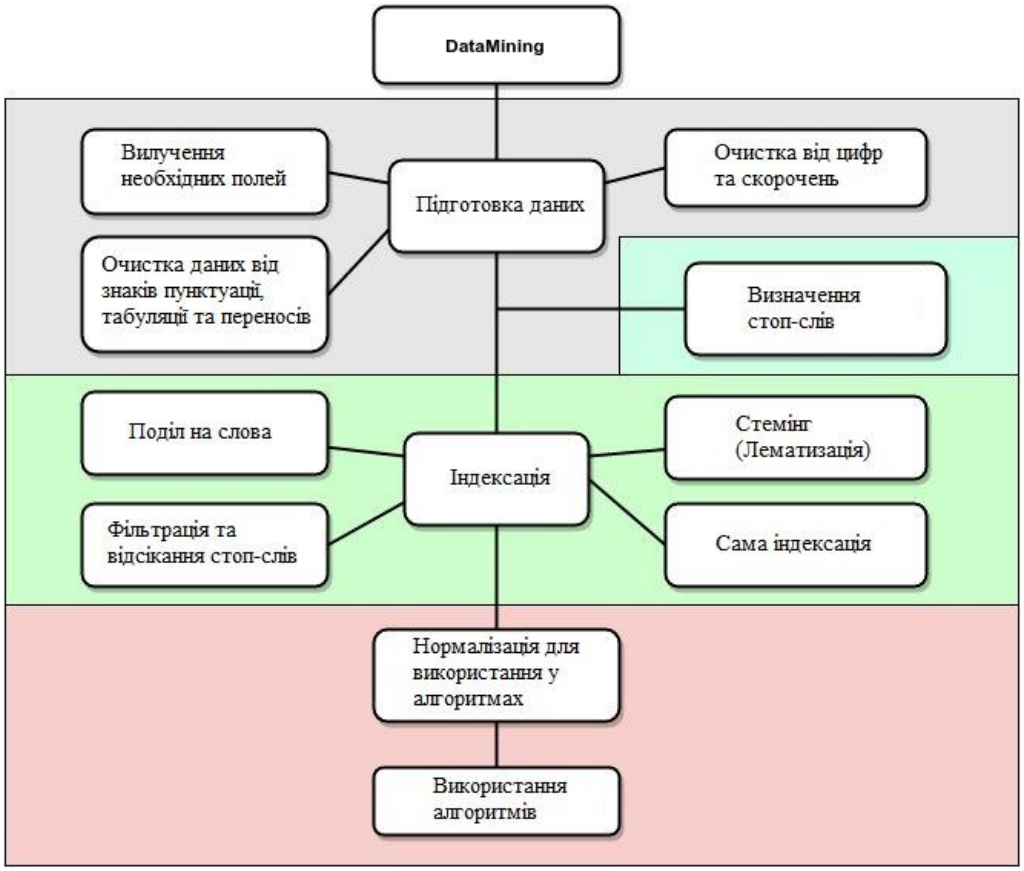


Рисунок 4.3 – Деревоподібна схема послідовності етапів у використанні технології Data Mining

Підготовка даних. На даному етапі мається на увазі наявність даних, отриманих в результаті парсингу сайтів та/або документів, що містять рішення судів. Підготовка даних полягає в очищенні тексту від розділових знаків (.,!?:;), знаків табуляції, пробілів, символів перенесення рядків, усунення скорочень (кг, кв.м., м.куб., грн., ст., год., і так далі) та непотрібних цифр (дати, суми, номери).

Індексацію даних можна умовно розділити на кілька частин: поділ на слова, фільтрація та витяг стопслов, стеммінг або лематизація, індексація.

Поділ на слова - це процедура дроблення рядка тексту (в даній роботі дроблення рядка тексту нормативного акта суду) для отримання масиву

даних в якому як одиничний елемент є слово з тексту, так як усі слова в тексті розділені символом пробілу, відповідно дроблення рядка здійснюється за допомогою символу пробілу. Фільтрування слів і вилучення стоп-слів — етап у якому з масиву слів усуваються слова які мають значення аналізу тексту у майбутньому. Стоп-слова — це слова, які додають відтінок у текст, доповнюють слова, які мають сенсового навантаження окремо і є допоміжними, до них належать і прийменники, частки, союзи.

Стемінг або лематизація слів – це приведення слів до нормальної форми. По суті ці два підходи відрізняються один від одного, але ціль у них та сама. Стемінг являє собою алгоритм відщеплення від слова суфіксів, префіксів і закінчень, у результаті виходить фактично обрізана форма слова, яка зберегла лише основну частину, даний підхід простіше у реалізації, але буває і помилковим, оскільки дроблення то, можливо некоректним [12].

Лематизація слова - це алгоритм метою якого є приведення слова до нормальної форми в однині називному відмінку. У реалізації цей алгоритм складніший, але результат більш придатний для використання у подальшій діяльності. Індексація даних у поточній роботі є лічильник слів у тексті, кожне слово, що зустрілося один раз матиме індекс рівний одному, відповідно скільки разів слово зустрічається в тексті такий індекс воно і матиме.

Для використання даних у різних алгоритмах необхідно дані привести у відповідну форму, просто текст або набір слів не завжди може підійти для виконання операцій, в першу чергу необхідно мати модель обробки даних та розуміння чи підходить набір даних для цієї операції моделі чи ні, якщо ні, треба або відмовитися від застосування даного алгоритму у зв’язку з неможливістю використання, або отримати з існуючого набору даних додаткові дані.

Виконання алгоритму - це власне запуск програми, яка реалізує потрібний алгоритм

4.6 Вибір та перевірка оптимального алгоритму для майбутнього використання

Для використання алгоритмів нелінійного скорочення розмірності була зроблена вибірка частини судових рішень із загальної кількості в наявній базі даних рішень даної роботи, для цього всі рішення були перемішані таким чином, що в кожному алгоритмі фігуруватимуть однакові вхідні дані і при цьому у вибірку потраплять усі типи рішень виходячи з теорії ймовірностей, таким чином була обрана тільки частина рішень, проте ця частина відображатиме картину аналогічну якщо були взяті всі наявні рішення, була зроблена вибірка в кількості 4000 рішень з 93000. При використанні обраних алгоритмів були отримані графічні результати, що значно відрізняються один від одного, проведемо аналіз даних результатів.

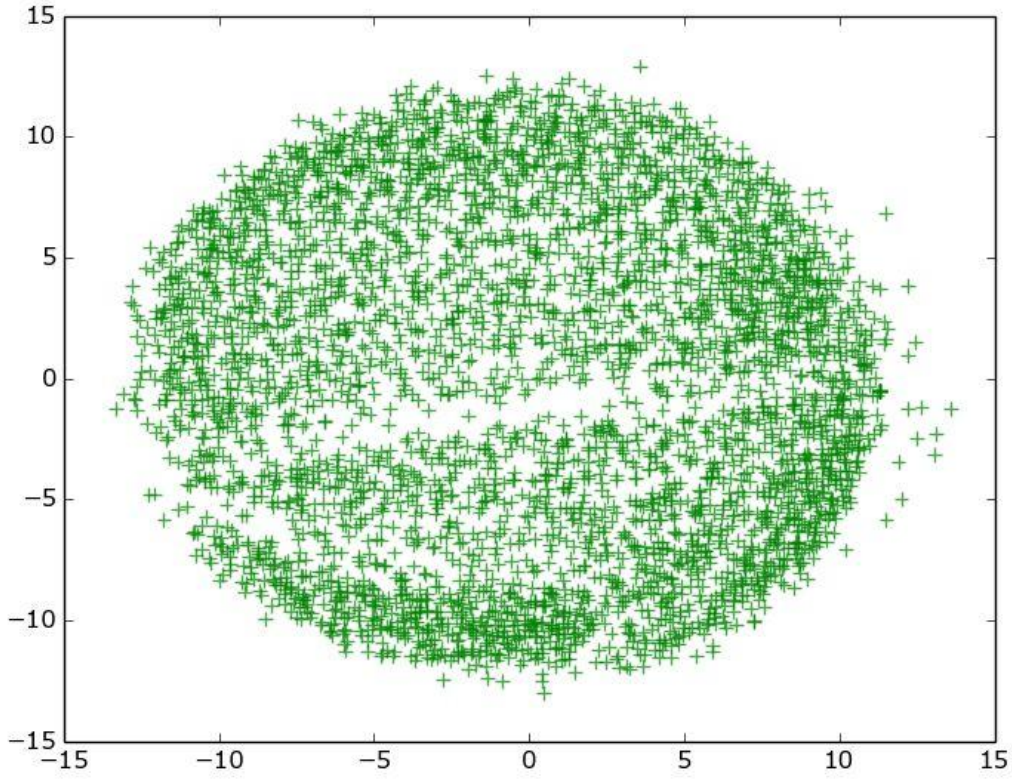


Рисунок 4.4 – Результат виконання алгоритма MDS

На рисунку 4.4 представлено графічне уявлення сукупності судових рішень, отримане в результаті обробки багатовимірних об’єктів судових рішень, перетворені у двовимірні у вигляді точок на площині у двовимірному евклідовому просторі за допомогою алгоритму MDS, координати даних точок є відносними одиницями у просторі, у графічному вигляді дана діаграма дає поки що мало інформації. Необхідно виділити кластери більш явно, для цього в цій роботі звертаємось до алгоритму k-means (метод k-середніх), це метод кластеризації точок розроблений у 50-х роках 20-го століття математиком Гуго Штейнгаузом та майже одночасно Стюартом Ллойдом. Дія алгоритму така, що він прагне мінімізувати сумарне квадратичне відхилення точок кластерів від цих кластерів. Внаслідок обробки сукупності точок на площині методом k-середніх, які були попередньо оброблені алгоритмом MDS, можна бачити на рисунку 4.5.

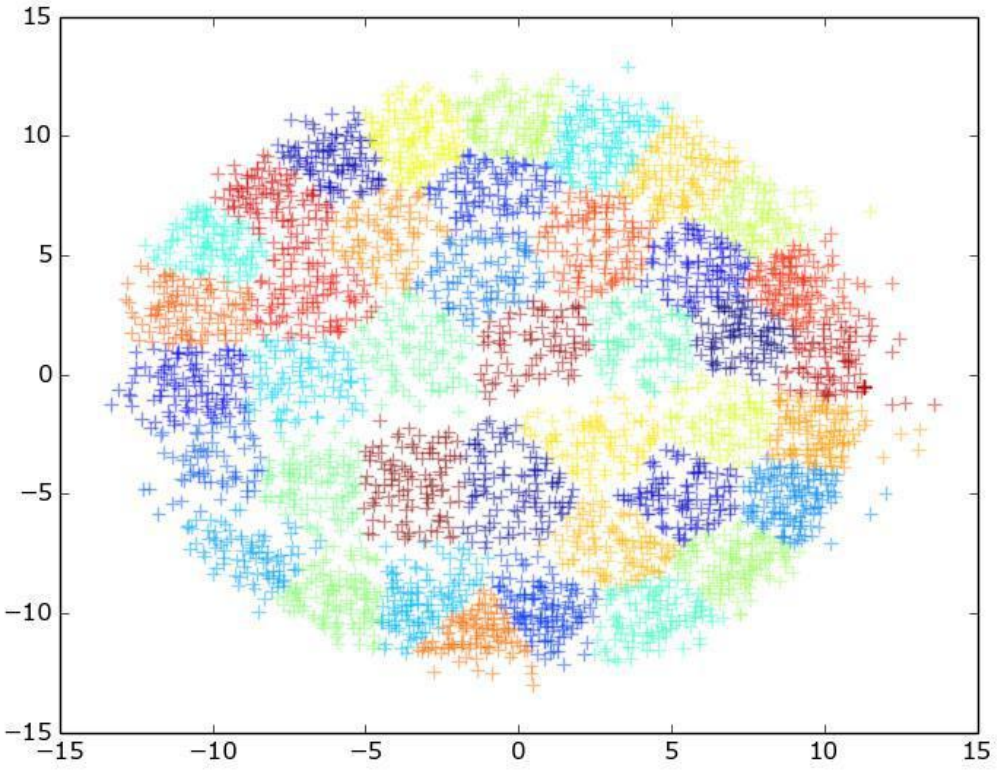


Рисунок 4.5 – Кластеризація судових рішень представлених у вигляді точок на площині, кожна точка представлена у вигляді знака "+"

У даному випадку кластеризація була представлена 40 кластерами, хоча у цій роботі проводилася також кластеризація 100 кластерами. Далі для обробки багатовимірних об’єктів рішень судів був використаний алгоритм LTSA. На рисунку 4.6 представлено графічне уявлення сукупності судових рішень, яке отримано в результаті обробки багатовимірних об’єктів судових рішень цим алгоритмом, перетворені на двовимірні об’єкти, у вигляді точок на площині у двовимірному евклідовому просторі, так само як і в попередньому прикладі, є відносними одиницями у просторі.

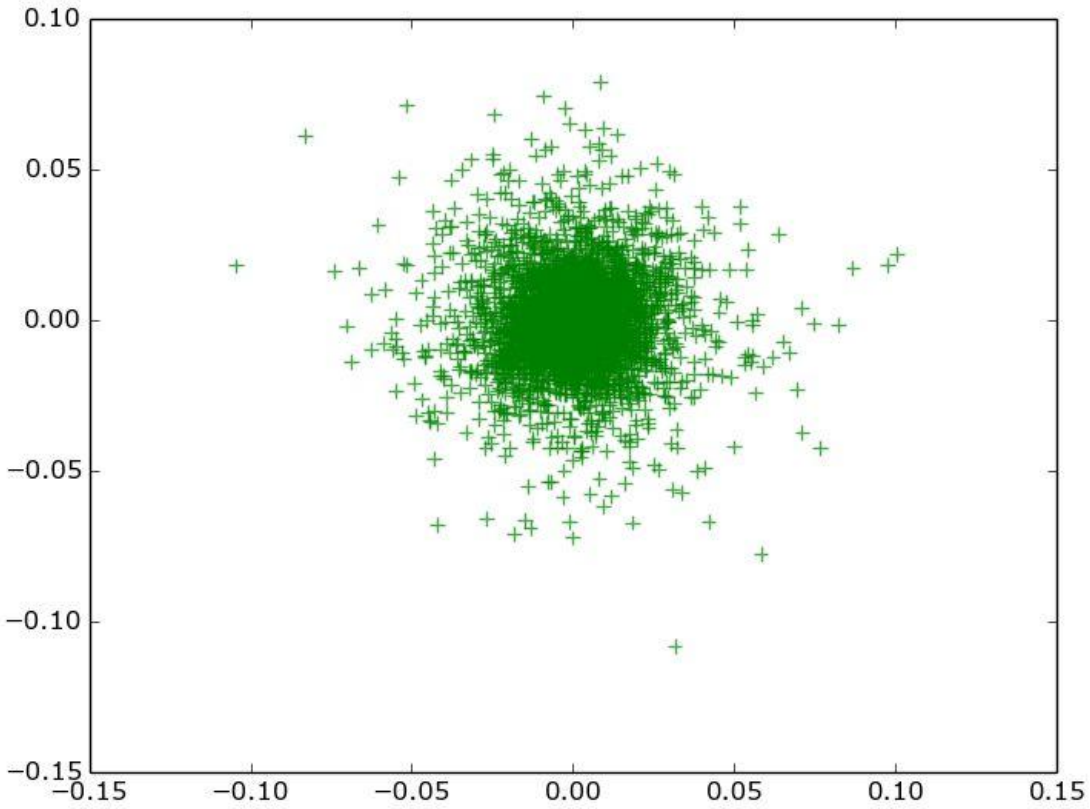


Рисунок 4.6 – Результат виконання алгоритму LTSA

Так само, як і в попередньому алгоритмі, складно побачити явні угруповання об’єктів на даній діаграмі. Тому також, як і для попереднього варіанту необхідно явно провести кластеризацію об’єктів, для чого

скористаємося алгоритмом k-means (метод k-середніх), в даному випадку також візьмемо кількість кластерів рівну 40. Результат обробки сукупності точок на площині методом k-середніх, які були попередньо оброблені алгоритмом LTSA можна бачити на рисунку 4.7.

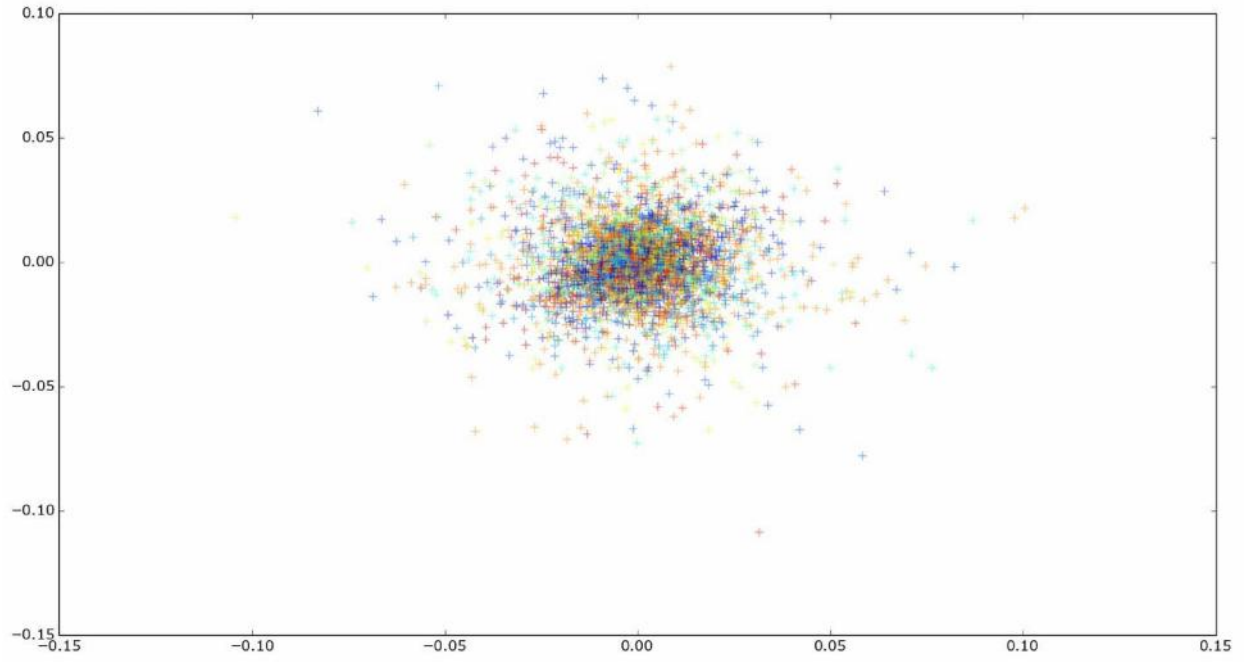


Рисунок 4.7 – Кластеризація судових рішень представлених у вигляді точок на площині, кожна точка представлена у вигляді знака "+"

На рисунку 4.7 дуже складно визначити кластери з судових рішень, тому що візуально можна лише виділяти окремі рішення, але не групи, внаслідок чого візуальні інструменти для об’єктів рішень, оброблених за допомогою алгоритму LTSE, не є в даному випадку показовими. Для аналізу об’єктів судових рішень перейдемо до результатів алгоритму LLE, який дозволяє знижувати розмірність багатовимірних об’єктів, до двомірних об’єктів, на рисунку 4.8 можна побачити графічний результат виконання даного алгоритму.

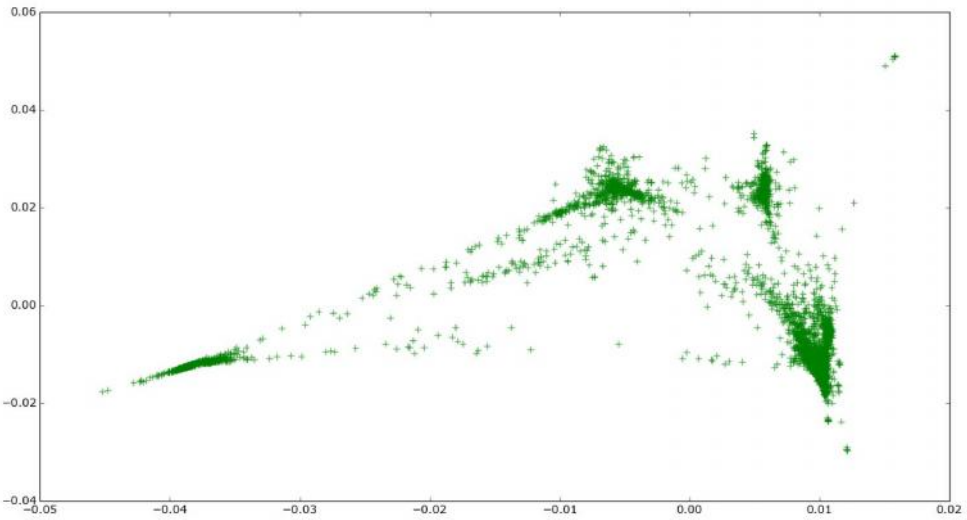


Рисунок 4.8 – Результат виконання алгоритму LLE

Візуально результат виконання алгоритму LLE є більш показовим у плані угруповання об’єктів рішень, можна бачити явні сукупності об’єктів, проте на даній діаграмі також необхідно провести явну кластеризацію з областей різними кольорами для наочності результату. І тут проведемо це за допомогою алгоритму k-means, також розділивши на 40 кластерів. Результат виконання можна побачити на рисунку 4.9.

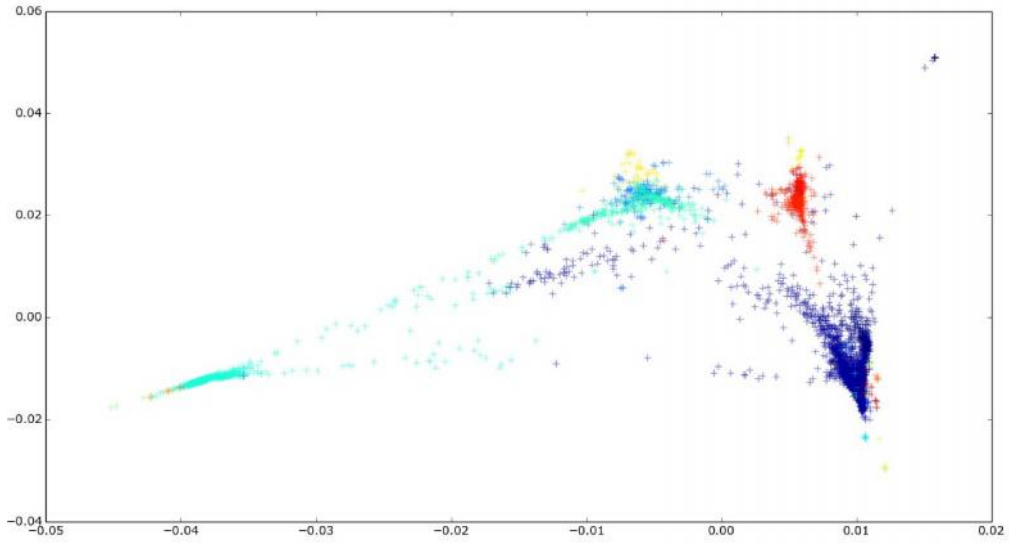


Рисунок 4.9 – Кластеризація судових рішень представлених у вигляді точок на площині, кожна точка представлена у вигляді знака "+"

Кольорова кластеризація на рисунку 4.9 явно відрізняється від кластеризації на рисунку 4.8, хоча деякі області все ж таки залишилися такими ж. Перейдемо до наступного алгоритму та результатів його виконання. Даним алгоритмом є t-SNE, він також знижує розмірність об’єктів із збереженням їхньої збіжності. Результат виконання алгоритму t-SNE можна побачити рисунку 4.10.

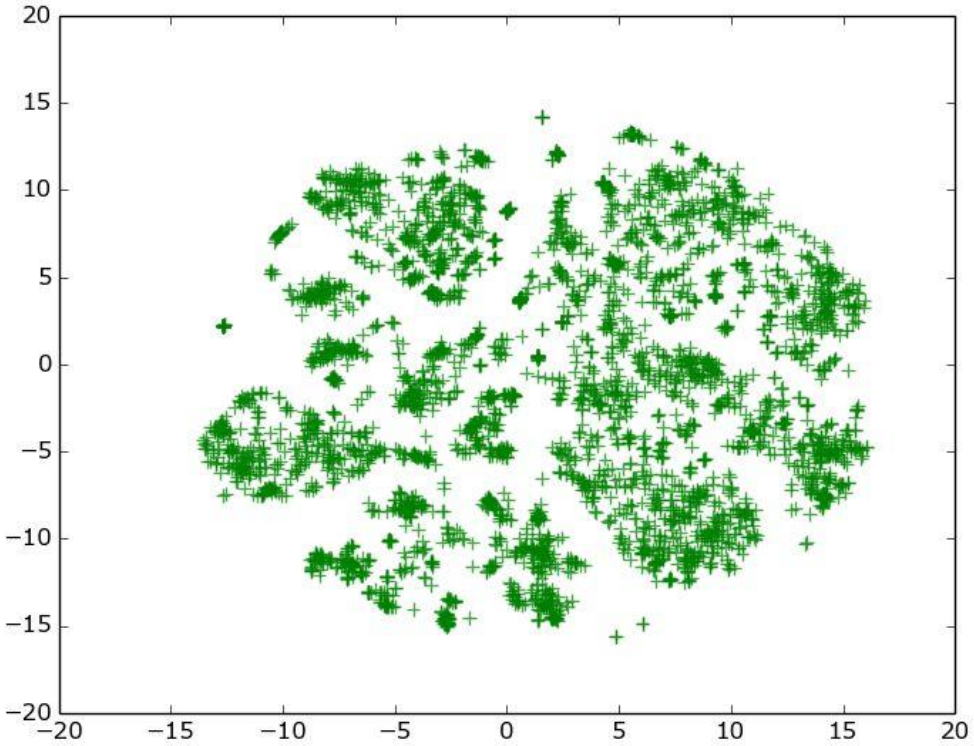


Рисунок 4.10 – Результат виконання алгоритму t-SNE

На даній діаграмі візуально дуже добре виділені групи об’єктів судових рішень, і сукупності об’єктів дуже добре диференційовані, однак і в цьому випадку додатково опрацювати результати виділивши конкретні області та поставивши конкретну кількість класів, для цього використовуємо алгоритм k-means. В результаті обробки отримуємо діаграму, відображену на рисунку 4.11.

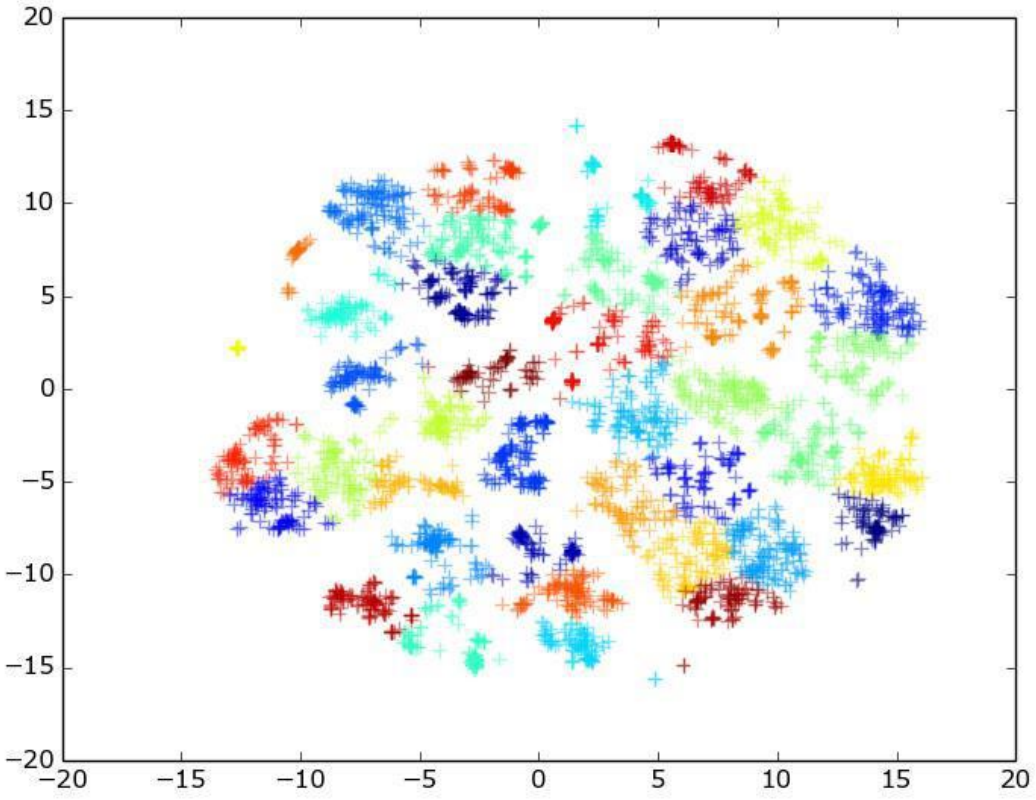


Рисунок 4.11 – Кластеризація судових рішень представлених у вигляді точок на площині, кожна точка представлена у вигляді знака "+"

Результат на рисунок 4.11 дає більш наочну картинку з розподілу рішень та їх збіжності. Також у процесі використання алгоритмів було виявлено дані про продуктивність цих алгоритмів:

- MDS – 4000 рішень, час виконання 25 хвилин;
- LTSA – 4000 рішень, час виконання 19 хвилин;
- LLE – 4000 рішень, час виконання 7 хвилин;
- t-SNE – 4000 рішень, час виконання 3 хвилини.

Ці алгоритми були використані в однакових умовах на машині Lenovo G580, Intel Core i5, RAM 8Gb. На наведених вище діаграмах кожен знак "+" це об’єкт рішення. Кожен об’єкт рішення переданий на вхід алгоритму впливає на своє положення та положення кожного іншого рішення, тобто на

всю картину у цілому. Присутність у вирішенні того чи іншого слова впливає на його належність тому чи іншому кластеру, таким чином кожен елемент має значення у всій системі цілком і для кожного рішення окремо.

## 4.7 Алгоритм t-SNE

Алгоритм - стохастичне вкладення сусідів із розподілом Стьюдента (t-distributed stochastic neighbor embedding) (t-SNE) - це алгоритм машинного навчання зі скороченням розмірності, розроблений Лоренсом ван дер Маатеном і Джеффрі Хінтоном. Це нелінійний метод скорочення розмірності, що особливо добре підходить для вбудовування багатовимірних даних у просторі двох або трьох вимірювань, які можуть бути відображені в діаграмі. Зокрема, цей алгоритм моделює кожен багатовимірний об’єкт у двох або тривимірні точки таким чином, що подібні об’єкти моделюються прилеглими точками і різнорідні об’єкти моделюються віддаленими точками.

Алгоритм t-SNE складається з двох основних етапів. По-перше, t-SNE створює розподіл ймовірностей над парами багатовимірних об’єктів таким чином, що подібні об’єкти мають високу ймовірність бути обраним, у той час як несхожі точки мають нескінченно малу можливість бути обраним. По-друге, t-SNE визначає аналогічний розподіл ймовірностей по точках у низькорозмірній карті, а також зводить до мінімуму відстані Кульбака-Лейблера між розподілами щодо розташування точок на карті. Слід звернути увагу, що в той час як оригінальний алгоритм використовує Евклідову відстань між об’єктами як бази його показників подібності, і це має бути відповідним чином змінено. У алгоритму t-SNE широкий діапазон використання, включаючи наукові дослідження з комп’ютерної безпеки, аналіз музики, дослідження раку та біоінформатика.

4.8 Обробка даних

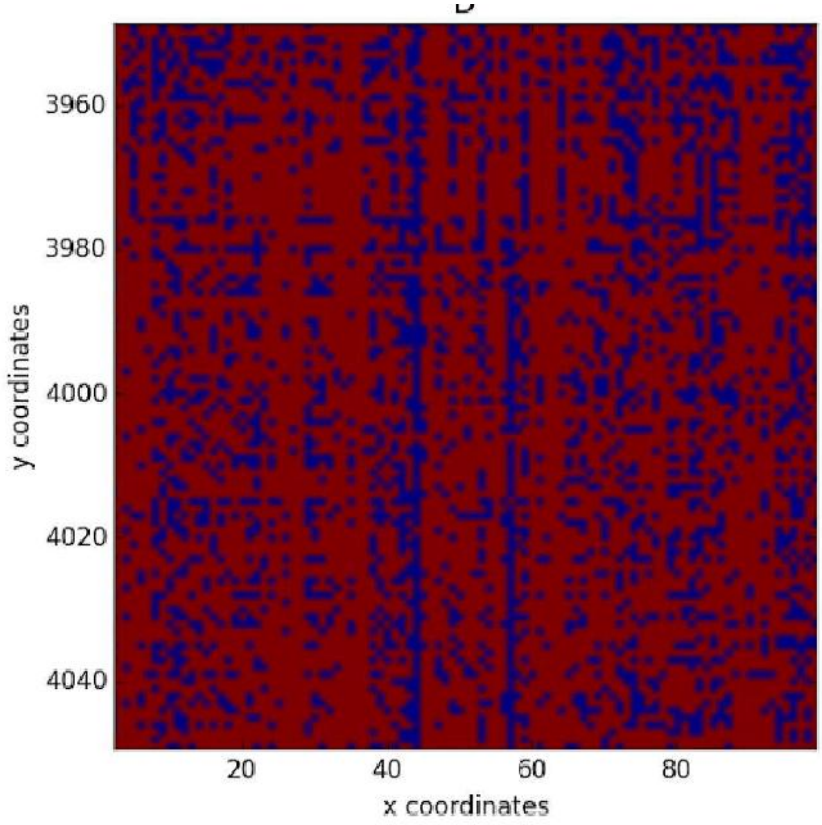


Рисунок 4.12 – Подання вхідних даних у графічному вигляді Python

Як вхідні дані для використання в алгоритмі t-SNE використовуються слова-рішення, заздалегідь підготовлені, відфільтровані за стопсловами, приведені до нормальної форми. На рисунку 4.12 наведено приклад часткового відображення вхідних підготовлених даних кожна синя точка - це наявність слова у рішенні, по осі Y розташовані судові рішення по осі X слова, вже з вхідних даних видно присутність слів, які є в багатьох рішеннях, з дослідження стало ясно що цими словами є специфічні для судової практики слова, наприклад «постановив», «судовий», «стягнути» тощо, але ці слова залишені для чистоти дослідження.

Обробка даних за допомогою алгоритму t-SNE

Як раніше розглядалося в алгоритмі t-SNE використовуються багатовимірні об’єкти для дослідження, як багатовимірний об’єкт

представлені судові рішення і кожною мірою є слово в судовому рішенні приведене до нормальної форми. Щоб представити кожне слово за важливістю проводиться індексація тексту, і по суті позначає кількість разів, яке зустрічається слово в тексті, таким чином ми повністю представили масив слів з кожного рішення у вигляді числових еквівалентів, для того щоб можна було застосовувати математичні моделі. Алгоритм t-SNE знижує багатовимірність нелінійно, і відображає ставлення одного судового рішення до іншого, тобто порівняння судових рішень, спираючись на набір слів, які вони містять. Ще одна особливість даного алгоритму в тому, що він відображає результат на двовимірній площині у графічному вигляді та у разі наявності деяких залежностей і особливо за наявності кластерів розподіляє точки на площині так, що кластери не накладаються один на одного. Результат можна побачити рисунку 4.13.

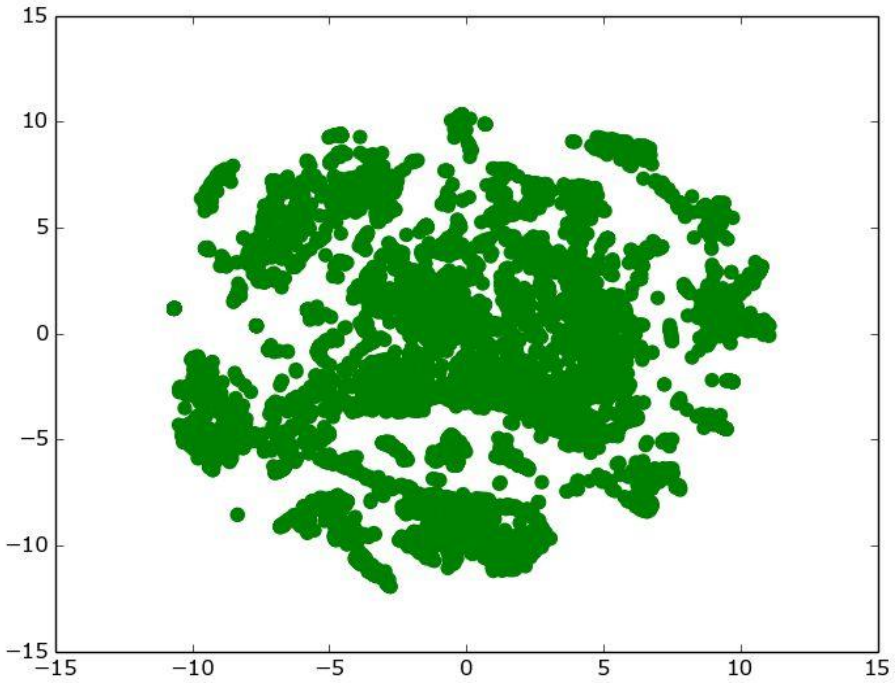


Рисунок 4.13 – Подання результату виконання алгоритму t-SNE у вигляді точкової діаграми

Як видно з рисунка 4.13 точки накладаються одна на одну, і кластеризація хоч і помітна, але виглядає нечітко, для цього змінюється спосіб відображення, кожен елемент виводиться у вигляді знака “+”, а також додається властивість прозорості для кожного елемента, таким чином, кластеризація чітко видно в місцях, де вона є, і ті місця на діаграмі, в яких присутність елементів не значна, ними відповідно не заповнена.

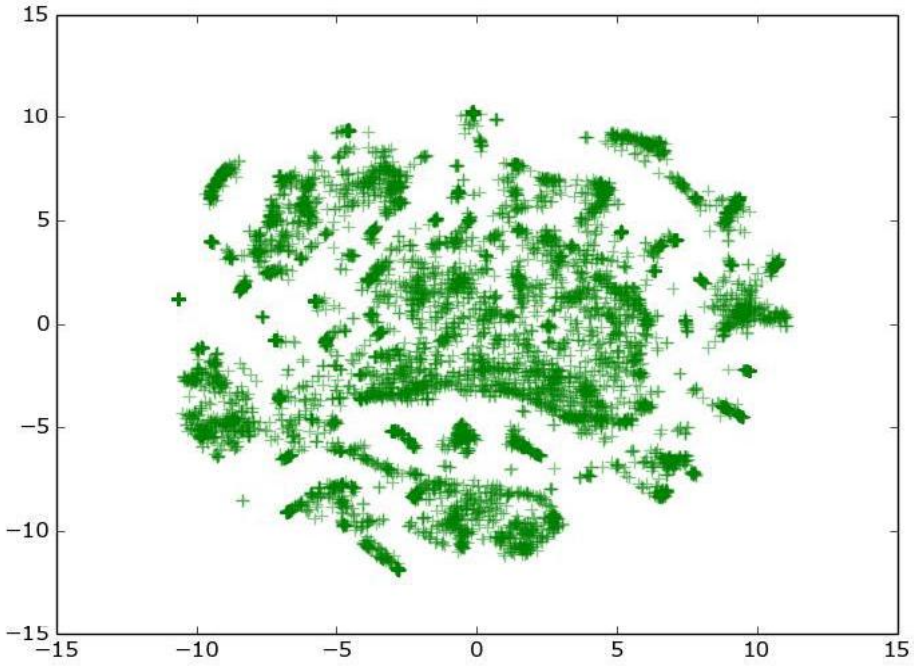


Рисунок 4.14 – Подання результату виконання алгоритму t-SNE у вигляді точкової діаграми з використанням знака "+" та альфа-прозорості

На рисунку 4.14 виразно видно кластери та окремі об’єкти між кластерами. Схожість текстового складу наближає об’єкти один до одного на діаграмі, відповідно відмінності віддаляють, також відбувається угруповання об’єктів за групами слів що і створює кластери на діаграмі.

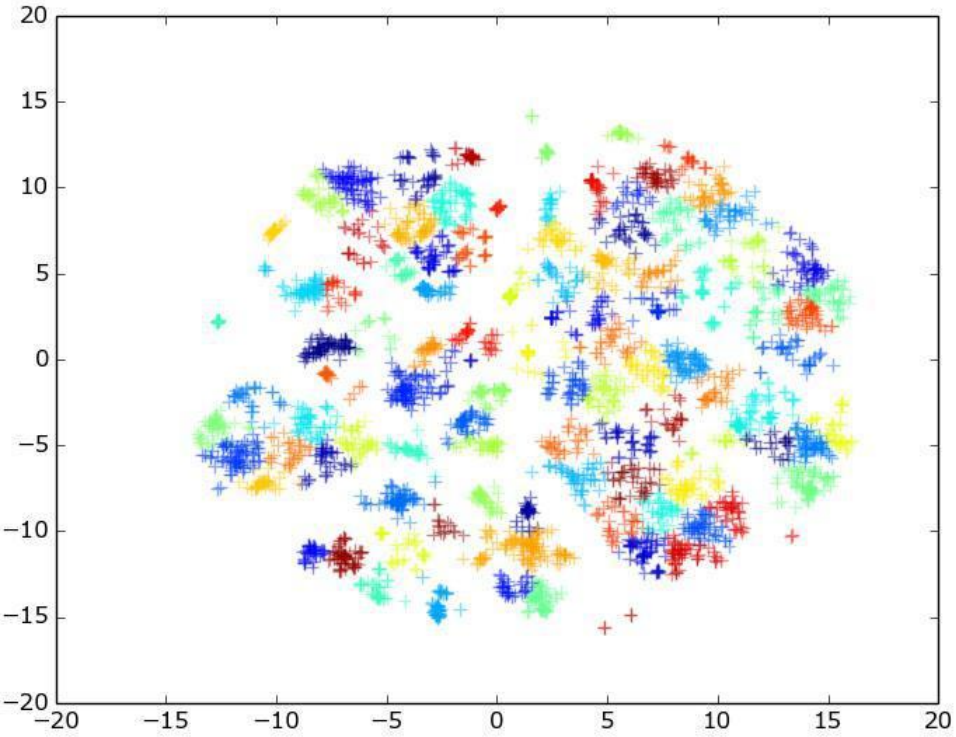


Рисунок 4.15 – Подання результату виконання алгоритму t-SNE у вигляді точкової діаграми з позначеними кластерами

На рисунку 4.15 кластери розмальовані у різні кольори та явно виділяються.

## 4.9 Результат обробки та аналізу даних

В результаті обробки та аналізу судових рішень було виявлено певні факти та закономірності, а також можливості для спрощення роботи з великою кількістю текстових документів.

При вивченні окремих об’єктів із кластерів були виявлені прямі залежності між полями об’єктів-рішень та розташуванням об’єктіврішень на точкових діаграмах та в таблицях для аналізу рішень, залежність проявляється в першу чергу від наступних полів:

- ім’я судді;
- назва суду;

- тип суду;

- норма права;

- тип документа;

- тип суперечки;

- результат справи.

У ході застосування алгоритму t-SNE використали різні способи відображення результатів.

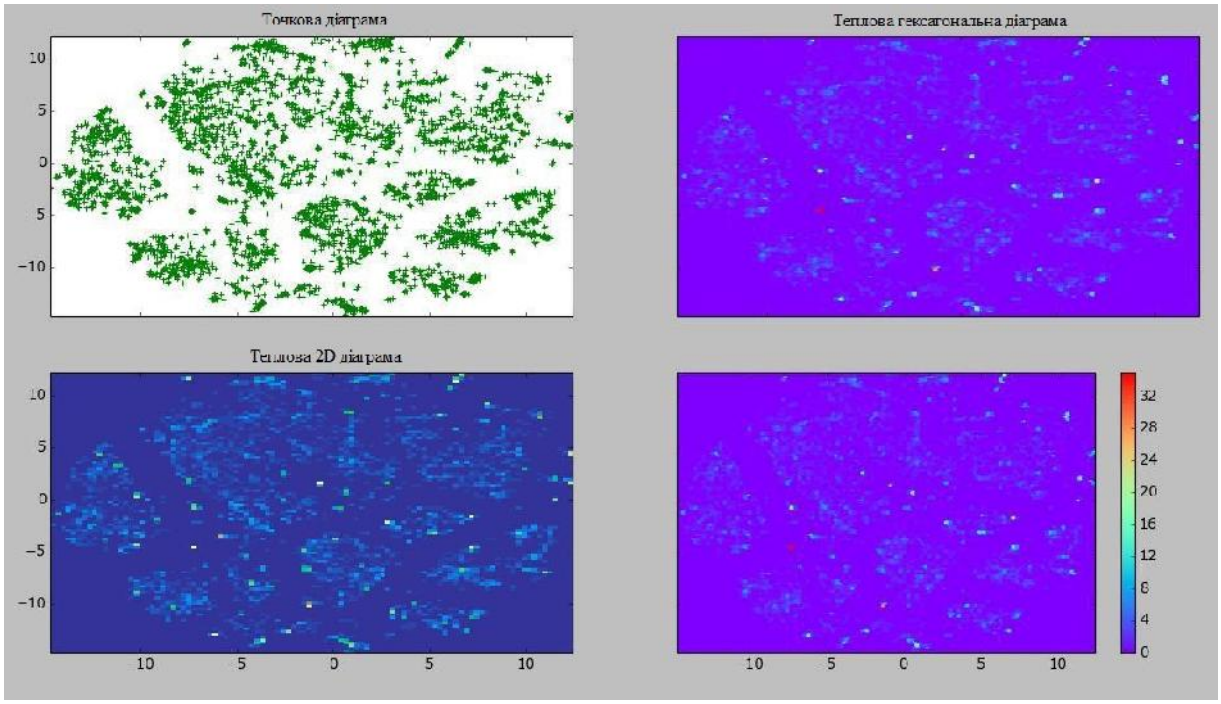


Рисунок 4.16 – Порівняння різних діаграм відображення результату виконання алгоритму t-SNE

Як видно порівняно різних діаграм, саме точкова діаграма дає найбільш наочну картину у взаємозв’язках об’єктів та його кластеризацію. Отриманний результат:

- виявлені основні чинники, що впливають збіжність рішень друг з одним;

- було проведено порівняння алгоритмів, що обробляють багатовимірні об’єкти методами нелінійного зниження розмірності та кластеризації,

визначено найкращий алгоритм для обробки таких багатовимірних об’єктів;

- отримано інструмент, який дає змогу прискорити процес аналізу судових рішень;

- були виявлені деякі відхилення у судових рішеннях, які необхідно вивчати надалі;

- отримано сукупність інструментів для програмної класифікації судових рішень в автоматичному режимі, що дозволяє структурувати раніше неструктуровані дані, також цей спосіб може бути застосований і до інших текстових документів, а не до судових рішень.

# ВИСНОВКИ

У ході виконання роботи було освоєно весь цикл технологій, що використовується в Data Mining. Зокрема, підготовка даних, очищення даних, індексування даних, формування необхідної структури даних успішного застосування алгоритмів Data Mining. Було обрано алгоритми для обробки багатовимірних об’єктів і проведено порівняльний аналіз вибраних алгоритмів. Виявлено явні групи рішень, пов’язані між собою різними атрибутами, причому дані групи формуються повністю автоматично з допомогою технологій і алгоритмів Data Mining. У процесі пошуку неявних залежностей було виявлено деякі факти, які можуть бути для подальшої перевірки рішень, такі як зарахування до груп рішень з кримінальними статтями рішень з адміністративними статтями. Варто врахувати, що кількість судових рішень, що публікуються, зростає і в майбутньому стане можливим провести обробку ще більшої кількості рішень. А також при певній зміні вихідного коду можна досягти значного поліпшення у підготовці та вилученні даних, що підвищить їх точність, як наслідок з одного боку збільшиться кількість рішень, а з іншого зменшить кількість виключених фільтрами судових рішень. Також у процесі використання алгоритмів було виявлено дані про продуктивність цих алгоритмів:

- MDS – 4000 рішень, час виконання 25 хвилин;
- LTSA – 4000 рішень, час виконання 19 хвилин;
- LLE – 4000 рішень, час виконання 7 хвилин;
- t-SNE – 4000 рішень, час виконання 3 хвилини.

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Закон України №3262-IV від 22.12.2005 (ред. від 04.10.2021) «Про доступ по судових рішень».

2. Закон України №2939-VI від 13.01.2011 (ред. від 02.10.2021) «Про доступ до публічної інформації».

3. ZakonOnline [Електронний ресурс]. URL: https://zakononline.com.ua/ (дата звернення 20.11.2021).

4. Pravosud [Електронний ресурс]. URL: https://pravosud.com.ua/ (дата звернення 20.11.2021).

5. Порівняльний аналіз методів визначення нечітких дублікатів для WEB-документів / Ю. Г. Зеленков, І. В. Сегалович // Праці 9-ої Всеросійської наукової конференції «Електронні бібліотеки: перспективні методи та технології, електронні колекції» RCDL 2007: Сбірник робіт учасників конкурсу. — Переславль-Залєський, Росія. — 2007.

6. Двійкові коди з виправленням випадань, вставок та заміщень символів / В. І. Левенштейн // Доповіді Академій наук СРСР – 1965.

7. Регулярні вирази / Фрідл, Дж. — СПб., 2001. — 352 с..

8. «An Introduction to MDS» Wickelmaier, Florian [Електронний ресурс]. URL: https://homepages.unituebingen.de/florian.wickelmaier/pubs/Wickelmaier2003SQRU.pdf (дата звернення: 18.11.2021).

9. Adaptive Neighborhood Graph for LTSA Learning Algorithm without Free-Parameter. Xianlin Zou, Qingsheng Zhu [Електронний ресурс]. URL: http://www.ijcaonline.org/volume19/number4/pxc3873070.pdf (дата звернення: 18.11.2021).

10. Nonlinear Dimensionality Reduction I: Local Linear Embedding. Sam T. Roweis [Електронний ресурс]. URL: http://www.stat.cmu.edu/~cshalizi/350/lectures/14/lecture-14.pdf (дата звернення:

18.11.2021).

11. t-SNE. Laurens van der Maaten. [Електронний ресурс]. URL: https://lvdmaaten.github.io/tsne/ (дата звернення: 17.11.2021).

12. Вплив морфологічного аналізу на якість інформаційного пошуку. М.В. Губин, А.Б. Морозов [Електронний ресурс]. URL: http://rcdl.ru/doc/2006/paper_67_v2.pdf (дата звернення: 17.11.2021).

13. Конспект лекцій з дисципліни «Методи та системи підтримки прийняття рішень» / Марценюк В. П. // 2018.