Національний технічний університет України

«Київський політехнічний інститут»

ім. Ігоря Сікорського

Навчально-науковий комплекс «Інститут прикладного системного аналізу»
(повна назва інституту/факультету)

Кафедра Системного проектування
(повна назва кафедри)

«До захисту допущено»
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)
"___"_____2017 р.

Дипломна робота

на здобуття ступеня бакалавра

з напряму підготовки 6.050101 Комп'ютерні науки
(код і назва)

на тему: "Розробка функціональної моделі Web-сервісної побудови аналітичних моделей класифікації епілептичних нападів"

Виконала: студентка 4 курсу, групи ДА-32
(шифр групи)

Костюк Катерина Євгеніївна _____ _____
(прізвище, ім'я, по батькові) (підпис)

Керівник _____ас., Сергєєв-Горчинський О.О. _____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант економічний к.е.н., доцент Рощина Надія Василівна _____
(назва розділу) (посада, вчене звання, науковий ступінь, прізвище, ініціали) (підпис)

Рецензент _____доцент, к.т.н. Тимощук О.Л. _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Нормоконтроль _____старший викладач Британ О.А._____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2017 року

Національний технічний університет України
«Київський політехнічний інститут»
ім. Ігоря Сікорського

Інститут (факультет) ННК «Інститут прикладного системного аналізу
(повна назва)

Кафедра Системного проектування
(повна назва)

Рівень вищої освіти – перший (бакалаврський)
Напрям підготовки 6.050101 Комп'ютерні науки
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ А.І.Петренко
(підпис) (ініціали, прізвище)
«___»_____20__ р.

ЗАВДАННЯ

на дипломну роботу студенту

Костюк Катерині Євгеніївні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка функціональної моделі Веб-сервісної побудови аналітичних моделей класифікації епілептичних нападів

керівник роботи Сергєєв-Горчиньский, ас.,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «10» травня 2017 р. №1477-с

2. Термін подання студентом роботи 15 червня 2017 р.

3. Вихідні дані до роботи

1) Медичні дані про епілептичні припадки;
2) Документація по використанню систем інтелектуального аналізу даних.

4. Зміст роботи

1) Дослідження предметної області;
2) Дослідження систем ІАД та порівняння функціональності монолітних систем та Веб-СОА
3) Розробка функціональних моделей ;
4) Функціонально-вартісний аналіз програмного продукту;

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) Презентація, 3 плакати(таблиця 3.1, рисунок 3.5, рисунок 3.6).

6. Консультанти розділів роботи*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Н.В., к.е.н., доцент		

7. Дата видачі завдання 10.03.2017

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	1.02.2017	
2	Аналіз предметної області	20.02.2017	
3	Збір інформації	1.03.2017	
4	Аналіз існуючиї методів	10.04.2017	
5	Розробка тестової функціональної моделі	26.04.2017	
6	Функціонально-вартісний аналіз	15.05.2017	
7	Підготовка графічного матеріалу	15.06.2017	
8	Оформелення дипломного проекту	19.06.2017	

Студент _____ К.Є. Костюк
(підпис) (ініціали, прізвище)

Керівник роботи _____ О.О. Сергєєв-Горчинський
(підпис) (ініціали, прізвище)

* Консультантом не може бути зазначено керівника дипломної роботи.

АНОТАЦІЯ

До бакалаврської дипломної роботи Костюк Катерини Євгеніївни

на тему:

“Функціональне моделювання розподіленої Web-сервісної побудови аналітичних моделей класифікації епілептичних нападів”

Основою даної роботи є аналіз даних, що проводиться з метою попередження приступів епілепсії. За результатами якого можливо розробити найбільш ефективне персональне лікування пацієнтів.

Тема роботи актуальна у зв’язку з тим, що в медицині зараз існує необхідність в системі, яка буде попереджати пацієнта про напад, до того як він станеться та допоможе подолати обмеження на життя людей з діагнозом епілепсії.

В роботі показано наявні підходи та прилади до виявлення нападів. Також розглядаються основні алгоритми та детектори виявлення нападів. Проводиться аналіз існуючих систем інтелектуального аналізу даних(Weka, Weka4WS, ClowdFlows).

В результаті виконання практичної частини роботи було розроблено функціональні моделі монолітної та розподіленої Веб-сервісної побудови. Було проведено аналіз даних на основі підготованих наборів датасетів. За результатами даного аналізу була проведена оцінка ефективності лікування пацієнтів різними засобами.

Дипломну роботу виконано у відповідності до норм проектування, її зміст відповідає темі і завданню, а пояснювальна записка — вимогам до оформлення текстових документів.

Ілюстративний матеріал повністю і наочно розкриває основні положення виконаної роботи.

Загальний об’єм: 81 сторінок, 24 рисунків, 23 таблиць, 12 посилань.

Ключові слова: ЕЕГ, ЕКГ, класифікація, модель, аналіз даних, віджет, монолітна система ІАД, Веб-СОА, датасет.

АННОТАЦИЯ

К бакалаврской дипломной работы Костюк Екатерины Евгеньевны

на тему:

"Функциональное моделирование распределенной Web-сервисной построения аналитических моделей классификации эпилептических припадков"

Основой данной работы является анализ данных, проводится с целью предупреждения приступов эпилепсии. По результатам которого возможно разработать наиболее эффективное персональное лечение пациентов.

Тема работы актуальна в связи с тем, что в медицине сейчас существует необходимость в системе, которая будет предупреждать пациента о нападении, прежде чем он произойдет и поможет преодолеть ограничения на жизнь людей с диагнозом эпилепсии.

В работе показано имеющиеся подходы и приборы к выявлению нападений. Также рассматриваются основные алгоритмы и детекторы обнаружения нападений. Проводится анализ существующих систем интеллектуального анализа данных (Weka, Weka4WS, ClowdFlows).

В результате выполнения практической части работы была разработана функциональные модели монолитной и распределенной Веб-сервисной построения. Был проведен анализ данных на основе подготовленных наборов датасета. По результатам данного анализа была проведена оценка эффективности лечения пациентов различными средствами.

Дипломную работу выполнено в соответствии с нормами проектирования, ее содержание соответствует теме и заданию, а пояснительная записка - требованиям к оформлению текстовых документов.

Иллюстративный материал полностью и наглядно раскрывает основные положения выполненной работы.

Общий объем: 81 страниц, 24 рисунков, 23 таблиц, 12 ссылок.

Ключевые слова: ЭЭГ, ЭКГ, классификация, модель, анализ данных, виджет, монолитная система ИАД, Веб-СОА, датасет.

ANOTATION

On Kostiuk Kateryna Evgenyivna bachelor's degree thesis

"Functional modeling of distributed Web-service construction of analytical models classification of epileptic seizures"

The basis of this work is the analysis conducted to prevent attacks of epilepsy. The results which may develop the most effective personalized treatment of patients.

This work is relevant due to the fact that in medicine there is now a need for a system that will alert the patient about the attack before it happens and help overcome the limitations on the life of people diagnosed with epilepsy.

The paper shows the existing approaches and instruments to detect attacks. Also considered basic algorithms and detectors detect attacks. The analysis of existing data mining (Weka, Weka4WS, ClowdFlows).

As a result of the practical part of functional models were developed and distributed a monolithic Web service construction. It analyzes the data based on the prepared sets datasetiv. The results of this analysis was an assessment of the effectiveness of treating patients by various means.

Thesis made in accordance with design standards, its content and relevant to the task and explanatory note - Requirements for text documents.

Illustrative material completely and clearly reveals the basic provisions of the work done.

Total volume: 81 pages, 24 figures, 23 tables, 12 references.

Keywords: EEG, ECG, classification, model analysis, widget, monolithic system of IAD, SOA Web, dataset.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

ЕЕГ — електроенцефалограма.

ЕКГ — електрокардіографія.

ІАД — інтелектуальний аналіз даних.

GUI (Graphical User Interface)— графічний користувацький інтерфейс.

NAT (Network Address Translation) — перетворення мережних адрес.

СОА — сервісно орієнтована архітектура.

SVM (Support Vector Machine) — метод опорних векторів.

ВСТУП

Основою даної роботи є аналіз даних, що проводиться з метою попередження приступів епілепсії. Результати даних досліджень також можуть бути використані для розробки найбільш ефективного персонального лікування.

Епілепсія − це хронічне захворювання центральної нервової системи людей, що призводить до періодичних нападів, що відбувається раптово внаслідок тимчасового відхилення від норми електричної активності мозку та дає руйнівні симптоми. Серед цих симптомів тимчасова втрата сприйняття (зору), галюцинації або конвульсії всього тіла.

Вилікуватися від епілепсії неможливо, але за умови відповідного лікування епілептичні напади можна стримати у 70% випадків. Окрім медикаментозного лікування, існує можливість проведення оперативного втручання, проведення нейростимуляції. Багатьом хворим вдається відмовитися від лікувальних препаратів без поновлення нападів.

В усьому світі налічується близько п'ятдесяти мільйонів людей з діагнозом епілепсії, з них у 1,2 мільйона непередбачувані напади зберігаються, не зважаючи на лікування одним або декількома проти епілептичними препаратами. В медицині такі напади названі нерозв'язними судомами, які дуже сильно обмежують незалежність і мобільність (активний спосіб життя) людини і, як наслідок, можуть призвести до соціальної ізоляції і економічних труднощів, а також збільшити шанси людини на отримання опіків, порізів, переломів черепа, і навіть раптову смерть.

Епілепсія може бути успадкованою, або може розвинутись в результаті черепно-мозкової травми, такої як серйозний удар по голові, інсульт, церебральна інфекція або злоякісна пухлина мозку, але це не просто хвороба, а сукупність синдромів, які сприяють повторним нападам.

Негативний вплив неконтрольованих судом впливає не тільки на особу з епілепсією, а також на членів сім'ї, які піклуються про неї та суспільство, яке

зазнає щорічні збитки в розмірі 12,5 мільярда доларів на охорону здоров'я і компенсацію втрати працездатності. Отже, існує потреба в нових методах лікування, які краще б передбачали епілептичні напади, а також технології, яка б допомогла забезпечити охорону та зменшити наслідки судом для осіб з діагнозом епілепсії.

Неможливо заперечити актуальність даної теми, тому існує необхідність в медичній системі, яка буде попереджати пацієнта про напад, до того як він станеться та допоможе подолати обмеження на життя людей з діагнозом епілепсії. Цього можна досягти завдяки комп'ютеризованому виявленню початку приступу, та не тільки попереджати про напад, а також нейрофізично стимулювати ділянки головного мозку, що зупинятиме прогресування нападу та зменшить кількість хімічних препаратів, які були потрібні раніше.

1 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз наявних підходів (алгоритмів, приладів) до вирішення задачі, що розглядається

1.1.1 Виявлення нападів

Початок нападу та виявлення причин проводиться за допомогою аналізу електроенцефалограми (ЕЕГ), на якій зображається активність головного мозку. Існують ЕЕГ скальпа, при якій електроди розміщують поверх шкіри, та внутрішньочерепна ЕЕГ, з назви зрозуміло, що електроди вбудовуються в череп пацієнта та розміщуються на поверхні головного мозку. Проблемою є мінливість ЕЕГ у різних людей, оскільки нормальна активність мозку в одного пацієнта, може бути аномальною для іншого, а тобто перед епілептичним станом, саме тому потрібно розглядати та аналізувати дані кожного пацієнта, після чого можна робити висновки .

Деколи поява нападів, може бути виявлена лише аналізом імпульсів, які відбуваються глибоко всередині мозку. При невчасному виявленні, може відбутися прогресування нападу, яке супроводжуватиметься фізичними ускладненнями, такими як м’язові скорочення.

Варто зауважити, що при детектуванні нападів поверх шкіри голови складніше виявляти напади, оскільки реєструється тільки активність нейронів поверхні мозку, а змінний струм може вплинути на детектор такого роду та знизити його точність.

1.1.2 Алгоритми та детектори виявлення нападів

Детектори нападів класифікують як:

- *детектори початку нападу*, метою яких є визначення нападів якомога скоріше, але точність або якість не є максимальною.

В області діагностики, виявлення початку нападу може бути використано для локалізації церебрального походження нападу та для запуску нейростимуляторів, призначених для впливу на прогресування нападу. В такому випадку, найважливіше якомога скоріше визначити відхилення від звичного стану пацієнта, щоб почати стимуляцію та використати швидкодійні протисудомні препарати якомога скоріше після початку нападу.

- *детектори нападу*, які виявляють напади з максимальною точністю, але не в найкоротший термін.

Комп'ютеризовані детектори нападів можуть дозволити лікарям назначиити кращу терапію, яка в наш час, застосована на розрахунку індивідуальної кількості і тяжкості нападів, які пацієнти відчували в періодах між візитами до клініки. На жаль, занадто багато пацієнтів не точно сповіщають лікаря, що може привести до неправильного лікування, через що хімічних препаратів може бути назначено більше ніж потрібно, що призведе до збільшення ймовірності отримання побічних ефектів, якщо ж їх замало, то тривалість судом буде збільшуватися. Детектор нападів дає лікарю потрібні дані, при зіставленні яких він зможе розробити найбільш ефективне лікування.

1.1.3 Бінарна класифікація

Бінарна класифікація - це процес присвоєння спостереження до одного з двох класів або категорій таким чином, щоб оптимізувати обрану мету продуктивності; наприклад, звести до мінімуму ймовірність помилкової класифікації. В проблемі виявлення початку судом спостереженням є часовий інтервал ЕЕГА і два класи:

- судомної активності (клас C1)
- не судомної активності (клас C2).

Визначення класу до якого належить спостереження містить два етапи. По-перше, характерні властивості або функції, які є найбільш дискримінаційними між екземплярами кожного класу зібрані в вектор ознак, що імпортуються зі

спостереження. Далі, класифікатор, який навчений розпізнавати різницю між векторами ознак, витягнутих з примірників кожного класу визначає до якого класу належить спостереження.

Успішність процесу бінарної класифікації залежить від того, які функції будуть вилучені зі спостережень, а які використовуватимуться класифікатором для визначення належності до класу. Особливості, які мають розподіл значень для спостережень, що належать до класу C1 та до класу C2, дуже різняться між собою, завдяки чому отримуємо високу продуктивність. Класифікатор здатний виробити просте правило з навчальних даних, яке точно розрізняє вектори ознак, які були отримані з екземплярів кожного класу.

1.1.4 Практичне застосування детекторів

Блок-схема зображена на рис. 1.1 показує етапи обробки детектора у конкретного пацієнта. Детектор проходить L-секундний період від кожного з N каналів ЕЕГ через набір фільтрів. У свою чергу, набір фільтрів визначає ознаки для кожного каналу M, які відповідають значенню енергій в межах смуг частот M. M характеристика імпортується з кожного з N каналів, які формують MxN елемент вектора X_T, який автоматично захоплює спектральні та просторові співвідношення між каналами. Далі, W вектори ознак X_T, X_{T-L}, ..., і $X_T - (W - 1)L$ об'єднуються, для формування вектора X_T, який містить WxMxN елементів та автоматично фіксує еволюцію в часі спектральних і просторових відносин між вхідними каналами ЕЕГ.

Нарешті, вектору функцій X_T відповідає клас судомної або не судомної активності. Детектор повідомляє про напад після того, як одному вектору характеристик X_T відповідає клас судомної активності.

Оскільки для різних пацієнтів значення варіюються, опорно-векторна машина проходить навчання в режимі тренування, щоб зробити висновки, щодо стану окремого пацієнта та визначити, належність векторів до класу судомної активності, чи не судомної активності.

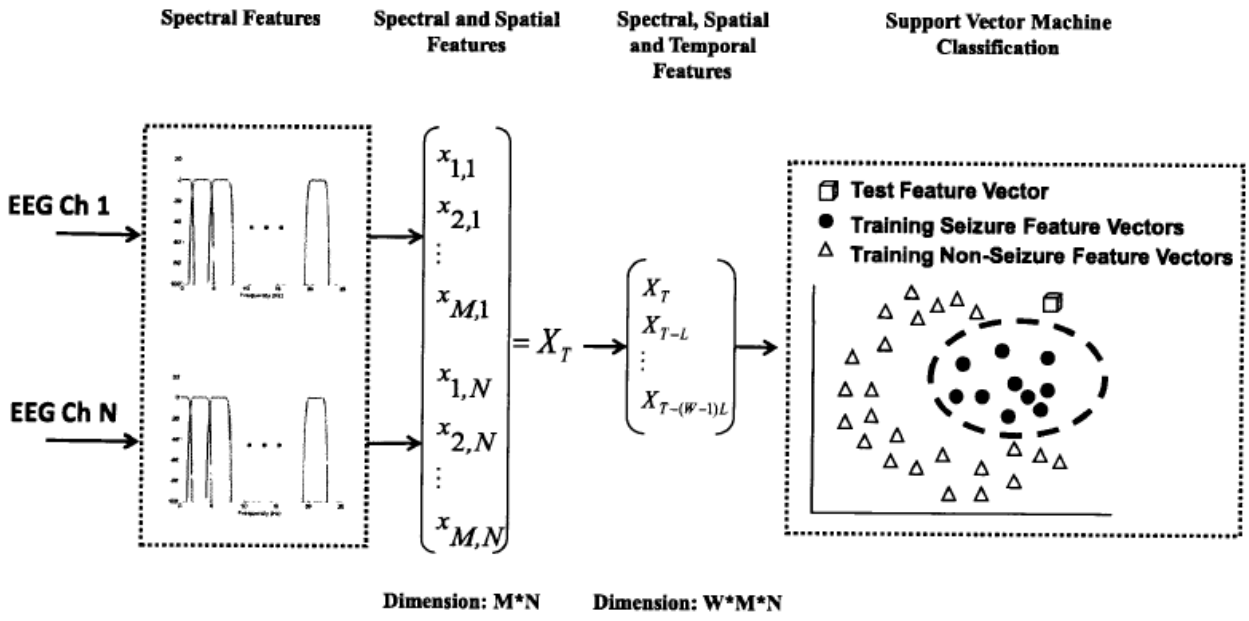


Рисунок 1.1 – Архітектура детектора нападу конкретного пацієнта [1]

Навчання векторів не судомної активності імпортують з H годин безперервного запису скальпа ЕЕГ. Вектори судомної активності є похідними від перших S секунд після початку K навчальних судом.

Рис. 1.2 ілюструє, як послідовні навчальні вектори, не судомного стану отримують дані з тренувальних записів не судомного стану, коли L = 2 і W = 3. Для цих значень параметрів, шести секундне вікно зсувається, на одну секунду, по вектору записів. Для кожного розташування шести секундного вікна генерується вектор з навчальними записами не судомного стану. Більш конкретно, на рис. 1.2, часовий інтервал 0-6 секунд розбивається на три частини, по 2 секундні проміжки часу, які використовуються для генерації векторів ознак X2, X4 і X6. Ці вектори потім групуються для формування першого навчального вектора для визначення не судомного стану X6. Далі, часовий інтервал 1-7 секунди розбивається на три частини, які використовуються для генерації векторів ознак X3, X5 і X7. Ці вектори потім групуються для формування другого навчального вектора A7. Третє тренування, вектора X формується шляхом угрупування векторів ознак X4, X6 і X8, з часового інтервалу 2-8 секунд.

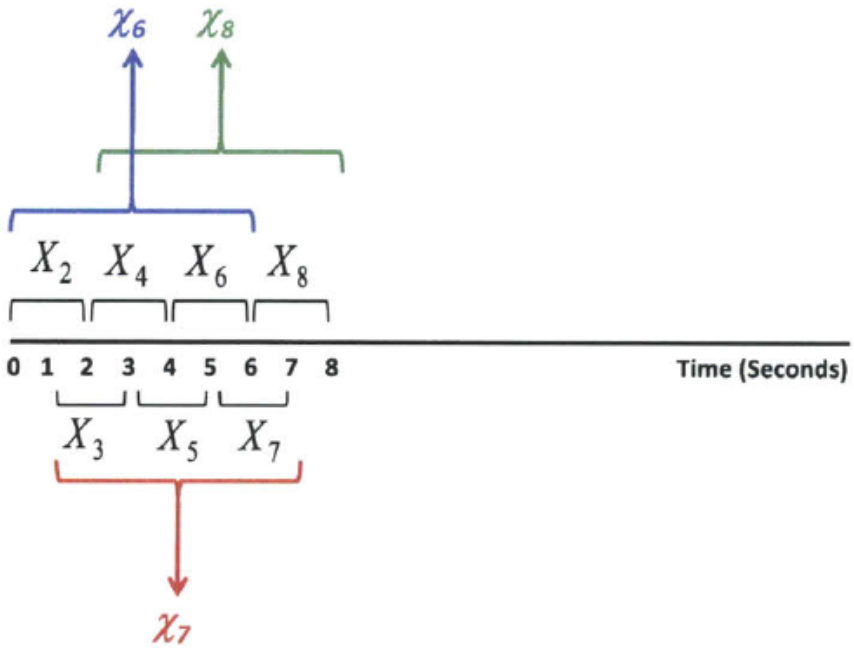


Рисунок 1.2 – Генерація навчальних векторів не судомного стану [1]

Процес, який використовується для створення навчання векторів нападів аналогічний тому, який використовується для створення навчальних векторів, що визначають стан не нападу.

Рис. 1.3 ілюструє, як навчальні вектори нападів отримуються з тренувального нападу, коли L = 2 і W = 3. На рис. 1.3, початок нападу знаходиться в момент часу 0 секунд. Перший навчальний вектор нападу X2 формується шляхом угруповання векторів ознак, отриманих з шести секундн965ого вікна в інтервалі часу від -4 до +2 секунд. Другий навчальний вектор нападу A3 формується шляхом угруповання векторів ознак, отриманих з шести секундного вікна в інтервалі часу від -3 до +3 секунд. Цей процес повторюється до тих пір, поки один з векторів не просунеться на S секунд стану нападу.

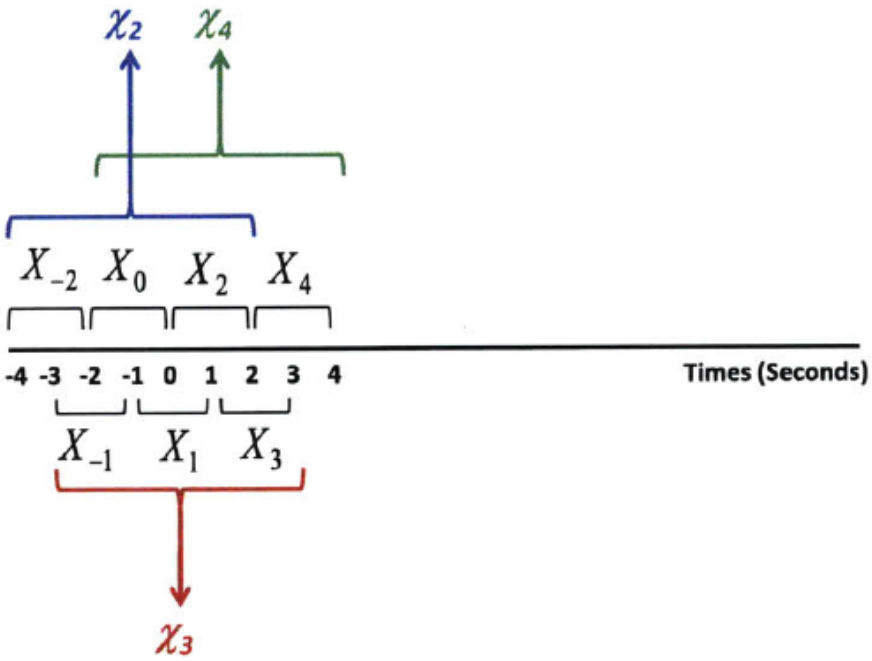


Рисунок 1.3 – Генерація навчальних векторів судомного стану [1]

1.1.5 Приклад персоналного виявлення нападів на основі ЕЕГ-ЕКГ

Існує можливіст виявити напад використовуючи дані з ЕЕГ і ЕКГ сигналів. Детектор виявлення початку нападу пропускає L-секундний період від N = 16 каналів ЕЕГ і подвійний канал, ЕКГ за допомогою функції екстрактора. Функція екстрактор використовує набір з M = 8 фільтрів для обчислення для кожного каналу ЕЕГА енергії з восьми смуг частот, які перекриваються; смуги частот, кожна по 3 Гц в ширину і тривалістю 0-25Hz. 8 функцій, виділені з кожного з 16 каналів потім об'єднуються, щоб сформувати 128 мірний вектор ознак X_T, який автоматично захоплює спектральні і просторові кореляції між каналами (рис. 1.4).

Функція екстрактор використовує канал ЕКГ для вилучення значень вектора $X_{\text{Т,ЕКГ}}$, яка складається з двох функцій. Перша особливість, $X_{1,\text{ЕКГ}}$, відповідає середній частоті серцевих скорочень в межах L-секундного періоду. Друга особливість, $X_{2,\text{ЕКГ}}$, відповідає різниці між виміряною на початку і в кінці L - секундного періоду миттєвої частоти серцевих скорочень. При значенні $X_{2,\text{ЕКГ}}$= 0, отримуємо результат не судомної активності, оскільки послідовні

миттєві вимірювання частоти серцевих скорочень в межах L-секундного періоду будуть коливатися в межах середньої частоти серцевих скорочень. Для пацієнтів, у яких трапляється напад $X_{2,\text{ЕКГ}} > 0$, так як миттєва частота серцевих скорочень в кінці періоду перевищуватиме частоту на початку періоду.

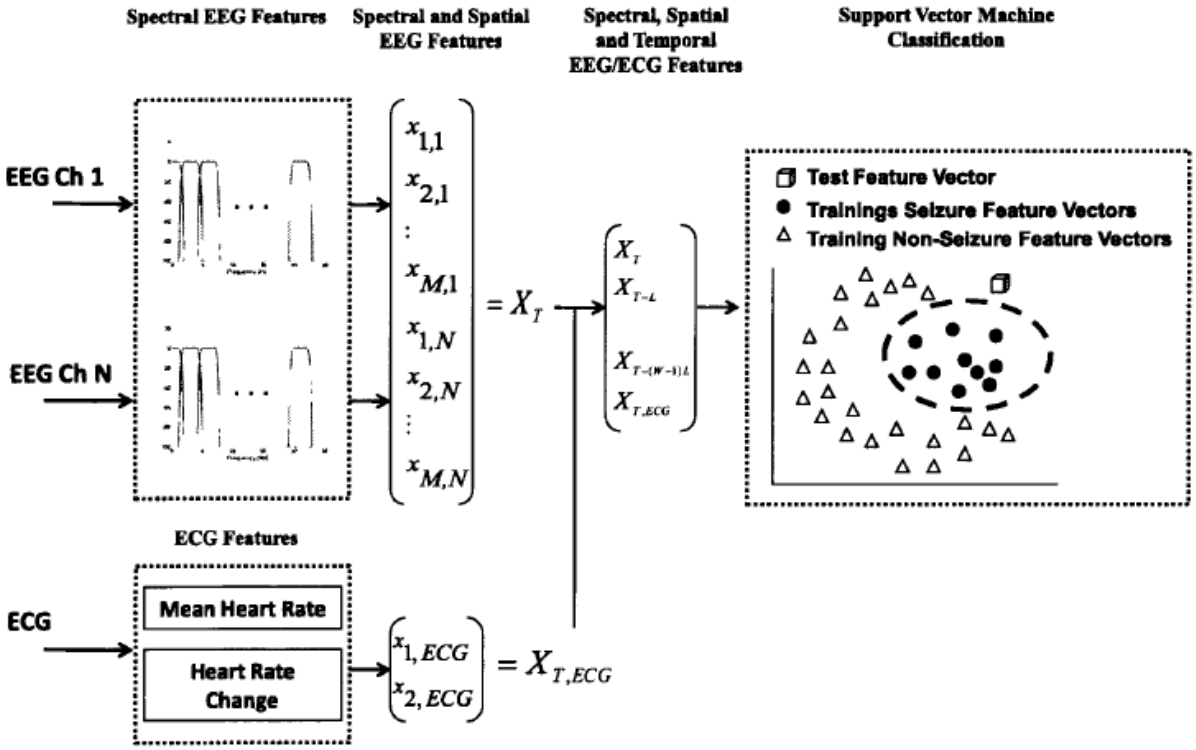


Рисунок 1.4 – Блок-схема детектора початку нападу конкретного пацієнта [1]

Далі, W = 3 ЕЕГ ознаки векторів X_T, X_{T-L}, і X_{T-2L}, а також ЕКГ ознаки вектора X_T, ЕКГ об'єднуються для формування єдиного вектора ознак X_τ. Цей вектор ознак кодується еволюцією в часі спектральних і просторових властивостей шкіри голови ЕЕГ, а також інформацією серцевого ритму і зміною частоти серцевих скорочень, витягнутих з ЕКГ. Це кодування автоматично генерується без втручання користувача.

Нарешті, вектор значень X_T отримує значення судомного або не судомного стану пацієнта [1].

1.2 Висновки до розділу

Тип детектору може бути обраний згідно задачі, яку треба розв’язати. Наприклад на початку нападу, важлива швидкість виявлення, завдяки чому можна застосовувати нейростимуляцію, щоб попередити прогресування судом, у той час як антиепілептична терапія та дослідження для конкретної особи вимагають високої точності незалежно від терміну їх проведення.

2 НАЯВНІ СИСТЕМИ ДЛЯ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ

2.1 Монолітні системи ІАД

У даному підпункті представлено порівняльний огляд пайпопулярніших систем для аналізу даних – Weka, Knime, RapidMiner.

Weka - це ціла колекція інструментів і алгоритмів для аналізу даних і прогнозування. Серед плюсів інструменту:

- зручний інтерфейс (наприклад, текстовий рядок для введення команд);

перетворення даних (в тому числі попередня обробка сирих даних);

- підтримка безлічі алгоритмів машинного навчання і можливість їх швидкого застосування;
- зручний виведення результатів роботи алгоритму (легко порівнювати точність різних моделей);
- вибір ознак;
- візуалізація даних;
- можливість проведення експериментів (причому можна запускати відразу декілька алгоритмів на різних завданнях і отримати загальний звіт);
- можливість подання всього процесу рішення задачі в формі графа.

RapidMiner і Knime – широко відомі інструменти схожі і за формою, і за змістом (хоча перший, на відміну від другого, існує на повністю безкоштовній основі) - тому вони об'єднані одну категорію. Обидва інструменти підтримують безліч стандартних завдань - що стосуються перетворення даних, статистики, машинного навчання та візуалізації. Весь процес аналізу даних представлений у вигляді інтерактивного графа - послідовності операторів, при цьому користувачеві доступні оператори Weka.

2.1.1 Порівняльні характеристики монолітних систем ІАД

В даному підрозділі представлено табл. 2.1-2.6 з результатами оцінки п'яти інструментів (Knime і RapidMiner з вищезгаданих причин об'єднані в одну категорію). За підсумками підрахуваних сумарних оцінок кожної системи у відношенні до конкретної характеристики створена підсумкова табл. 2.7.

Таблиця 2.1 – Обробка даних

	Weka	Knime / RapidMiner
Практичні навички	3	2
Чи можливо робити складні перетворення	2	2
Прості перетворення	3	3
Сума	8	7

Таблиця 2.2 –Візуалізація

	Weka	Knime / RapidMiner
Гнучкість	2	2
Естетика	3	2
Сума	5	4

Таблиця 2.3 – Машинне навчання

	Weka	Knime / RapidMiner
Машинне навчання	3	2
Естетика	2	2
Сума	5	4

Таблиця 2.4 – Представлення результатів роботи

	Weka	Knime / RapidMiner
Гнучкість	2	3
Витрата часу на налаштування чіткого виведення (менше 3)	3	2
Сума	5	5

Таблиця 2.5 – Швидкість отримання попередніх результатів

	Weka	Knime / RapidMiner
Гнучкість	3	3
Витрата часу на наналаштування чіткого виведення	3	3
Сума	6	6

Таблиця 2.6 – Наочність процесу аналізу даних

	Weka	Knime / RapidMiner
Наочність	2	3
Сума	2	3

У заключній таблиці 2.7 - підсумки аналізу. У кожній з шести номінацій було обрано умовного «лідера» або «лідерів» - тобто ті програмні продукти, які найефективніше вирішують конкретні завдання.

Таблиця 2.7 – Підсумки

	Weka	Knime / RapidMiner
Обробка даних	8	7
Візуалізація	5	4
Машинне навчання	5	4
Представлення результатів роботи	5	5
Швидкість отримання попередніх результатів	6	6
Наочність процесу аналізу даних	2	3
Сума	31	29

За результатами аналізу очевидно, що Weka має більше переваг, ніж Knime, Rapid Miner [10].

2.1.2Weka

Weka (Waikato Environment for Knowledge Analysis) - це безкоштовне програмне забезпечення для аналізу даних, написане на Java в університеті Уайкато (Нова Зеландія), що розповсюджується за ліцензією GNU GPL.

Weka дозволяє виконувати такі завдання аналізу даних, як:

- підготовка даних - попередня обробка - preprocessing,
- відбір ознак - feature selection,
- кластеризація,
- класифікація, зокрема, дерева рішень,
- пошук асоціативних правил,
- регресійний аналіз,
- візуалізація результатів.

Weka - набір засобів візуалізації та бібліотека алгоритмів машинного навчання для вирішення завдань інтелектуального аналізу даних (data mining) та прогнозування, з графічної користувальницької оболонкою для доступу до них, добре підходить для розробки нових підходів в машинному навчанні.

Система дозволяє безпосередньо застосовувати алгоритми до вибірок даних, а також викликати алгоритми з програм на мові Java. На сьогоднішній день це найкраща Open source бібліотека для Data Mining. Користувачами Weka є дослідники в області машинного навчання і прикладних наук. Вона також широко використовується в навчальних цілях.

Передбачається, що вихідні дані представлені в вигляді матриці прізнакових описів об'єктів. Weka надає доступ до SQL-баз через Java Database Connectivity (JDBC) і в якості вихідних даних може приймати результат SQL-запиту. Можливість обробки безлічі пов'язаних таблиць не підтримується, але

існують утиліти для перетворення таких даних в одну таблицю, яку можна завантажити в Weka.

Функціональні можливості:

Weka має користувальницький інтерфейс, показаний на рис. 2.1 Explorer, але та ж функціональність доступна через компонентний інтерфейс Knowledge Flow і з командний рядок.

Explorer має кілька панелей.

- Preprocess panel дозволяє імпортувати дані з бази, CSV файлу і т.д., і застосовувати до них алгоритми фільтрації, наприклад, переводити кількісні ознаки в дискретні, видаляти об'єкти і ознаки по заданому критерію.

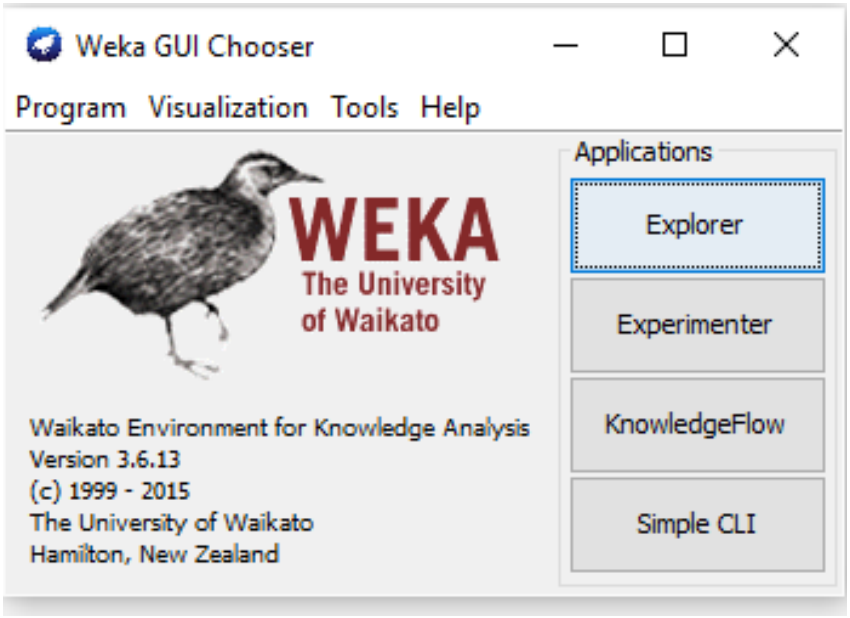


Рисунок 2.1 – Інтерфейс Weka [10]

- Classify panel дозволяє застосовувати алгоритми класифікації та регресії (в Weka вони не розрізняються і називаються classifiers) до вибірки даних, оцінювати передбачувану здатність алгоритмів, візуалізувати помилкові передбачення, ROC-криві, і сам алгоритм, якщо це можливо (зокрема, дерева рішень).

- Associate panel переймається тим виявлення всіх значущих взаємозв'язків між ознаками.
- Cluster panel дає доступ до алгоритму k-середніх, EM-алгоритму для суміші гауссіанов і іншим.
- Select attributes panel дає доступ до методів відбору ознак.
- Visualize будує матрицю графіків розкиду (scatter plot matrix), дозволяє вибирати і збільшувати графіки, і т.д.

Інтеграція:

Weka надає прямий доступ до бібліотеки реалізованих в ній алгоритмів. Це дозволяє легко використовувати вже реалізовані алгоритми з інших систем, реалізованих на Java. Наприклад, ці алгоритми можна викликати з MATLAB.

2.2 Веб-СОА.

Поступове збільшення обсягів інформації в сучасних інформаційних системах формує необхідність у розробці та впровадженні нових підходів до аналізу великих обсягів даних (Великих даних, Big Data). Для аналізу Великих даних розробляються нові системи та методи інтелектуального аналізу даних, засновані на концепції розподілених та незалежних обчислень, у тому числі системи сервісів, що реалізовані у вигляді сервісно-орієнтованих архітектур (СОА) та Веб-СОА. У Веб-СОА стандартні методи аналізу даних представлені в вигляді окремих Веб-сервісів доступних для взаємодії з іншими сервісами чи Веб-агентами за допомогою стандартних протоколів мережі Інтернет.

Розглянемо поширені системи Веб-сервісів для інтелектуального аналізу даних.

Можна виділити наступні системи Веб-сервісів для інтелектуального аналізу даних: Weka4WS, Orange4WS, KNIME, MATLAB, ClowdFlows і DAME.

Доступ до Веб-сервісів, зазвичай, реалізують на базі протоколів Simple Object (Simple Object Access Protocol, SOAP) та Representational State (Representational State Transfer, REST).

Призначення Веб-сервісів та варіанти взаємодії з ними описують на мові Web Service Definition Language (WSDL) [2].

Система Weka4WS є розширенням пакету Weka, яке забезпечує виконання методів інтелектуального аналізу даних у вигляді Веб-сервісів на розподілених вузлах мережі Інтернет. Weka4WS побудована на базі Грідсервісів за технологією віддаленого управління ресурсами (Web Services Resource Framework, WSRF) та пакету Globus для управління виконанням методів інтелектуального аналізу у вигляді потоків розрахунків (Workflows). Архітектура Weka4WS передбачає аналіз даних різними методами з фіксованими параметрами чи одним методом з різними параметрами на розподілених вузлах мережі Інтернет [3].

Система Orange4WS є розширенням пакету Orange. В порівнянні з Orange, Orange4WS включає в себе такі функції, як можливість застосування Веб-сервісів в як ості компонентів потоків розрахунків. Потоки розрахунків мають вигляд онтологій, в яких описані типи та джерела даних, сервіси

інтелектуального аналізу даних в абстрактному вигляді, зручному для автоматичного управління. В Orange4WS є можливість імпорту зовнішніх Вебсервісів за їх WSDL-описом [5].

Система KNIME також заснована на архітектурі Веб-сервісів. Клієнтські вузли представлені у вигляді Веб-сервісів, які компонуються в потоки розрахунків. Потоки розрахунків включають вузли для попередньої обробки даних, побудови та візуалізації аналітичних моделей [6].

В MATLAB реалізовані два типи взаємодії з Веб-сервісами – REST та SOAP. Для аналізу великих обсягів даних існує можливість створення Вебсервісів та завантаження їх у хмарні системи [7].

Окрім розглянутих систем Weka4WS, Orange4WS, KNIME та MATLAB, розроблених у вигляді локального програмного забезпечення з графічним інтерфейсом, існує дві системи, які потребують встановлення браузера та наявності доступу до мережі Інтернет: ClowdFlows та DAME.

Система ClowdFlows розроблена у вигляді Веб-додатку для управління потоками розрахунків, які виконуються у хмарному середовищі. Система має відкритий програмний код для створення та спільного виконання потоків розрахунків в задачах інтерактивного машинного навчання та інтелектуального аналізу даних. ClowdFlows підтримує взаємодію з Вебсервісами за протоколом SOAP [8].

Система DAME створена у вигляді пакету Веб-додатків для взаємодії з розподіленими середовищами. DAME забезпечує зручний доступ до великих обсягів даних у хмарних системах та включає Веб-сервіси для обробки та аналізу даних [9].

Порівняємо системи за зручністю їх застосування.

2.2.1 Порівняльний аналіз.

Для порівняння систем Веб-сервісів були вибрані наступні критерії :

1. SOAP і REST (C1) – підтримка популярних протоколів доступу до Веб-сервісів. Майже всі системи підтримують SOAP, у системі DAME реалізована взаємодія за протоколом REST. Лише в MATLAB реалізовані обидва типи протоколів.

2. Кросплатформеність (C2) – можливість встановлення на операційних системах Windows, Linux і Mac. Усі системи є кросплатформеними.

3. Хмарні обчислення (C3) – підтримка хмарних обчислень та Грідобчислень при вирішенні складних і трудомістких задач інтелектуального аналізу даних. Лише 3 системи Weka4WS, MATLAB й ClowdFlows підтримують хмарні та Грід-обчислення.

4. Аналітична універсальність (C4) – можливість застосовування кількох різних методів інтелектуального аналізу даних в одному потоці розрахунків. Стандартні три групи методів інтелектуального аналізу реалізовані в системах Weka4WS, Orange4WS, KNIME та MATLAB.

5. Потоки розрахунків (С5) – можливість виконання потоків розрахунків, збереження результатів та проведення експериментів. Потоки розрахунків реалізовані в чотирьох системах Weka4WS, Orange4WS, KNIME та ClowdFlows.

6. Імпорт Веб-сервісів (С6) – підтримка імпорту сторонніх Вебсервісів з мережі Інтернет.

7. Відкритий програмний код (С7) – можливість розширення та поліпшення коду. Чотири системи Weka4WS, Orange4WS, KNIME і ClowdFlows мають відкритий програмний код.

8. Веб-інтерфейс (С8) – можливість роботи у Веб-браузерах. Дві системи ClowdFlows та DAME реалізовані у вигляді Веб-додатків, проте вимагають підключення до мережі Інтернет.

9. Веб-збереження (С9) – можливість збереження даних у Вебсховищі, що дозволяє виконувати різні експерименти з даними без повторного завантаження. Така особливість наявна лише в DAME. У таблиці 2.8 наведені результати порівняння систем Веб-сервісів для інтелектуального аналізу даних.

Таблиця 2.8 – Результати порівняння Веб-СОА

	C1		**C2**	**C3**	**C4**				**C5**	**C6**	**C7**	**C8**	**C9**	**Σ**
	SOAP	*REST*	*Кросплатформеність*	*Хмарні обчислення*	*Класифікація*	*Кластеризація*	*Зменшення розмірності*	*Універсальність системи*	*Потоки розрахунків*	*Імпорт Веб-сервісів*	*Відкритий програмний код*	*Веб-інтерфейс*	*Веб-збереження*	
Weka4WS	+	–	+	+	+	+	+	+	+	–	+	–	–	**9**
Orange4WS	+	–	+	–	+	+	+	+	+	+	+	–	–	**9**
KNIME	+	–	+	–	+	+	+	+	+	+	+	–	–	**9**
MATLAB	+	+	+	+	+	+	+	+	–	+	–	–	–	**9**
ClowdFlows	+	–	+	+	+	+	–	–	+	+	+	+	–	**9**
DAME	–	+	+	–	+	+	–	–	–	–	–	+	+	**6**
	5	**2**	**6**	**3**	**6**	**6**	**4**	**4**	**4**	**4**	**4**	**3**	**1**	

З результатів порівняльного аналізу слідує, що розглянуті системи Вебсервісів для інтелектуального аналізу даних поки не відповідають усім критеріям зручності застосування, так, у системах Weka4WS, Orange4WS, KNIME і ClowdFlows реалізовані 9 з 13 пунктів порівняльного аналізу та у системі DAME лише 6 з 13 пунктів (див. табл. 2.8).

У системах Orange4WS та KNIME відсутня підтримка хмарних обчислень, яка полегшує та пришвидшує роботу з великими обсягами даних.

Основною перевагою системи Weka4WS порівняно з системою ClowdFlows є універсальність системи за кількістю реалізованих методів інтелектуального аналізу даних, а порівняно з системою MATLAB – підтримка виконання потоків розрахунків та наявність відкритого програмного коду, у свою чергу головною перевагою цих систем порівняно з системою Weka4WS є можливість імпорту зовнішніх Веб-сервісів.

Отже, після порівнянь систем Веб-сервісів, найбільш перспективною для вирішення комплексних задач з аналізу Великих даних є система Weka4WS, обчислювальні можливості якої обмежуються лише кількістю та

характеристиками обладнання, а широкий перелік реалізованих методів інтелектуального аналізу забезпечує можливість створення комбінованих потоків розрахунків для аналізу різнотипних даних.

2.2.2 Архітектура системи Weka4WS

Weka4WS була розроблена з використанням бібліотек Java Web Services Framework ресурсу (WSRF), що надаються Globus Toolkit (GT) і використовує свої послуги для стандартних Grid-функції, такі як безпека і передачі даних. Додаток складається з двох окремих частин:

• Вузол користувача: це та частина, де працює клієнтська сторона програми. Це робиться за допомогою розширення Weka графічного інтерфейсу користувача, модуль клієнта і бібліотеки Weka. Це вимагає присутності ядра Globus Java-WS (компонент Globus Toolkit);

• Обчислювальний вузол: на стороні сервера додатка, він дозволяє виконувати завдання інтелектуального аналізу даних за допомогою веб-служб. Необхідна повна установка Globus Toolkit. Дані, що підлягають видобутку можуть бути розташовані або в вузлі користувача, або в обчислювальному вузлі, або по якому-небудь віддаленому ресурсу (наприклад, деякі загальні сховища). Якщо дані не доступні на обчислювальному вузлі вони передаються за допомогою протоколу GridFTP, з високою продуктивністю, безпечністю і надійністю передачі даних, яка є частиною Globus Toolkit.

На рис. 2.2 показані компоненти Weka4WS програмного забезпечення на вузлі користувача і обчислювального вузла, і взаємодія між ними. Користувач працює на графічному інтерфейсі користувача для підготовки і контролю завдань інтелектуального аналізу даних: ті, які будуть виконуватися на місці будуть здійснюватися через місцеву бібліотеку Weka, в той час як ті, які будуть виконуватися на віддаленому хості будуть оброблятися клієнтським модулем, який буде взаємодіяти з обчислювальним вузлом з використанням пропонованих послуг Globus Java WS Core, встановленого на машину користувача вузла. У кращому випадку мережевий обмін даними між клієнтським модулем і веб-службою буде на основі «Push-style» режиму роботи механізму доставки NotificationMessage, де модуль клієнта є NotificationConsumer і веб-служби є NotificationProducer. Клієнтський модуль викликає службу і чекає на завершення завдання.

Є певні обставини, при яких доставка NotificationMessage на основі «Pushstyle» не підходить, наприклад, коли клієнт знаходиться за NAT маршрутизатором/брандмаузером, як показаний на рис. 2.3, так як повідомлення, відправлені клієнту будуть заблоковані, якщо з'єднання буде ініційоване самим клієнтом.

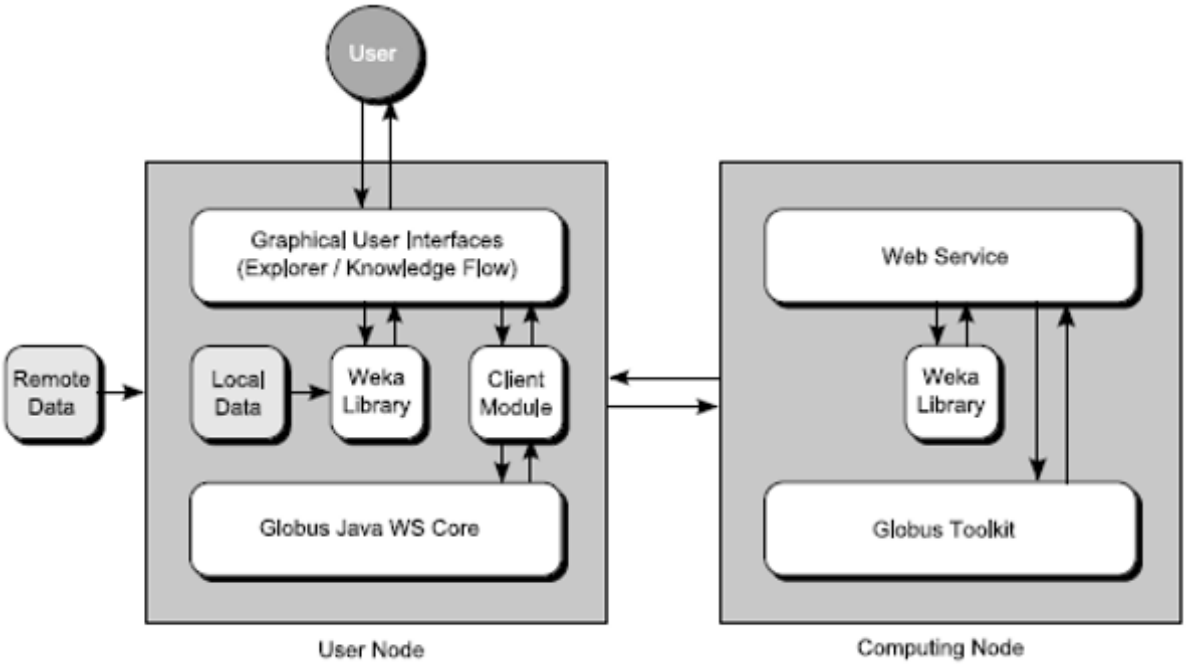


Рисунок 2.2 - Вузол користувача та серверний вузол [4]

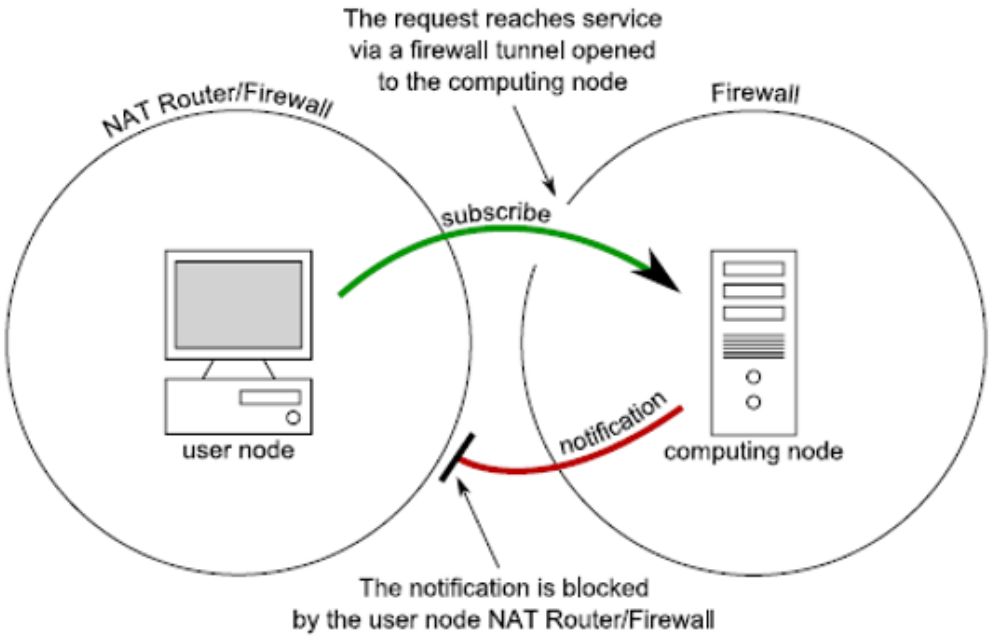


Рис. 2.3 - Блокування повідомлень, при роботі на компютері з захистом в вигляді NAT маршрутизатора / брандмауера [4]

Коли клієнт підписується на повідомлення про завершення необхідної задачі, яка також переходить до номера порту служби, що генерується

випадковим чином, до якого клієнту потрібно перейти, щоб отримати повідомлення: з цього моменту клієнт буде виступати в якості сервера, очікування повідомлення до даного порту.

Коли в обчислювальному вузлі будуть готові результати, обчислювальним вузлом ініціюється з'єднання, але відправити повідомлення не вдасться, тому що воно блокується вузлом користувача NAT маршрутизатора / брандмауера.

Єдиний спосіб отримання результатів клієнтським модулем -- працювати в режимі “pull-style”, тобто безперервно намагатися отримувати, в певні задані інтервали часу, результати обчислювального вузла. Для того, щоб система автоматично адаптувалася до всіх можливих сценаріїв мережі, прозоро для користувача, вузол користувача запускає pull режим за замовчуванням. У момент підписки на повідомлення також потребує негайної відправки фіктивного повідомлення, мета якого полягає в тому, щоб перевірити, чи може вузол користувача отримувати повідомлення. Коли вузол користувача отримає це фіктивне повідомлення, то буде переключитися на «Push-style» режим, в іншому випадку буде зберігатися режим “pull-style”.

На стороні обчислювальних вузлів, веб-служба використовує послуги Інструментарій Globus взаємодіяти з вузлом користувача і відповіді на свої запити, посилаючись на необхідний алгоритм в базовій Weka бібліотеці. Співвідношення між веб-служб і Globus Toolkit складається з трьох частин: Web Services побудовані з використанням деяких бібліотек, що надаються Globus Toolkit, вони розгорнуті і розміщується над контейнером Globus Toolkit, і вони використовують деякі послуги Інструментарій Globus, як безпека і механізм повідомлення для доступу до даних і взаємодіяти з вузлом користувача.

Вузол користувача

Призначений для користувача вузол складається з трьох компонентів: графічний інтерфейс користувача (GUI), модуль клієнта і бібліотека Weka. GUI зроблений компонентами Weka провідника і потоку знань, розширений для

підтримки виконання віддалених завдань інтелектуального аналізу даних. Провідник являє собою інструмент для «вивчення» даних через дані попередньої обробки, інтелектуалного аналізу і візуалізації. Потік знань має по суті ті ж функції провідника, але з випадаючим меню, що дозволяє будувати інтелектуальний аналіз даних робочих процесів.

Локальні завдання виконуються через виклик місцевої бібліотеки Weka, в той час як віддалені здійснюється через клієнтський модуль, який виконує роль посередника між GUI і віддаленими веб-службами. Кожне завдання виконується у власному потоці, що дозволяє запускати кілька завдань паралельно, або за допомогою провідника або потоку знань. Дистанційні хости адреси завантажуються з текстового файлу, розташованого в каталозі додатки. Цей текстовий файл для читання в фоновому режимі, коли додаток запускається і для кожного віддаленого хоста перевіряється, що:

• контейнер Globus і GridFTP працюють;

• Weka4WSService розгортається;

• версія клієнта і сервіса однакова.

Тільки ті хости, які проходять всі перевірки будуть доступні користувачеві в графічному інтерфейсі. Для того, щоб врахувати можливі зміни конфігурації мережі сітки без перезапуску програми, адреса віддалених хостів може бути перевантажений в будь-який час, просто натиснувши на кнопку надану для цієї мети. Статичний список адрес, що зберігаються в текстовому файлі на клієнтській машині насправді не ідеальне рішення, як Globus Toolkit вже забезпечує точку запиту (блідого Index Service) про характеристики фізичної системи в сітці і дозволяє клієнтам або агенти для пошуку послуг, які приходять і йдуть динамічно на сітці. Вибір за допомогою цього файлу було прийнято в якості тимчасового рішення для того, щоб прискорити розробку прототипу Weka4WS і зосередитися на більш складних і важливих аспектів застосування. Використання індексного служби Globus замість текстового файлу планується ввести найближчим часом версій Weka4WS.

Обчислювальний вузол

Обчислювальний вузол включає в себе два компоненти: веб-служби та бібліотеки Weka. Веб-служба відповідає на запит вузла користувача, викликавши необхідний алгоритм в базовому Weka бібліотеці. Виклик алгоритмів виконуються в асинхронному режимі, тобто, клієнт відправляє завдання в режимі неблокируемого і результати або повідомлені до нього всякий раз, коли вони обчислюються (режим пуш-стиль) або вони багаторазово перевірені на готовність клієнт (режим висувною стиль), в залежності від конфігурації мережі.

У таблиці 2.9 перераховані операції, що надаються кожним веб-сервісом в рамках Weka4WS. Перші три операції використовуються для запиту виконання конкретного завдання інтелектуального аналізу даних; Наступнітри операції таблиці – деякі корисні додаткові операції, в той час як останні чотири операції пов'язані з WSRF-спецмеханізмами заклику. Операція GetVersion викликається в момент перевірки хостів, і використовується для перевірки того, однакові версії клієнта і сервіса. Операція notifCheck викликається безпосередньо після операції підписки, щоб перевірити, чи здатний клієнт отримувати повідомлення. Всі інші операції описані в таблиці 2.10.

Параметри, необхідні для кожної операції наведені в таблиці 2.10; операції GetVersion і notifCheck не потребують будь-якого параметра.

TaskId поле виключно з метою зупинки завдання. Поле алгоритму комплексного типу, показане в таблиці 2.10, яке містить два підполя: ім'я та параметри. Набір даних і тестовий набір інші два поля, складного типу, який містить чотири підполя: FileName, рядок, що містить ім'я файлу набору даних (або тестового набору), Filepath, рядок, що містить повний шлях (ім'я файлу в комплекті) набір даних, dirPath, рядок, що містить шлях (ім'я файлу виключено) з набору даних і CRC, який визначає контрольну суму файлу набір даних (або тест комплект).

Таблиця 2.9 – Операції обчислювального вузла

Operation	Description
classification	Submits the execution of a classification task.
clustering	Submits the execution of a clustering task.
associationRules	Submits the execution of an association rules task.
stopTask	Explicitly requests the termination of a given task.
getVersion	Returns the version of the service.
notifCheck	Checks whether the client is able to receive notifications.
createResource	Creates a new stateful resource.
subscribe	Subscribes to notifications about resource properties changes.
getResourceProperty	Retrieves the resource property values.
destroy	Explicitly requests destruction of a resource.

Поле classIndex є цілим числом, що позначає, який атрибут набору даних повиннен бути розглянутий атрибутом класу при виклику алгоритму класифікації або кластеризації.

Поле testOptions має складний типу, що містить три підполя: TestMode, ціле число, яке представляє тестовий режим, щоб бути застосовані до класифікації або алгоритму кластеризації (1 для крос-перевірки, 2 для Відсотку розколу, 3, щоб використовувати набір навчання та 4, щоб використовувати окремий тестовий набір), і numFolds два додаткових поля, використовувані при застосуванні перехресної перевірки або тестового режиму в процентах поділу.

Поле evalOptions використовується спеціально для алгоритмів класифікації і містить кілька підполів, як costMatrix (для оцінки помилок по відношенню до матриці витрат), outputModel (для виведення модель, отримана з повного набору професійної підготовки), та інші наведені в таблиці 2.11.

Таблиця 2.10 – Параметри операцій

Parameters type	Field name	Field type
classificationParameters	taskID	long
	algorithm	algorithmType
	dataset	datasetType
	testset	datasetType
	classIndex	int
	testOptions	testOptionsType
	evalOptions	evalOptionsType
clusteringParameters	taskID	long
	algorithm	algorithmType
	dataset	datasetType
	testset	datasetType
	classIndex	int
	testOptions	testOptionsType
	selectedAttributes	array of int
associationRulesParameters	taskID	long
	algorithm	algorithmType
	dataset	datasetType
stopParameters	taskID	long

Поле selectedAttributes використовується спеціально для алгоритмів кластеризації і містить ті атрибути в даних, які повинні бути проігноровані при кластеризації.

Таблиця 2.11 – Поля параметрів операцій

Fields type	Subfield name	Subfield type
algorithmType	name	string
	parameters	string
datasetType	fileName	string
	filePath	string
	dirPath	string
	crc	long
testOptionsType	testMode	int
	numFolds	int
	percent	int
evalOptionsType	costMatrix	string
	outputModel	boolean
	outputConfusion	boolean
	outputPerClasss	boolean
	outputSummary	boolean
	outputEntropy	boolean
	rnd	int

Робота системи

У цьому прикладі ми припускаємо, що клієнт дає запит на виконання завдання класифікації на наборі даних, який присутній на лише вузлі користувача, а на обчислювальному вузлі він відсутній. Це слід розглядати як найгірший випадок, оскільки в багатьох випадках набір даних, які видобуватимуться вже є на вузлі сітки, де завдання повинно бути представлене. Коли віддалений аналіз даних завдання запускається генерується, так званий TaskID - однозначний ідентифікаційний номер: це число пов'язане з цим конкретним завданням і використовується, коли операція stopTask викликається для ідентифікації завдання серед усіх інших, які працюють на обчислювальному вузлі. Весь процес виклику може бути розділений на 8 кроків, показаних на рис. 2.4:

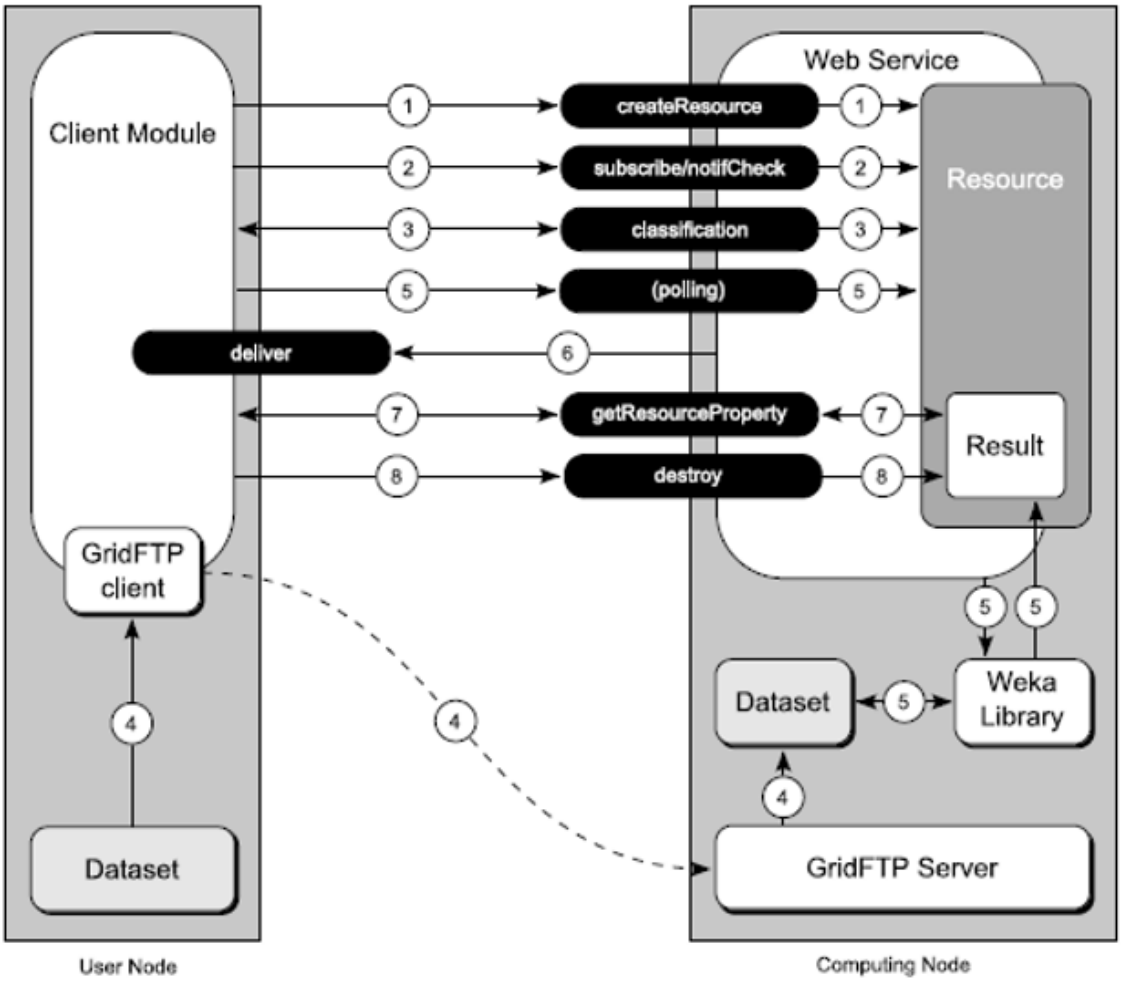


Рисунок 2.4 – Weka4WS: завдання заклику [4]

1. Створення ресурсу: операція createResource викликається для створення нового ресурсу, який буде підтримувати свій стан протягом всіх наступних викликів веб сервіса до його руйнування. Стан зберігається як властивості ресурсу; більш конкретно, це властивість Результат, докладно описано в таблиці 2.12, використовується для зберігання результатів аналізу даних завдання. Перші три поля виведення моделей і / або моделі, результати оцінки та додаткова інформація про візуалізацію і прогнозування. Останні три поля, винятки, зупинки і готові використовуватися тільки тоді, коли виникають певні обставини, якщо на етапі обчислення щось йде не так, то поле виключення встановлюється відповідним чином: його логічний параметр встановлюється в значення true, і в рядку параметрів, повідомлення, зберігається, як виключення;

Таблиця 2.12 – Властивості ресурсу

Field name	Type	Subfield names
model	ModelResult	model, models
evaluation	EvalResult	summary, classDetails, matrix
visualization	VisResult	predInstances, predictions, plotShape, plotSize
exception	Weka4WSException	thrown, message
stopped	boolean	
ready	boolean	

Після того, як ресурс був створений веб-служба повертає посилання на кінцеву точку (EPR) створюваного ресурсу. ЕПР є унікальним в рамках Веб-сервісів, і відрізняє цей ресурс з усіх інших ресурсів в цьому сервісі. Наступні запити від клієнтського модуля буде спрямований на ресурс, визначений цим ЕПР;

2. Підписка на повідомлення і перевірка повідомлень: операція підписки викликається для того, щоб отримувати повідомлення про зміни, які відбуватимуться у Результаті. Після завершення завдання інтелектуального аналізу даних модуль Клієнт буде повідомлений про це. Відразу після операції підписки викликається операція notifCheck для запиту негайної відправки фіктивного повідомлення, щоб перевірити, чи здатний клієнт отримувати повідомлення: коли і якщо вузол користувач отримає цей фіктивний повідомлення буде переключатися в режимі “push-style”, в іншому випадку буде зберігатися в “pull-style”;

3. Подання Завдання: операція класифікації викликається для того, щоб задати для виконання завдання класифікації. Ця операція вимагає 7 параметрів див. табл. 2.10, серед яких TaskID згадувалося раніше. Операція повертає об'єкт Response, детальніше в таблиці 2.13. Якщо копія набору даних ще не доступна в обчислювальному вузлі, то для поля datasetFound встановлюються значення False , а поле dirPath встановлюються в URL, де набір даних повинен бути

завантажений; Аналогічно, коли потрібна перевірка на тестовому наборі, який відрізняється від набору даних і тестового набору ще не доступний на обчислювальному вузлі, то поле testsetFound встановлено значенням false. URL, де тестовий набір повинен бути завантажений так само, як набір даних.

Таблиця 2.13

Field name	Type
datasetFound	boolean
testsetFound	boolean
dirPath	string
exception	Weka4WSException

4. Передача файлів: так як в цьому прикладі ми припускали, що набір даних не був доступним в обчислювальному вузлі, модуль Клієнт повинен передати його в обчислювальний вузол. З цією метою клієнт Java-GridFTP інстанціюється і ініціаюється для взаємодії з GridFTP сервером обчислювального вузла: набір даних (або тестовий набір) потім переносять в машину обчислювального вузла і зберігаються в директорії, шлях якої був вказаний в полі dirPath, що міститься в об'єкті Відповіді, повернутому операцією класифікації;

5. Видобуток даних: аналіз класифікації запускається веб-службою за допомогою виклику відповідного класу Java в бібліотеці Weka. Результати обчислень зберігаються в значеннях Result в ресурсі, створеному на кроці 1;

6. Повідомлення прийому: як тільки властивість Result змінено повідомлення відправляється в модуль клієнта, викликаючи його неявну операцію deliver. Цей механізм дозволяє асинхронну доставку результатів виконання, коли вони генеруються. У тих випадках, коли клієнт не може отримувати повідомлення клієнта буде періодично перевіряти результати для готовності через значення поля Result’s результатів ready(див таблицю 2.12ри);

7. Результати вилучення: Модуль Клієнта викликає операцію GetResourceProperty для того, щоб повернути значення результату, що містять результати обчислень;

8. Знищення ресурсів: Модуль Клієнт викликає операцію знищення, яка видаляє ресурс, створений на кроці 1 [4].

2.2.3 ClowdFlows

У даному розділі представлена основна інформація для роботи із технологією ClowdFlows, розглянуті основні можливості і особливості архітектури платформи.

Архітектура системи ClowdFlows

ClowdFlows платформа в першу чергу веб-додаток та складається з двох частин: серверна та клієнтська. Сторона сервера є частиною програми, яка виконується на сервері або кластері серверів.

Сторона клієнта представляє частини системи, які виконуються на комп'ютері користувача. Архітектура системи здійснюється через доступ декількох користувачів що зображено на рис.2.5.

Графічний користувацький інтерфейс реалізований в HTML і CSS, тому він працює з використанням веб-браузерів.

Функціональні можливості реалізовані на JavaScript за допомогою library4 JQuery.

Серверна частина написана на Python з використанням Django framework5. Python було обрано тому, що це дозволило легко інтегрувати багато бібліотек для машинного навчання та інтелектуального аналізу даних інструментів, реалізованих в Python, таких як Orange і Scikitlearn. Об'єктно-реляційного відображення надає API, який пов'язує об'єкти бази даних.

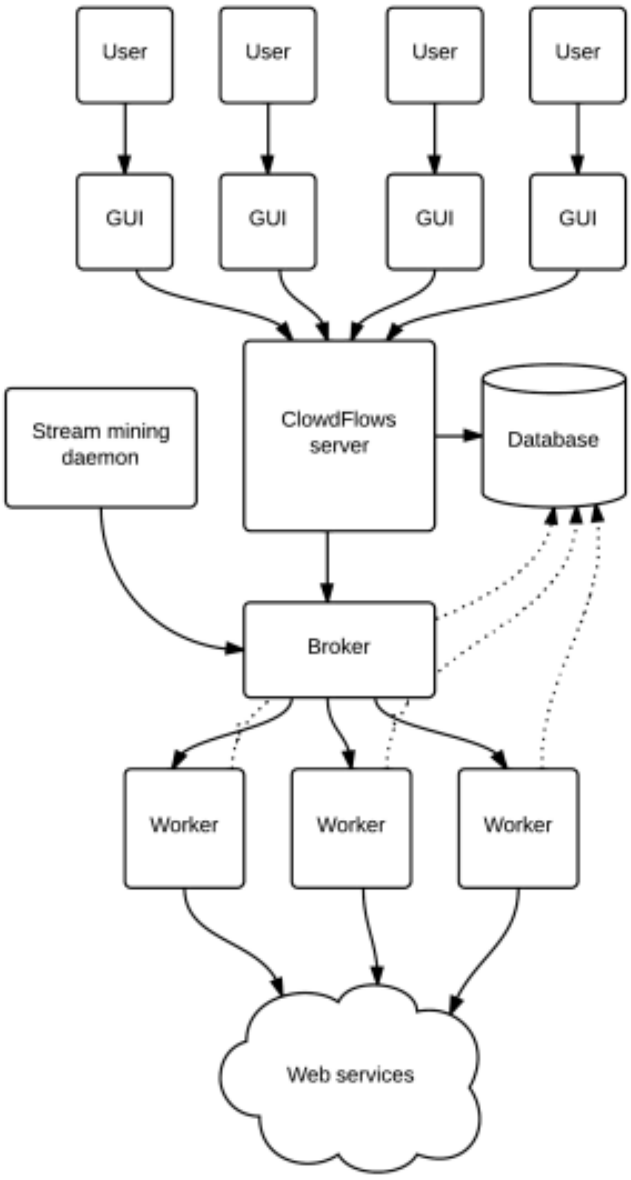


Рисунок 2.5 – Архітектура системи ClowdFlows [12]

Підтримуються бази даних PostgreSQL, MySQL, SQLite і Oracle, хоча MySQL використовується в ClowdFlows за замовченням.

ClowdFlows також може бути встановленим на декількох комп'ютерах, з активним з'єднанням за допомогою сервера обміну повідомленнями RabbitMQ6 і Джанго з Celery7, що в свою чергу дозволяє відправляти асинхронні завдання між серверами. Робочі області, які виконують завдання можуть бути встановлені на кластері машин, або на одній машині.

Для того, щоб включити споживання веб-сервісів і їх використання в якості компонентів робочого процесу, використовується PySimpleSoap library8.

PySimpleSoap надає інтерфейс для клієнта і сервера веб-служби зв'язку, яка дозволяє імпортувати веб-сервіси WSDL як компоненти робочого процесу.

Віджети та інтерфейс

Графічний інтерфейс користувача використовує візуальну парадигму програмування для побудови робочих процесів. Це спрощує уявлення складних процедур в просторовому розташуванні блоків програми.

Блоки або компоненти робочого процесу в ClowdFlows називаються віджетами.

Кожен блок отримує деякі дані або параметри на вході, виконує певне завдання, і повертає результати на виходах. Робочі процеси складаються з віджетів і з'єднань, які з'єднують виходи віджетів на входи інших віджетів.

Графічний інтерфейс користувача на платформі ClowdFlows складається з двох основних частин: сховище віджетів і середовища робочого процесу. Інтерфейс системи можна побачити на рис. 2.6.

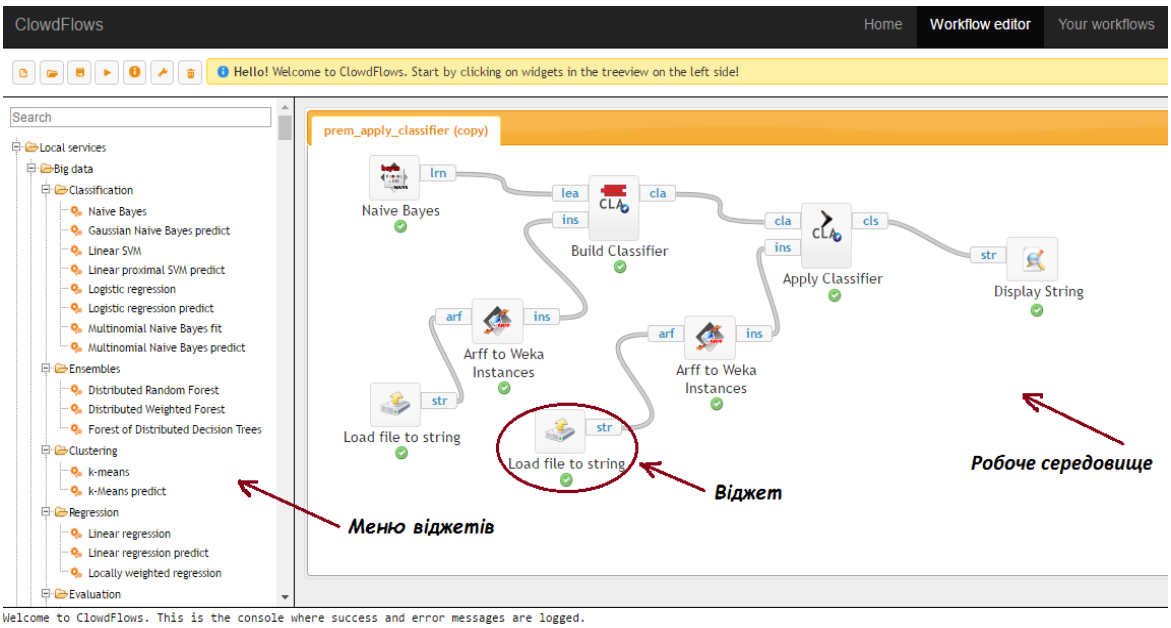


Рисунок 2.6 – Система ClowdFlows

Репозиторій віджетів, які можуть бути використані у робочому середовищі, натиснувши на них, знаходится зліва.

Віджети в ClowdFlows належать до чотирьох основних груп:

- Звичайні віджети – віджети, які просто беруть дані на вході, виконують певне завдання і повертають дані на своїх виходах.
- Віджети візуалізації можуть надавати результати як HTML і Javascript і відображати їх в діалогових вікнах в браузері.
- Інтерактивні віджети можуть відкрити модальне вікно із запитом користувача про додаткові дані під час виконання.
- Віджети управління технологічних процесів дозволяють створювати суб-технологічні процеси (тобто віджет, який містить робочий процес).

Репозиторій віджетів містить віджети з декількох різних платформ з даними. Помаранчеві віджети були реалізовані в ClowdFlows за замовченням. Алгоритми Weka були виставлені в якості веб-служб і можут бути імпортовані в платформу.

Робоче середовище реалізує переміщення, підключення, видалення і видачу команд для виконання віджетів і робочих процесів. Віджети можуть бути довільно розташовані на полотні шляхом перетягування. З'єднання між віджетами створюються шляхом вибору вихідного сигналу віджета і вхід іншого віджета. Кожен раз, коли користувач виконує дію на графічному інтерфейсі вона відправляється на сервер використовуючи асинхронний запит HTTP POST. Операції перевіряються на сервері, і повідомлення про помилки, успіх або передаються в браузер клієнта, відформатований у форматі JavaScript Object Notation (JSON) або HTML.

Панель інструментів розташована над середовищем робочого процесу і може бути використана для збереження, копіювання та видалення цілих робочих процесів. Коли робочий процес зберігається, він може бути оприлюднений. Публічні робочі процеси отримують спеціалізовані URL, які можуть бути розділені таким чином, щоб інші люди могли отримати доступ до цих робочих процесів та використовувати фукціональні моделі один одного.

Workflow Engine

Workflow Engine запускає всі виконувані віджети в робочому процесі в правильному порядку. Двигун працює з віджетами і надає їм різні стани в залежності від їх стану виконання.

Стан кожного віджета може бути одним з наступних: виконуваний файл, в очікуванні, запущений, закінчений, або не вдача. Мета виконання Workflow Engine – не мати ніяких віджетів в виконуваному або робочому стані в робочому процесі.

Виконувані віджети – це віджети, стан попередників яких є завершеним, чи немає попередників взагалі.

Попередник віджета – це віджет, вихід якого підключений до входу свого нащадка. В очікуванні віджетів віджети, стан яких не є ні виконуваним, ні запущеним, ні закінченим і не в стані невдачі.

Запущені віджети – це віджети, які були виконані, але поки ще не завершені або в стані невдачі.

Готові і невдалі віджети – це віджети, які завершили виконання або успішно або невдало.

Невдалий або готовий віджет переходить в режим очікування або виконання щоразу, коли з'єднання, що веде до віджету видаляється, додається або змінюється, або коли значення параметра цього віджета змінюється, або коли користувач вручну змінює стан віджета, щоб файл виконувався. Коли запускається робочий процес двигуна постійно перевіряє наявність віджетів, які виконуються і виконує їх. Кожен раз, коли два або більше віджетів є виконуваними в той же час вони асинхронно паралельно виконуються, оскільки вони є незалежними. Механізм стану віджета гарантує, що два віджета, де вхід одного залежить від виходу іншого не можуть виконуватися одночасно. Виконання робочого процесу завершується, коли немає виконуваних або запущених віджетів.

2.2.4 Аналіз даних в режимі реального часу

Цей розділ описує адаптацію ClowdFlows до обробки даних в реальному часі.

Віджет в статичному режимі виконується кінцеву кількість разів, і робочий процес має кінцевий час виконання. Віджети в потоковому робочому процесі виконуються потенційно-нескінченну кількість разів, і вручну, поки користувач не припинить роботу. Ще однією важливою відмінністю між звичайними робочими процесами і потоковими є дані на вході. Дані, які обробляються регулярними робочими процесами доступні в цілому протягом усього часу обробки, в той час як дані потокового процесу є потенційно нескінченним і звернутис до них можна в конкретний проміжок часу.

Для того, щоб працювати з потенційно нескінченними потоками даних механізм виконання робочого процесу розбивається на декілка малих тимчасових інтервалів паралельно. Кількість паралельності і частота виконання є параметрами, які можуть бути (за умови наявності апаратних засобів), модифіковані для кожного потоку, щоб максимізувати пропускну здатність [12].

2.3 Висновки до розділу

Безумовним лідером для інтелектуального аналізу даних серед офлайн систем є Weka, оскільки фукціонал даної платформи є більш зручним та наочнішим, що можна побачити в таблиці 2.7.

Серед Веб-СОА слід відзначити ClowdFlows, оскільки він має надпотужну систему віджетів і зручний та інтуїтивно зрозумілий інтерфейс. На робочій області завдяки простим операціям з елементами дозволено створювати складні та потужні моделі для аналізу даних. Можливість зєднання елементів а також їх довілного розміщення на робочій області. Є надзвичайно зручною для звичайних користучачів, а також відмінно підходить для візуалізації складних процедур з даними. Також існує підтримка веб сервісів Weka у вигляді окремих віджетів.

3 РОЗРОБКА ФУНКЦІОНАЛЬНИХ МОДЕЛЕЙ І АНАЛІЗ ДАНИХ

У цому розділі ми будемо проводити аналіз даних клінічного дослідження впливу анти-епілептичного препарату.

Для того, щоб проаналізувати вхідні дані необхідно створити функціональну модел аналізу. Для цього ми обрали такі інструменти, як Weka та ClowdFlows.

3.1 Вхідні дані

3.1.1 Формат

Дата-сет з 238 спостереженнями за наступними 6 змінними.

Treatment группа лікування, фактор з рівнями *placebo* і *progabide*.

Лікування та хімічний препарат.

Base число затримань до суду.

Age вік пацієнта.

seizure.rate кількість YFGFдів (змінна відгуку).

Period період лікування, впорядкований фактор з рівнем від 1 до 4.

Subject ідентифікатор пацієнта, фактор з рівнем від 1до 238.

В цьому клінічному дослідженні 238 пацієнтів, страждаючих епілепсією, були розподілені випадковим чином в групи (табл. 3.1), які або отримували антіепілептичний препарат Progabide(наркотик) або плацебо в додаток до стандартної хіміотерапії. Кількість припадків в кожному з чотирьох двотижневих періодів була записана для кожного пацієнта, а також із кількістю припадків за 8 тижнів до того. Основне питання, чи зменшує progabide кількість нападів в порівнянні зі звичайним плацебо.

Таблиця 3.1 – Вхідний датасет

ID	Y_1	Y_2	Y_3	Y_4	Trt	Base	Age
104	5	3	3	3	0	11	31
106	3	5	3	3	0	11	30
107	2	4	0	5	0	6	25
114	4	4	1	4	0	8	36
116	7	18	9	21	0	66	22
118	5	2	8	7	0	27	29
123	6	4	0	2	0	12	31
126	40	20	23	12	0	52	42
130	5	6	6	5	0	23	37
135	14	13	6	0	0	10	28
141	26	12	6	22	0	52	36
145	12	6	8	4	0	33	24
201	4	4	6	2	0	18	23
202	7	9	12	14	0	42	36
205	16	24	10	9	0	87	26
206	11	0	0	5	0	50	26
210	0	0	3	3	0	18	28
213	37	29	28	29	0	111	31
215	3	5	2	5	0	18	32
217	3	0	6	7	0	20	21
219	3	4	3	4	0	12	29
220	3	4	3	4	0	9	21
222	2	3	3	5	0	17	32
226	8	12	2	8	0	28	25
227	18	24	76	25	0	55	30
230	2	1	2	1	0	9	40
234	3	1	4	2	0	10	19
238	13	15	13	12	0	47	22
101	11	14	9	8	1	76	18
102	8	7	9	4	1	38	32
103	0	4	3	0	1	19	20
108	3	6	1	3	1	10	30
110	2	6	7	4	1	19	18
111	4	3	1	3	1	24	24
112	22	17	19	16	1	31	30
113	5	4	7	4	1	14	35
117	2	4	0	4	1	11	27
121	3	7	7	7	1	67	20
122	4	18	2	5	1	41	22
124	2	1	1	0	1	7	28
128	0	2	4	0	1	22	23
129	5	4	0	3	1	13	40
137	11	14	25	15	1	46	33
139	10	5	3	8	1	36	21
143	19	7	6	7	1	38	35
147	1	1	2	3	1	7	25

Приклад

```
data("epilepsy", package = "HSAUR")
  library(lattice)
  dotplot(I(seizure.rate / base) ~ period | subject, data
= epilepsy,
          subset = treatment == "Progabide")
  dotplot(I(seizure.rate / base) ~ period | subject, data
= epilepsy,
          subset = treatment == "Progabide")
```

3.2 Метод опорних векторів (SVM)

Метод опорних векторів (Support Vector Machine - SVM) відноситься до групи граничних методів. Він визначає класи за допомогою кордонів областей.

За допомогою даного методу вирішуються завдання бінарної класифікації. В основі методу лежить поняття площин рішень.

Площина (plane) рішеннь розділяє об'єкти з різною класовою приналежністю.

На рис 3.1 наведено приклад, в якому беруть участь об'єкти двох типів. Розділяє лінія задає кордон, праворуч від якої - всі об'єкти типу brown (коричневий), а зліва - типу yellow (жовтий). Новий об'єкт, що потрапляє направо, класифікується як об'єкт класу brown або як об'єкт класу yellow, якщо він розташувався ліворуч від розділяє прямий. У цьому випадку кожен об'єкт характеризується двома вимірами.

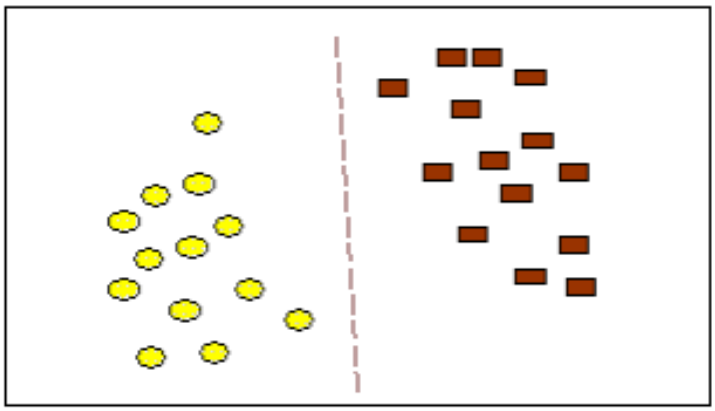


Рисунок 3.1 – Поділ класів прямою лінією [12]

Мета методу опорних векторів - знайти площину, що розділяє дві множини об'єктів. Така площина показана на рис. 3.2. На цьому малюнку безліч зразків поділено на два класи: жовті об'єкти належать класу A, коричневі - класу B.

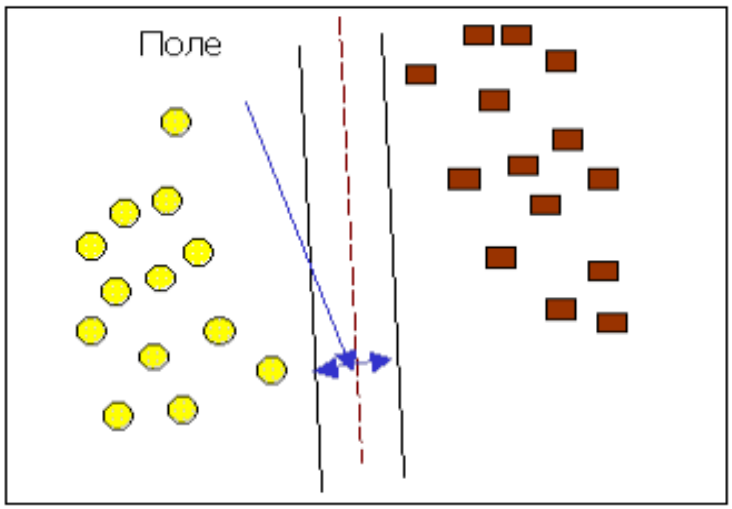


Рисунок 3.2 – Площина, що розділяє класи [12]

Метод відшукує зразки, що знаходяться на кордонах між двома класами, тобто опорні вектора; вони зображені на рис. 3.3.

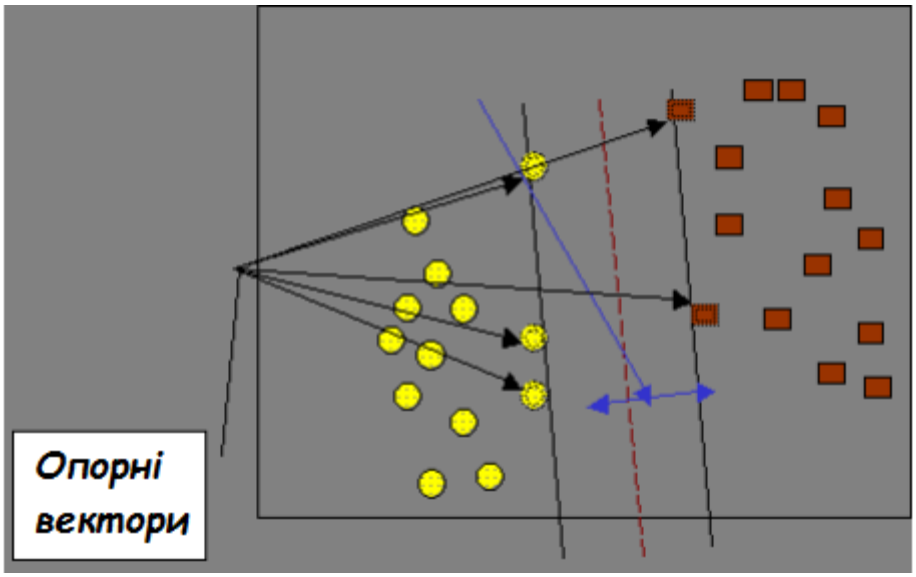


Рисунок 3.3 – Опорні вектори [12]

Опорними векторами називаються об'єкти множин, що лежать на кордонах областей. Класифікація вважається хорошою, якщо область між кордонами порожня.

На рис. 3.3 показано п'ять векторів, які є опорними для даної множини.

Лінійний SVM

Рішення завдання бінарної класифікації за допомогою методу опорних векторів полягає в пошуку деякої лінійної функції, яка правильно поділяє набір даних на два класи. Розглянемо задачу класифікації, де число класів дорівнює двом.

Завдання можна сформулювати як пошук функції f (x), що приймає значення менше нуля для векторів одного класу і більше нуля - для векторів іншого класу. В якості вихідних даних для вирішення поставленого завдання, тобто пошуку классифицирующей функції f (x), дан тренувальний набір векторів простору, для яких відома їх приналежність до одного з класів. Сімейство класифікують функцій можна описати через функцію f (x). Гіперплоскость визначена вектором a й значенням b, тобто f (x) = ax + b. Рішення даного завдання проілюстровано на рис. 3.4.

В результаті рішення задачі, тобто побудови SVM-моделі, знайдена функція, приймаюча значення менше нуля для векторів одного класу і більше нуля – для векторів іншого класу. Для кожного нового об'єкта негативне або позитивне значення визначає приналежність об'єкта до одного з класів.

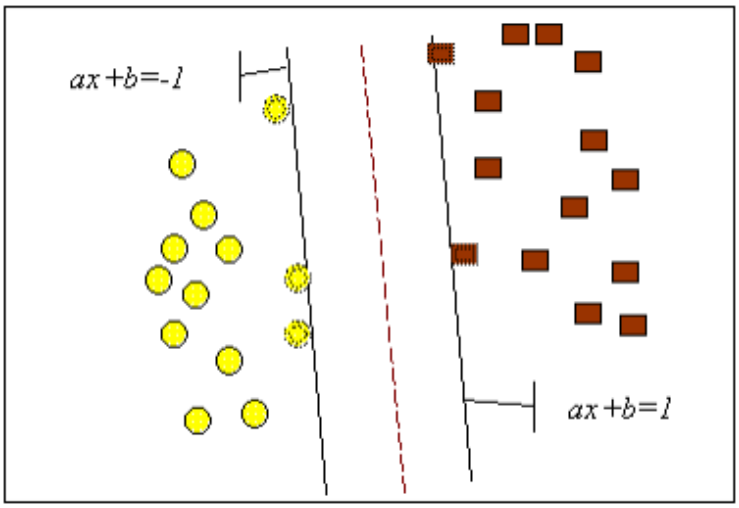


Рисунок 3.4 - Лінійний SVM [12]

Найкращою функцією класифікації є функція, для якої очікуваний ризик мінімальний. Поняття очікуваного ризику в даному випадку означає очікуваний рівень помилки класифікації.

Безпосередньо оцінити очікуваний рівень помилки побудованої моделі неможливо, це можна зробити за допомогою поняття емпіричного ризику. Однак слід враховувати, що мінімізація останнього не завжди призводить до мінімізації очікуваного ризику. Цю обставину слід пам'ятати при роботі з відносно невеликими наборами тренувальних даних.

Емпіричний ризик - рівень помилки класифікації на тренувальному наборі.

Таким чином, в результаті рішення задачі методом опорних векторів для лінійно поділюваних даних ми отримуємо функцію класифікації, яка мінімізує верхню оцінку очікуваного ризику. Однією з проблем, пов'язаних з вирішенням завдань класифікації розглядаються методом, є та обставина, що не завжди можна легко знайти лінійну кордон між двома класами.

У таких випадках один з варіантів - збільшення розмірності, тобто перенесення даних з площини в тривимірний простір, де можливо побудувати таку площину, яка ідеально розділить безліч зразків на два класи. Опорними векторами в цьому випадку слугуватимуть об'єкти з обох класів, які є екстремальними.

Таким чином, за допомогою додавання так званого оператора ядра і додаткових розмірностей, знаходяться кордону між класами у вигляді гіперплоскостей.

Складність побудови SVM-моделі полягає в тому, що чим вище розмірність простору, тим складніше з ним працювати. Один з варіантів роботи з даними високої розмірності - це попереднє застосування будь-якого методу зниження розмірності даних для виявлення найбільш істотних компонент, а потім використання методу опорних векторів.

Як і будь-який інший метод, метод SVM має свої сильні і слабкі сторони, які слід враховувати при виборі даного методу.

Недолік методу полягає в тому, що для класифікації використовується не все безліч зразків, а лише їх невелика частина, яка знаходиться на кордонах.

Гідність методу полягає в тому, що для класифікації методом опорних векторів, на відміну від більшості інших методів, досить невеликого набору даних. При правильній роботі моделі, побудованої на тестовому безлічі, цілком можливо застосування даного методу на реальних даних.

Метод опорних векторів дозволяє :

• отримати функцію класифікації з мінімальною верхньою оцінкою очікуваного ризику (рівня помилки класифікації);

• використовувати лінійний класифікатор для роботи з нелінійно розділяються даними, поєднуючи простоту з ефективністю [12].

3.3 Розробка функціональної моделі у Weka

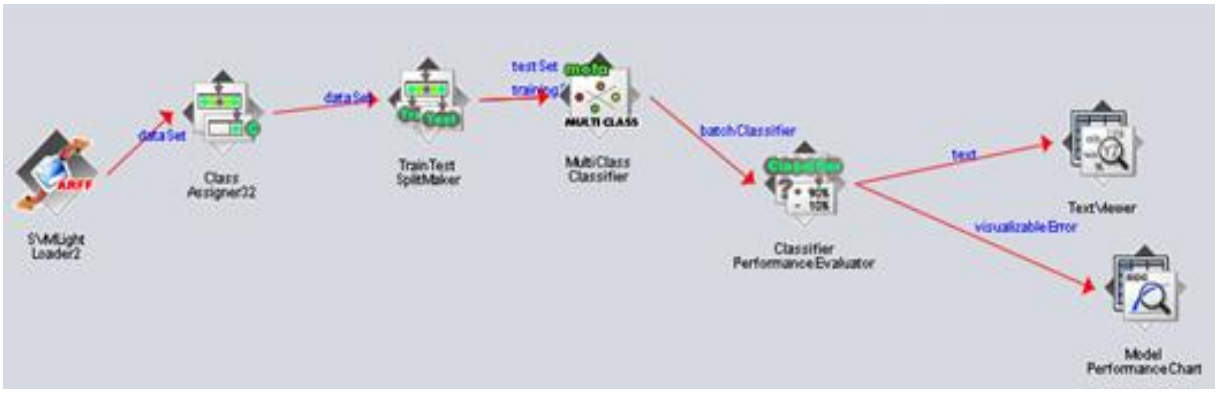


Рисунок 3.5 – Функціональна модель у Weka

3.4 Розробка функціональної моделі у ClowdFlows

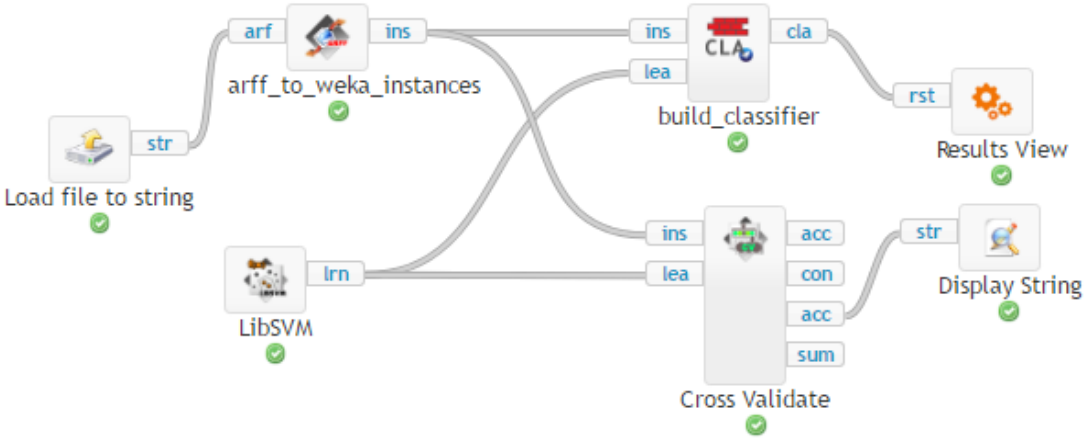


Рисунок 3.6 – Функціональна модель у ClowdFlows

3.5 Результати аналізу

В таблиці 3.2 ми бачимо скорельовану кількість зафіксованих нападів за 2 тижні відвідувань пацієнтами клініки, які вживали та відповідно не вживали наркотичну речовину. Додаткова інформація та статистика нападів зображена в табл. 3.3-3.4.

Таблиця 3.2 - Сумарна статистика за 2 тижні (4 відвідування)

	Placebo ($M_1 = 28$)					Progabide ($M_2 = 31$)				
Visit	$\bar{Y}$ s^2	Correlations				$\bar{Y}$ s^2	Correlations			
1	9.36 102.76	1.00				8.58 332.72	1.00			
2	8.29 66.66	.78	1.00			8.42 140.65	.91	1.00		
3	8.79 215.29	.51	.66	1.00		8.13 193.05	.91	.92	1.00	
4	7.96 58.18	.67	.78	.68	1.00	6.71 126.88	.97	.95	.95	1.00

Таблиця 3.3 –Незалежні змінні та дисперсія параметрів оцінки моделі

Model	11	12	13	21	22	23	41
Int	−2.695	−1.456	−2.753	−1.590	−1.350	−1.735	−1.456
	(.902)	(.933)	(.907)	(.907)	(.904)	(.915)	(.927)
Base	.933	.870	.922	.892	.877	.888	.896
	(.087)	(.105)	(.085)	(.107)	(.103)	(.098)	(.123)
Trt	−1.439	−.987	−1.532	−.983	−.958	−1.063	−.907
	(.418)	(.423)	(.419)	(.412)	(.390)	(.418)	(.412)
Base.Trt	.595	.372	.629	.375	.352	.401	.351
	(.171)	(.209)	(.172)	(.203)	(.196)	(.206)	(.204)
Age	.895	.567	.923	.589	.531	.639	.542
	(.264)	(.272)	(.267)	(.264)	(.266)	(.268)	(.266)
Visit_4	−.168	−.170	−.170	−.156	−.159	−.155	−.150
	(.065)	(.070)	(.066)	(.077)	(.072)	(.077)	(.080)
$\hat{\alpha}_1$	3.641	1.936	3.575	.372	.148	.364	.452
	(.924)	(.126)	(.887)	(.151)	(.138)	(.143)	(.139)
$\hat{\alpha}_2$	4.768	2.769	4.846	.638	.383	.664	
	(1.558)	(.196)	(1.629)	(.276)	(.249)	(.296)	
$\hat{\alpha}_3$	8.250	5.749	8.255	.777	.546	.786	
	(3.656)	(.423)	(3.682)	(.310)	(.298)	(.319)	
$\hat{\alpha}_4$	2.456	1.000	2.479	.218	.000	.224	
	(.443)	(.079)	(.447)	(.080)	(.078)	(.081)	
$\hat{\alpha}_0$		.225	.348		.227	.334	
		(.056)	(.070)		(.055)	(.069)	
log(H)	26.98	47.55	32.29	41.46	47.96	46.75	29.11

Таблиця 3.4 – Статистика нападів за два тижні пацієнтів, які вживали препарат прогабід.

Visit	$\bar{Y}$ s^2	Correlations			
1	5.47				
	33.22	1.00			
2	6.53				
	31.43	.45	1.00		
3	6.00				
	54.34	.63	.70	1.00	
4	4.83				
	18.35	.77	.72	.83	1.00

Тепер є можливість порівняти кількість реалних нападів із очікуваними протягом 4 тижнів відвідування клініки рис. 3.7.

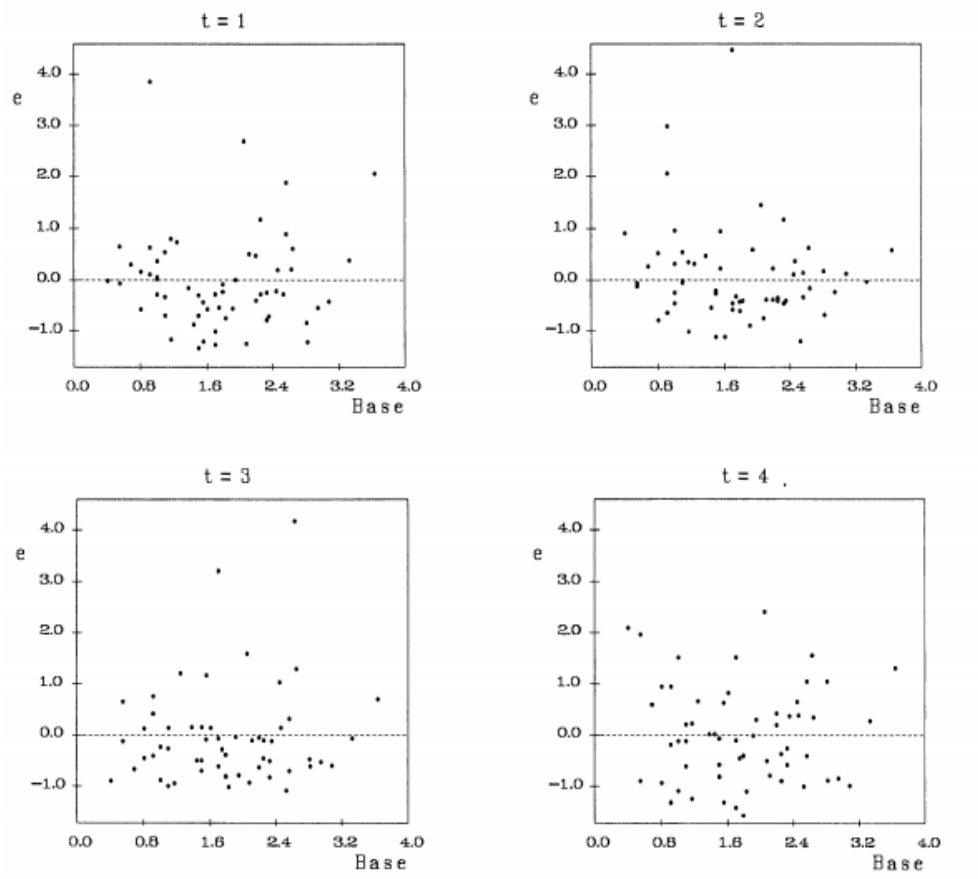
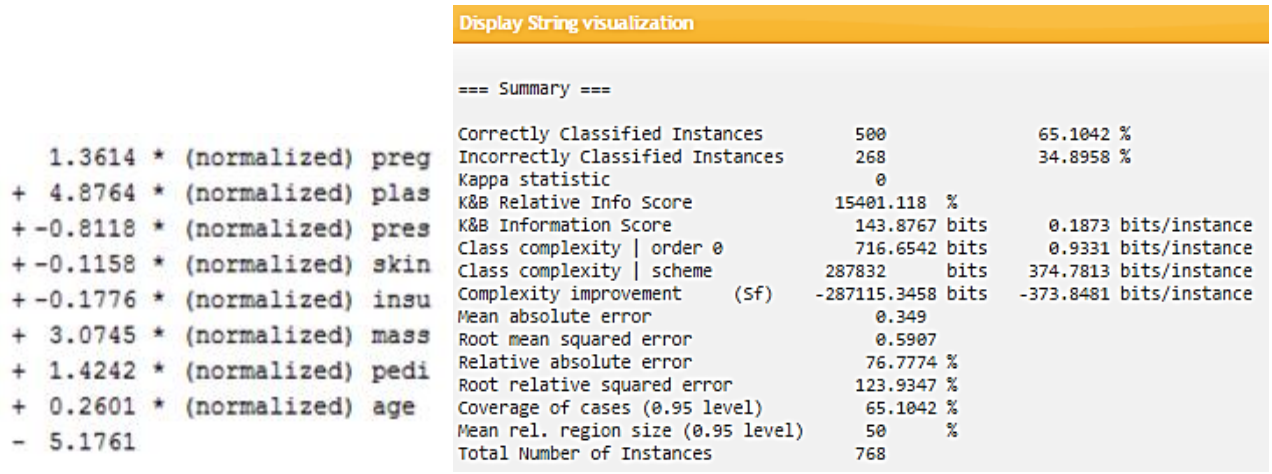


Рисунок 3.7 – Класифікація вхідних даних

Classifier output

```
Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         500               65.1042 %
Incorrectly Classified Instances       268               34.8958 %
Kappa statistic                          0
Mean absolute error                      0.4545
Root mean squared error                  0.4766
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              768

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000     1.000     0.651       1.000   0.789      0.000    0.497     0.650     tested_negative
                 0.000     0.000     0.000       0.000   0.000      0.000    0.497     0.348     tested_positive
Weighted Avg.    0.651     0.651     0.424       0.651   0.513      0.000    0.497     0.544

=== Confusion Matrix ===

   a   b   <-- classified as
 500   0 |   a = tested_negative
 268   0 |   b = tested_positive
```

Рисунок 3.8 – Результати у Weka



```
   1.3614 * (normalized) preg
+  4.8764 * (normalized) plas
+ -0.8118 * (normalized) pres
+ -0.1158 * (normalized) skin
+ -0.1776 * (normalized) insu
+  3.0745 * (normalized) mass
+  1.4242 * (normalized) pedi
+  0.2601 * (normalized) age
-  5.1761
```

Display String visualization

```
=== Summary ===

Correctly Classified Instances         500               65.1042 %
Incorrectly Classified Instances       268               34.8958 %
Kappa statistic                          0
K&B Relative Info Score              15401.118  %
K&B Information Score                  143.8767 bits      0.1873 bits/instance
Class complexity | order 0             716.6542 bits      0.9331 bits/instance
Class complexity | scheme           287832      bits    374.7813 bits/instance
Complexity improvement     (Sf)    -287115.3458 bits   -373.8481 bits/instance
Mean absolute error                      0.349
Root mean squared error                  0.5907
Relative absolute error                 76.7774 %
Root relative squared error            123.9347 %
Coverage of cases (0.95 level)          65.1042 %
Mean rel. region size (0.95 level)      50      %
Total Number of Instances              768
```

Рисунок 3.9 – Результати у ClowdFlows

3.6 Висновки

Після проведених дослідів, можна зробити висновок, що обидві системи однаково зручні для користування та схожі за своїм функціоналом. Результати дослідів в Weka та ClowdFlows показали подібний результат класифікації, головною відмінністю є похибки, оскільки зачення абсолютної похибки є більшим у монолітній системі Weka, а значення квадратичної похибки є більшим у Веб-СОА ClowdFlows.

4 ФУНКЦІОНАЛЬНО-ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ

У даному розділі проводиться оцінка роботи з системами інтелектуального аналізу даних, з подальшою розробкою функціональної моделі. Функціональна модель, розроблена в системах Weka та ClowdFlows, призначених для інтелектуального аналізу даних з метою попередження нападів епілепсії.

Нижче наведено аналіз реалізації модулю з метою вибору оптимальної, з огляду при цьому як на економічні фактори, так і на характеристики продукту, що впливають на продуктивність роботи і на його сумісність з апаратним забезпеченням. Для цього було використано апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) – це технологія, яка дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. Як прямі, так і побічні витрати розподіляються по продуктам та послугам у залежності від потрібних на кожному етапі виробництва обсягів ресурсів. Виконані на цих етапах дії у контексті метода ФВА називаються функціями.

Мета ФВА полягає у забезпеченні правильного розподілу ресурсів, виділених на виробництво продукції або надання послуг, на прямі та непрямі витрати. У даному випадку – аналізу функцій програмного продукту й виявлення усіх витрат на реалізацію цих функцій.

Фактично цей метод працює за таким алгоритмом: визначається послідовність функцій, необхідних для виробництва продукту. Спочатку – всі можливі, потім вони розподіляються по двом групам: ті, що впливають на вартість продукту і ті, що не впливають. На цьому ж етапі оптимізується сама послідовність скороченням кроків, що не впливають на цінність і відповідно витрат. для кожної функції визначаються повні річні витрати й кількість робочих часів, для кожної функції на основі оцінок попереднього пункту визначається

кількісна характеристика джерел витрат. після того, як для кожної функції будуть визначені їх джерела витрат, проводиться кінцевий розрахунок витрат на виробництво продукту.

4.1 Постановка задачі техніко-економічного аналізу

У роботі застосовується метод ФВА для проведення техніко-економічний аналізу розробки. Відповідно цьому варто обирати і систему показників якості програмного продукту.

Технічні вимоги до продукту наступні:

- програмний продукт повинен функціонувати на персональних комп’ютерах із стандартним набором компонент;
- забезпечувати зручність і простоту взаємодії з користувачем або з адміністратором програмного забезпечення;
- передбачати мінімальні витрати на впровадження програмного продукту.

4.1.1 Обґрунтування функцій програмного продукту

Головна функція F0 – обрання програмних продуктів для подальшої модернізації сайту "Інституту фізики та біофізики" НАН України». Виходячи з конкретної мети, можна виділити наступні основні функції ПП:

F_1 – вибір програмного забезпечення;

F_2 –системи інтелектуального аналізу даних;

F_3 – вибір алгоритму обробки даних.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) реалізація на платформі Windows ;

б) реалізація на платформі Linux;

Функція F_2:

а) система інтелектуального аналізу даних Weka;

б) система інтелектуального аналізу даних ClowdFlows;

Функція F_3:

F_2 –системи інтелектуального аналізу даних;

F_3 – вибір алгоритму обробки даних.

Кожна з основних функцій може мати декілька варіантів реалізації.

Функція F_1:

а) реалізація на платформі Windows ;

б) реалізація на платформі Linux;

Функція F_2:

а) система інтелектуального аналізу даних Weka;

б) система інтелектуального аналізу даних ClowdFlows;

Функція F_3:

а) алгоритм kNN;

б) алгоритм SVM.

4.1.2 Варіанти реалізації основних функцій

Варіанти реалізації основних функцій наведені у морфологічній карті системи (рис. 4.1). На основі цієї карти побудовано позитивно-негативну матрицю варіантів основних функцій (таблиця 4.1).

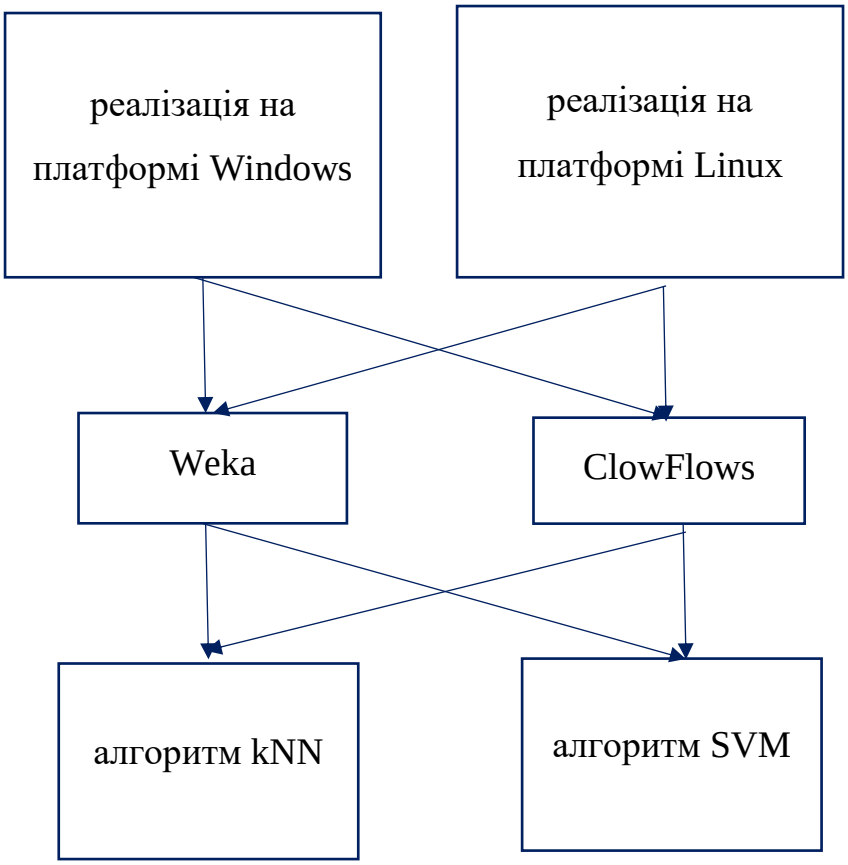


Рисунок 4.1 – Морфологічна карта

Морфологічна карта відображує всі можливі комбінації варіантів реалізації функцій, які складають повну множину варіантів ПП.

Таблиця 4.1 – Позитивно-негативна матриця

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	A	Простота в роботі та налаштуванні програмних засобів для інтелектуального аналізу даних	Менша різноманітність функціональних компонентів

Таблиця 4.1 – Позитивно-негативна матриця(продовження)

Основні функції	Варіанти реалізації	Переваги	Недоліки
F1	Б	Велика різноманітність функціональних компонентів, можливість налаштування для задоволення потреб користувача	Менш популярне програмне забезпечення серед звичайних користувачів, через важкість налаштування інструментарію для роботи
F2	А	Займає невеликий об’єм пам’яті, можлива робота без доступу до мережі інтернет	Високий поріг входу для аналізу та налаштування компонентів в Weka4WS,
	Б	Зручний інтерфейс користувача, можливість роботи одразу декількох алгоритмів на різних вузлах	Неможлива робота офлайн
F3	А	Легкий для розуміння та в реалізації	Може бути дуже ресурсозатратним
	Б	Безперервне зменшення емпіричної помилки класифікації і збільшення проміжку	Необхідність вибору ядра і погана інтерпретованість

На основі аналізу позитивно-негативної матриці робимо висновок, що при обранні програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція *F1:*

Оскільки вибір програмного забезпечення є дуже важливим, для зручності користування та відповідного функціоналу, зважаючи на популярність цих платформ, обираємо варіант А.

Функція *F2:*

Обидві системи інтелектуального аналізу даних доволі розвинені та мають багато розширень та доповнень, тому обидва варіанти А і Б гідні розгляду.

Функція *F3:*

У роботі системи інтелектуального аналізу данних важлива швидка обробка вхідних даних, тому обираємо варіант Б.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

1. F1а – F2а – F3б

2. F1а – F2б – F3б

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

4.2 Обґрунтування системи параметрів ПП

4.2.1 Опис параметрів

На підставі даних про основні функції, що повинен мати програмний продукт, вимог до нього, визначаються основні параметри виробу, що будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

– *X1* – швидкодія програмного забезпечення;

– *X2* – об’єм пам’яті для збереження даних;

– *X3* – час обробки даних;

– *X4* – кількість виконуваних операцій.

X1: Відображає швидкодію операцій залежно від обраного програмного забезпечення.

X2: Відображає об’єм пам’яті персонального комп’ютера або сервера необхідний для збереження та обробки даних під час виконання програми.

X3: Відображає час, який витрачається на обробку даних.

X4: Показує можливу кількість одночасно виконуваних операцій.

4.2.2 Кількісна оцінка параметрів

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у таблиці 4.2.

Таблиця 4.2 – Основні параметри ПП

Назва Параметра	**Умовні позначення**	**Одиниці виміру**	**Значення параметра**		
			гірші	**середні**	**кращі**
Швидкодія системи	X1	с	10	6	2
Об’єм пам’яті для збереження даних	X2	мб	1000	7000	15000
Час обробки даних	X3	с	200	120	40
Кількість одначасно виконуваних операцій	X4	шт	1	2	6

За даними таблиці 4.2 будуються графічні характеристики параметрів – рис. 4.2 – рис. 4.5.

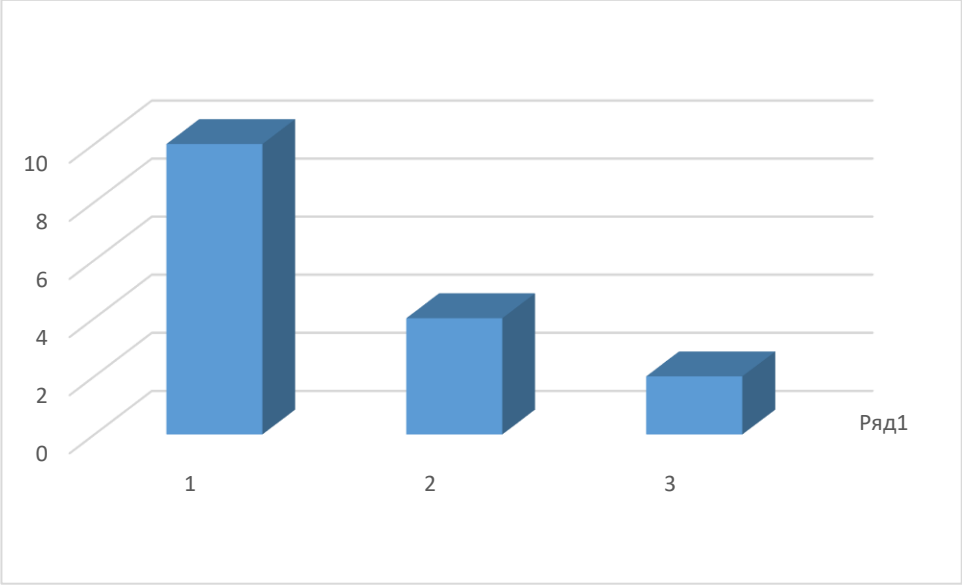


Рисунок 4.2 – X1, Швидкодія системи

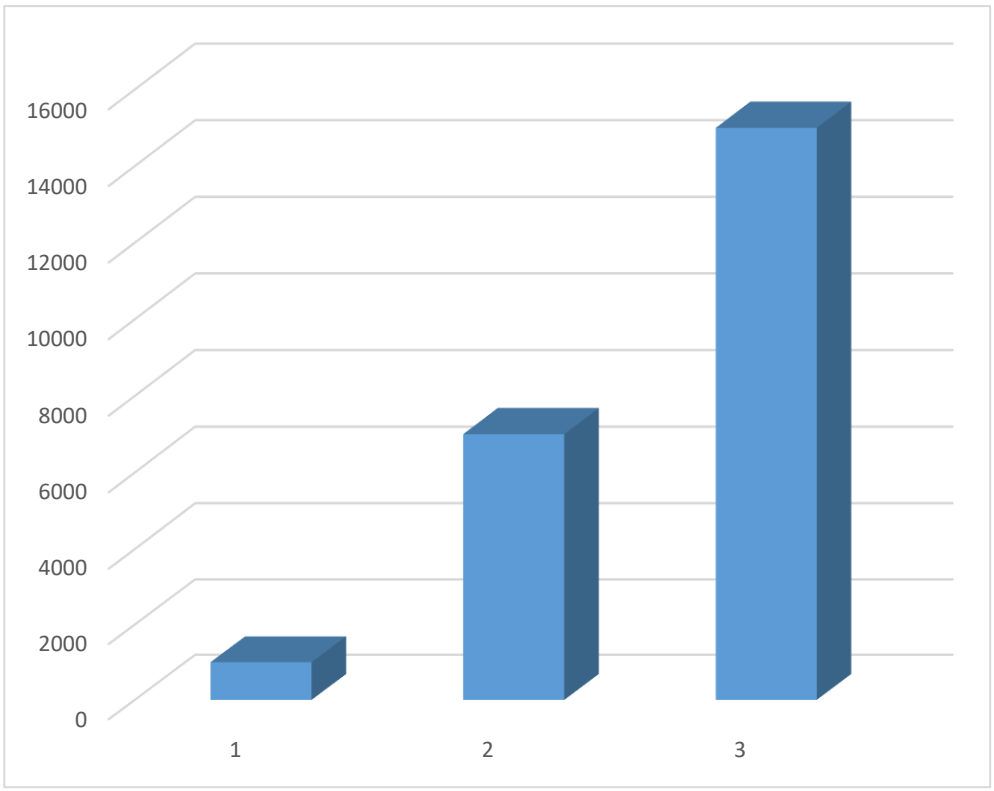


Рисунок 4.3 – X2, Об’єм пам’яті для збереження даних

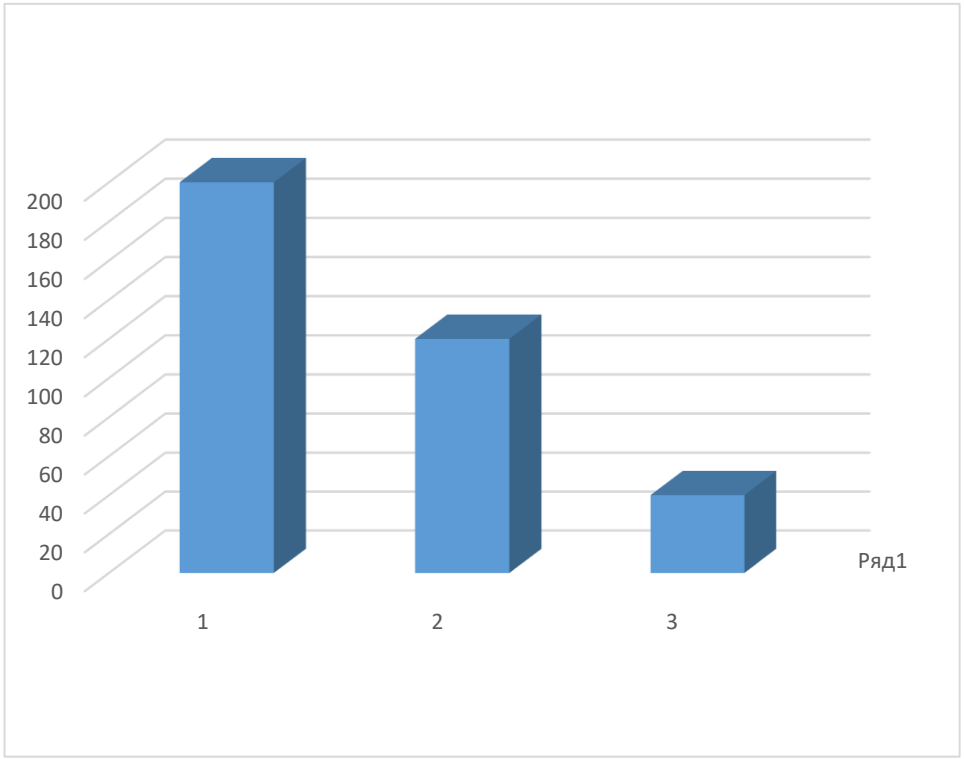
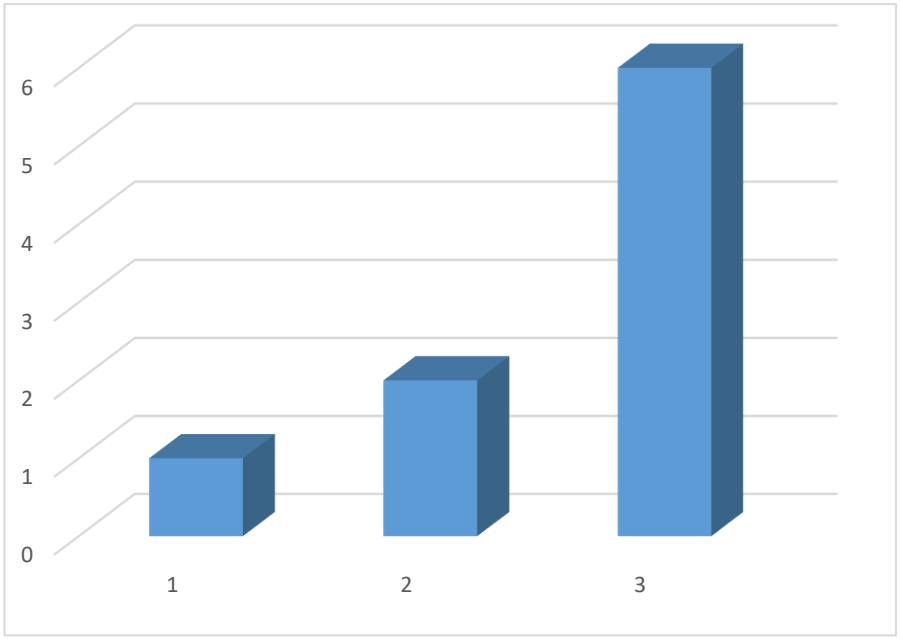


Рисунок 4.4 – X3, Час обробки даних

Рисунок 4.5 – X4, Кількість одначасно виконуваних операцій



Таблиця 4.3 – Результати ранжування параметрів

Позначення параметра	Назва параметра	Одиниці виміру	Ранг параметра за оцінкою експерта							Сума рангів R_i	Відхилення Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
Швидкодія системи	X1	с	2	2	1	2	1	2	3	13	-4,5	20,25
Об’єм пам’яті для збереження даних	X2	мб	1	1	2	1	2	1	1	9	-8, 5	72,25
Час обробки даних	X3	с	4	3	3	4	4	3	2	23	5,5	30,25
Кількість одначасно виконуваних операцій	X4	шт.	3	4	4	3	3	4	4	25	7,5	56,25
	Разом		10	10	10	10	10	10	10	70	0	179

На основі аналізу позитивно-негативної матриці робимо висновок, що при обранні програмного продукту деякі варіанти реалізації функцій варто відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам. Ці варіанти відзначені у морфологічній карті.

Функція *F1:*

Оскільки вибір програмного забезпечення є дуже важливим, для зручності користування та відповідного функціоналу, зважаючи на популярність цих платформ, обираємо варіант А.

Функція *F2:*

Обидві системи інтелектуального аналізу даних доволі розвинені та мають багато розширень та доповнень, тому обидва варіанти А і Б гідні розгляду.

Функція *F3:*

У роботі системи інтелектуального аналізу данних важлива швидка обробка вхідних даних, тому обираємо варіант Б.

Таким чином, будемо розглядати такі варіанти реалізації ПП:

3. F1а – F2а – F3б

4. F1а – F2б – F3б

Для оцінювання якості розглянутих функцій обрана система параметрів, описана нижче.

Таблиця 4.3 – Результати ранжування параметрів

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

а) сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70,$$

де N – число експертів, n – кількість параметрів;

б) середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17{,}5.$$

в) відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T$$

Сума відхилень по всім параметрам повинна дорівнювати 0;

г) загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 179.$$

Порахуємо коефіцієнт узгодженості:

$$W = \frac{12S}{N^2(n^3 - n)} = \frac{12 \cdot 179}{7^2(4^3 - 4)} = 0{,}731 > W_k = 0{,}67$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0,67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 4.4.

Таблиця 4.4 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X1 і X2	>	>	<	>	<	>	>	>	1,5
X1 і X3	<	<	<	<	<	<	>	<	0,5
X1 і X4	<	<	<	<	<	<	<	<	0,5
X2 і X3	<	<	<	<	<	<	<	<	0,5
X2 і X4	<	<	<	<	<	<	<	<	0,5
X3 і X4	>	<	<	>	>	<	<	<	0,5

Числове значення, що визначає ступінь переваги *i*–го параметра над *j*–тим, a_{ij} визначається по формулі:

$$aij = \begin{cases} 1.5 \text{ при } Xi > Xj \\ 1.0 \text{ при Xi} = \text{Xj} \\ 0.5 \text{ при Xi} < Xj \end{cases}$$

З отриманих числових оцінок переваги складемо матрицю A=$\| a_{ij} \|$.

Для кожного параметра зробимо розрахунок вагомості $K_{вi}$ за наступними формулами:

$$K_{\text{вi}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \text{ де } b_i = \sum_{i=1}^{N} a_{ij}.$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%).На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \text{ де } b_i' = \sum_{i=1}^{N} a_{ij} b_j.$$

Як видно з таблиці 4.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 4.5 – Розрахунок вагомості параметрів

Параметриx_i	Параметриx_j				Перша ітер.		Друга ітер.		Третя ітер	
	X1	X2	X3	X4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X1	1,0	1,5	0,5	0,5	3,5	0,219	22,25	0,216	100	0,215
X2	0,5	1,0	0,5	0,5	2,5	0,156	14,25	0,155	64,75	0,154
X3	1,5	1,5	1,0	0,5	4,5	0,281	27,25	0,282	124,25	0,283
X4	1,5	1,5	1,5	1,0	5,5	0,344	34,25	0,347	156	0,348
Всього:					16	1	59	1	445	1

4.3 Аналіз рівня якості варіантів реалізації функцій

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховується так (таблиця 4.6):

$$K_K(j) = \sum_{i=1}^{n} K_{вi,j} B_{i,j},$$

де n – кількість параметрів;$K_{вi}$– коефіцієнт вагомості i–го параметра; B_i – оцінка i–го параметра в балах.

Таблиця 4.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	Варіант реалізації функції	Параметри	Абсолютне значення параметра	Бальна оцінка параметра	Коефіцієнт вагомості параметра	Коефіцієнт рівня якості
F1	А	X1	2	5	0,215	1,075
F2	А	X2	7000	3	0,154	0,462
		X4	2	3	0,348	1,044
	Б	X2	15000	5	0,154	0,77
		X4	6	5	0,348	1,74
F3	А	X3	200	1	0,283	0,283

За даними з таблиці 4.6 за формулою

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \ldots + K_{\text{ТУ}}[F_{zk}],$$

визначаємо рівень якості кожного з варіантів:

K_{K1} = 1,075+ 0,462 + 1,044 + 0,283 = 2,864

K_{K2} = 1,935 + 0,77 + 1,74 + 0,283 = 4,728

Як видно з розрахунків, кращим є четвертий варіант, для якого коефіцієнт технічного рівня має найбільше значення

4.4 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Всі варіанти включають в себе два окремих завдання:

1. Розробка функціональної моделі в системах інтелектуального аналізу Weka та ClowdFlows;

Завдання 1 за ступенем новизни відноситься до групи А. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1.

Для реалізації завдання 1 використовується довідкова інформація.

Проведемо розрахунок норм часу на розробку та програмування завдання.

Загальна трудомісткість обчислюється як

$$T_O = T_P \cdot K_П \cdot K_{СК} \cdot K_M \cdot K_{СТ} \cdot K_{СТ.М}, \qquad (4.1)$$

де T_P – трудомісткість розробки ПП;$K_П$ – поправочний коефіцієнт;$K_{СК}$ – коефіцієнт на складність вхідної інформації; K_M – коефіцієнт рівня мови програмування;$K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм;$K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення

Для даного завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: T_P =90 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.7. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, врахуємо це

за допомогою коефіцієнта $К_{СТ}$ = 0.8. Тоді, загальна трудомісткість програмування першого завдання дорівнює:

Т = 90·1.7·0.8 = 122.4 людино-днів.

Проведемо аналогічні розрахунки для подальших завдань.

Складаємо трудомісткість відповідного завдання для кожного з обраних варіантів реалізації програми:

Т = (122.4 + 19.44 + 6.91 + 19.44) · 8 = 1345.52 людино-годин;

В розробці беруть участь два аналітики області інтелектуального аналізу даних з окладом 8000 грн., один фінансовий аналітик з окладом 12000грн. Визначимо зарплату за годину за формулою:

$$С_Ч = \frac{М}{T_m \cdot t} \text{ грн.},$$

де М – місячний оклад працівників;T_m – кількість робочих днів тиждень;t – кількість робочих годин в день.

$$С_Ч = \frac{8000 + 8000 + 12000}{3 \cdot 21 \cdot 8} = 55{,}56 \text{ грн.}$$

Тоді, розрахуємо заробітну плату за формулою

$С_{ЗП} = С_Ч \cdot Т_i \cdot К_Д$,

де $С_Ч$– величина погодинної оплати праці програміста;$Т_i$ – трудомісткість відповідного завдання; $К_Д$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників становить:

$С_{ЗП}$ = 55,56· 1345.52 · 1.2 = 89708,51 грн

Відрахування на єдиний соціальний внесок становить 22%:

$С_{ВІД} = С_{ЗП} \cdot 0.22$ = 89708,51 · 0.22 = 32985,82 грн.

Тепер визначимо витрати на оплату однієї машино-години. ($С_М$)

Так як одна ЕОМ обслуговує одного аналітика з окладом 8000 грн., з коефіцієнтом зайнятості 0,2 то для однієї машини отримаємо:

$С_Г = 12 \cdot М \cdot К_З = 12 \cdot 8000 \cdot 0,2 = 19200$ грн.

З урахуванням додаткової заробітної плати:

$С_{ЗП} = С_Г \cdot (1 + К_З) = 19200 \cdot (1 + 0.2) = 23040$ грн.

Відрахування на єдиний соціальний внесок:

$С_{ВІД} = С_{ЗП} \cdot 0.22 = 23040 \cdot 0,22 = 8471,81$ грн.

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 27000 грн.

$С_А = К_{ТМ} \cdot К_А \cdot Ц_{ПР} = 1.15 \cdot 0.25 \cdot 27000 = 7762,5$ грн.,

де $К_{ТМ}$– коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $К_А$– річна норма амортизації; $Ц_{ПР}$– договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$С_Р = К_{ТМ} \cdot Ц_{ПР} \cdot К_Р = 1.15 \cdot 27000 \cdot 0.05 = 1552,5$ грн.,

де $К_Р$– відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$Т_{ЕФ} = (Д_К – Д_В – Д_С – Д_Р) \cdot t_З \cdot К_В = (365 – 104 – 8 – 16) \cdot 8 \cdot 0.9 = 1706.4$ годин,

де $Д_К$ – календарна кількість днів у році; $Д_В$, $Д_С$ – відповідно кількість вихідних та святкових днів; $Д_Р$ – кількість днів планових ремонтів устаткування; t –кількість робочих годин в день; $К_В$– коефіцієнт використання приладу у часі протягом зміни. Витрати на оплату електроенергії розраховуємо за формулою:

$С_{ЕЛ} = Т_{ЕФ} \cdot N_С \cdot Ц_{ЕН} = 1706.4 \cdot 0.518 \cdot 1,94 = 1541,55$ грн.,

де N_C – середньо-споживча потужність приладу; $К_З$– коефіцієнтом зайнятості приладу; $Ц_{ЕН}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$С_Н = Ц_{ПР} \cdot 0.67 = 27000 \cdot 0,67 = 18090$ грн.

Тоді, річні експлуатаційні витрати будуть:

$$С_{ЕКС} = С_{ЗП} + С_{ВІД} + С_А + С_Р + С_{ЕЛ} + С_Н$$

$С_{ЕКС} = 23040 + 8471,81 + 7762,5 + 1552,5 + 1541,55 + 18090 = 60458,36$ грн.

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$С_{М-Г} = С_{ЕКС} / Т_{ЕФ} = 60458,36 / 1706,4 = 35,43$ грн/час.

Оскільки в даному випадку всі роботи, які пов‘язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$С_М = С_{М-Г} \cdot Т$

$С_М = 35,43 * 1345,52 = 47671,77$

Накладні витрати складають 67% від заробітної плати:

$С_Н = С_{ЗП} \cdot 0,67$

$С_Н = 89708,51 * 0.67 = 60104,7$ грн.;

Отже, вартість розробки ПП за варіантами становить:

$С_{ПП} = С_{ЗП} + С_{ВІД} + С_М + С_Н$

$С_{ПП} = 89708,51 + 32985,82 + 47671,77 + 60104,7 = 230470,8$ грн.;

4.5 Вибір кращого варіанта техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$К_{ТЕРj} = К_{Кj} / С_{Фj}, \quad (4.2)$$

$К_{ТЕР1} = 2,864 / 230470,8 = 1,24 \cdot 10^{-5}$;

$К_{ТЕР2} = 4,728 / 230470,8 = 2,05 \cdot 10^{-5}$;

Як бачимо, найбільш ефективним є другий варіант варіант реалізації програми з коефіцієнтом техніко-економічного рівня $К_{ТЕР1} = 2,05 \cdot 10^{-5}$.

4.5 Висновки до розділу

В даному розділі проведено повний функціонально-вартісний аналіз програмного забезпечення, який було розроблено в рамках дипломного проекту. Процес аналізу можна умовно розділити на дві частини.

У першій проведено дослідження програмного продукт з технічної точки зору: були поставлені основні параметри, що повинні бути головними при обранні кращої реалізації. На основі отриманих значень параметрів, оцінок експертів було обчислено коефіцієнт технічного рівня, який надалі у другій частині допоміг обрати найкращий варіант з техніко-економічної точки зору.

У другій частині виконувалося економічне обґрунтування альтернативних варіантів реалізації. Порівняння робились з урахуванням витрат на заробітні плати, електроенергії, накладні витрати.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється. Після проведення першої частини аналізу було виявлено, що перший варіант є найбільш оптимальним для реалізації. Його показник техніко-економічного рівня якості $К_{ТЕР1} = 2,05 \cdot 10^{-5}$;

Даний варіант виконання програмного комплексу відповідає усім поставленим вимогам та може бути розроблений відносно швидко.

ВИСНОВКИ

В даній роботі було розглянуто основні алгоритми та детектори виявлення нападів, а також приклади рішень для персонального виявлення нападів на основі ЕЕГ-ЕКГ.

У другому розділі було проведено аналіз систем інтелектуального аналізу даних, як монолітних, так і тих, що побудовані як Веб-СОА. На основі аналізу було складено порівняльну характеристику і обрано найзручнішу систему для проведення аналізу датасету із статистикою нападів хворих пацієнтів.

У третьому розділі було побудовано функціональні моделі у найзручніших двох системах ІАД(монолітній Weka та з використанням веб-сервісів ClowdFlows) і проведено аналіз на основі даних 238 пацієнтів в залежності від вживаних препаратів.

Створені функціональні моделі можна використовувати у подальшому і проводити аналіз даних з іншими даними. Наприклад, для обрання найкращого шляху лікування чи попередження приступів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Ali Hossam Shoeb. Application of Machine Learning to Epileptic: Seizure Onset Detection and Treatment/ Ali Hossam Shoeb// Massachusetts Institute of Technology 2009.— P.27-73.

2. Medvedev Viktor. Cloud Technologies: A New Level for Big Data Mining / Viktor Medvedev, Olga Kurasova // Computer Communications and Networks. – Springer, 2016. – P. 55-67.

3. Talia D. The Weka4WS framework for distributed data mining in service-oriented Grids / D. Talia, P. Trunfio, O. Verta // Concurrency and Computation. - Vol. 20, No. 16. - Wiley, 2008. - P. 1933-1951.

4. Lackovic M. A Framework for Composing: Knowledge Discovery Workflows in Grids/ M. Lackovic, D. Talia, and P. Trunfio// Foundations of Comput. Intel. – Springer, 2009. – 345-362.

5. Podpecan V. Orange4WS Environment for Service-Oriented Data Mining / Vid Podpecan, Monika Zemenova, Nada Lavrac // The Computer Journal. – Vol. 55, No. 1. – Oxford Press, 2012. – P. 82-98.

6. Berthold M. KNIME: The Konstanz Information Miner / Michael R. Berthold et al. // Data Analysis, Machine Learning and Applications. - Springer, 2008. - P. 319-326.

7. Gula M. Matlab Adapter – Online Access to Matlab/Simulink Based on REST Web Services / M. Gula , K. Zakova // Intelligent Systems in Cybernet. and Automat. Theory. - Vol. 348. - Springer, 2015. - P. 199-206.

8. Kranjc J. ClowdFlows: A Cloud Based Scientific Workflow Platform / J. Kranjc, V. Podpecan, N. Lavrac // Lecture Notes in Computer Science. - Vol. 7524. - Springer, 2012. - P. 816-819.

9. Brescia M. DAME: A Distributed Data Mining and Exploration Framework Within the Virtual Observatory / M. Brescia et al. // Remote Instrumentation for eScience and Related Aspects. - Springer, 2012. - P. 267-284.

10. Офіційний сайт Datareview – Порівнялний аналіз систем інтелектуалного аналізу даних [Електронний ресурс] – Режим доступу: http://datareview.info/article/5-instrumentov-data-mining-sravnitelnyiy-analiz/ – дата доступу: 20.05.2017

11. Офіційний сайт Floppyy – Інтелектуалний аналіз з Weka [Електронний ресурс] – Режим доступу: http://floppyy.ru/2016/02/05/weka/ – дата доступу: 1.06.2017

12. Kranjc J. ClowdFlows: A Cloud Based Scientific Workflow Platform / J. Kranjc, V. Podpecan, N. Lavrac // Lecture Notes in Computer Science. - Vol. 7524. - Springer, 2012. - P. 15-18.