Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

Магістр
(назва освітнього ступеня)

на тему: Методи та засоби інтелектуального аналізу динаміки захворювань за даними електронної медичної документації

Виконав: студент VI курсу, групи СНнм-61
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

_____ Шабля Р. А.
(підпис) (прізвище та ініціали)

Керівник _____ Фриз М. Є.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ _____
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль
2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
« 13 » квітня 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Шаблі Руслану Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Методи та засоби інтелектуального аналізу динаміки захворювань за даними електронної медичної документації

Керівник роботи Фриз Михайло Євгенович, к.т.н., доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 10 » березня 2026 року № 4/9-150

2. Термін подання студентом завершеної роботи

3. Вихідні дані до роботи

4. Зміст роботи (перелік питань, які потрібно розробити)

5. Перелік графічного матеріалу

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Гурик О. Я, к.т.н, доцент каф. МТ		
Безпека в надзвичайних ситуаціях	Теслюк В.М., проректор з адміністративно-господарської роботи та будівництва		

7. Дата видачі завдання 13 квітня 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	13.04.2026	*Виконано*
2.	Підбір та опрацювання наукових публікацій, збір	13.04.2026-20.04.2026	*Виконано*
	даних по темі роботи		
3.	Виконання дослідження згідно теми кваліфікаційної	21.04.2026-03.05.2026	*Виконано*
	роботи		
4.	Оформлення розділу «Теоретичні основи та аналіз	04.05.2026-10.05.2026	*Виконано*
	методів інтелектуального аналізу динаміки захворювань»		
5.	Оформлення розділу «Моделювання та проектування	04.05.2026-10.05.2026	*Виконано*
	архітектури інтелектуальної системи»		
6.	Оформлення розділу «Програмна реалізація	04.05.2026-10.05.2026	*Виконано*
	інтелектуальної системи прогнозування медичних		
	ризиків»		
7.	Виконання завдання до підрозділу «Охорона праці»	27.04.2026-10.05.2026	*Виконано*
8.	Виконання завдання до підрозділу «Безпека в	27.04.2026-10.05.2026	*Виконано*
	надзвичайних ситуаціях»		
9.	Оформлення кваліфікаційної роботи	11.05.2026-13.05.2026	*Виконано*
10.	Нормоконтроль		
11.	Перевірка на плагіат		
12.	Попередній захист кваліфікаційної роботи		
13.	Захист кваліфікаційної роботи	28.05.2026	

Студент _____ Шабля Р. А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Фриз М. Є.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Методи та засоби інтелектуального аналізу динаміки захворювань за даними електронної медичної документації // Кваліфікаційна робота освітнього ступеня «Магістр» // Шабля Руслан Андрійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СНнм-61 // Тернопіль, 2026 // С. 98, рис. – 20, табл. – 6, додат. – 1, бібліогр. – 60.

Ключові слова: інтелектуальний аналіз даних, прогнозування ризиків, електронна медична документація, градієнтний бустинг, xgboost, машине навчання, node.js, react.js

Кваліфікаційна робота присвячена розробці інтелектуальної системи аналізу динаміки захворювань на основі даних електронної медичної документації із використанням алгоритму XGBoost та сучасних технологій розробки.

У першому розділі розглянуто теоретичні засади інтелектуального аналізу даних у медицині та роль предиктивних моделей у кардіологічному моніторингу. Проаналізовано специфіку медичних даних, зокрема їх гетерогенність та розрідженість, а також обґрунтовано вибір технологічного стеку.

У другому розділі досліджено математичне моделювання на основі градієнтного бустингу. Висвітлено принципи роботи алгоритму XGBoost, підходи до формування вектора ознак, архітектуру трирівневої системи та механізм інтеграції Python-аналітичного ядра із серверною частиною на Node.js.

У третьому розділі наведено практичну реалізацію інтелектуальної системи, що охоплює розробку предиктивного модуля на XGBoost, серверної частини на Node.js, інтеграцію з Python-аналітичним ядром та створення веб-інтерфейсу на React.js.

ANNOTATION

Methods and Tools for Intelligent Analysis of Disease Dynamics Using Electronic Medical Records Data // The educational level "Master" qualification work // Shablia Ruslan // Ternopil Ivan Pulyuy National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, SNnm-61 group // Ternopil, 2026 // P. 98, fig. – 20, tables – 6, annexes – 1, ref. – 60.

Key words: data mining, risk forecasting, electronic medical records, gradient boosting, xgboost, machine learning, node.js, react.js.

The qualification thesis is devoted to the development of an intelligent system for analyzing disease dynamics based on electronic medical records (EMR) using the XGBoost algorithm and modern full-stack development technologies.

The first chapter examines the theoretical foundations of intelligent data analysis in medicine, the concept of P4 medicine, and the role of predictive models in cardiological monitoring. The specifics of medical data, including their heterogeneity and sparsity, are analyzed, and the choice of the technology stack is justified.

The second chapter focuses on mathematical modeling based on gradient boosting. The principles of the XGBoost algorithm, approaches to feature vector formation, the architecture of the three-tier system, and the integration mechanism of the Python analytical core with the Node.js server are described.

The third chapter presents the practical implementation of the intelligent system, including the development of the XGBoost-based predictive module, the Node.js server component, integration with the Python analytical core, and the React.js web interface.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API (англ. Application Programming Interface) – прикладний програмний інтерфейс.

ЕМД – електронна медична документація.

МН – машинне навчання.

JSON (англ. JavaScript Object Notation) – текстовий формат обміну даними, заснований на JavaScript.

XGBoost (англ. eXtreme Gradient Boosting) – бібліотека градієнтного бустингу на деревах рішень, оптимізована для швидкості та продуктивності.

ЗМІСТ

ВСТУП

Актуальність теми. Сфера охорони здоров’я сьогодні перебуває у стані активної цифрової трансформації, що супроводжується стрімким зростанням обсягів електронної медичної документації. Важливого значення набуває розвиток підходів P4-медицини, у межах яких особлива роль відводиться предиктивності та персоналізованому підходу до пацієнта. У контексті кардіологічного моніторингу, який є надзвичайно актуальним через високий рівень серцево-судинної смертності в Україні, виникає необхідність у створенні точних систем раннього виявлення ризиків.

Класичні діагностичні підходи здебільшого базуються на використанні фіксованих клінічних шкал, які не враховують складних нелінійних взаємозв’язків між показниками та динамічних змін стану пацієнта.

Застосування ансамблевих методів машинного навчання, зокрема алгоритмів градієнтного бустингу, у поєднанні із сучасними технологічними підходами відкриває нові можливості для створення доступних і ефективних медичних сервісів.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Магістр» є підвищення точності та оперативності моніторингу динаміки захворювань шляхом розробки інтелектуальної системи аналізу електронної медичної документації на основі алгоритму XGBoost.

Для досягнення цієї мети передбачається виконання таких завдань:

1. Здійснити аналіз сучасного стану та основних викликів у сфері інтелектуальної обробки медичних даних, дослідження математичних принципів роботи алгоритмів градієнтного бустингу в контексті медичної діагностики, а також порівняння існуючих програмних рішень для роботи з ЕМД з подальшим обґрунтуванням вибраного технологічного стеку.
2. Розробити архітектуру системи інтелектуального аналізу динаміки захворювань на основі електронної медичної документації.
3. Реалізувати прототип інтелектуального вебсервісу, а також провести

експериментальне тестування моделі XGBoost на реальних медичних наборах даних для оцінки її ефективності.

Об’єкт дослідження. Процес моніторингу та інтелектуального аналізу динаміки захворювань пацієнтів на основі параметрів електронної медичної документації.

Предмет дослідження. Методи та програмні засоби предиктивної аналітики медичних даних із використанням алгоритму XGBoost та веб-технологій.

Наукова новизна одержаних результатів полягає у створенні інтегрованого підходу до поєднання аналітичного ядра, побудованого на основі градієнтного бустингу, з архітектурою веборієнтованої інформаційної системи. Запропоновано метод обробки розріджених медичних даних у динаміці, який дає змогу підвищити рівень інтерпретованості результатів для лікаря завдяки візуалізації внеску окремих фізіологічних показників у загальний ризик розвитку захворювання.

Практичне значення одержаних результатів. Результати роботи мають прикладний характер і можуть бути впроваджені у медичних установах для автоматизації скринінгу пацієнтів, зниження операційного навантаження на персонал та забезпечення раннього реагування на критичні зміни стану здоров’я.

Апробація результатів магістерської роботи. Основні результати проведених досліджень обговорювались на Міжнародній студентській науково-технічній конференції «Природничі та гуманітарні науки. Актуальні питання»

Публікації. Основні результати кваліфікаційної роботи опубліковано у двох працях конференції (Див. додатки А).

Структура й обсяг кваліфікаційної роботи. Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку літератури з 60 найменувань та 1 додатка. Загальний обсяг кваліфікаційної роботи складає 98 сторінки, з них 89 сторінки основного тексту, який містить 20 рисунків та 6 таблиць.

1 ОГЛЯД МЕТОДІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДИНАМІКИ ЗАХВОРЮВАНЬ

1.1 Сучасний стан проблеми аналізу та прогнозування захворювань у контексті цифровізації медицини

Нинішній етап розвитку інформаційного суспільства характеризується суттєвою зміною підходів до організації охорони здоров’я внаслідок впровадження eHealth. Цифрова трансформація вийшла за межі простого переведення медичних карт у електронний формат і сформувала єдину екосистему, де медичні дані розглядаються як важливий ресурс для підвищення якості клінічних рішень. Основою цих змін став перехід до електронної медичної документації, що забезпечив накопичення довготривалих багатовимірних даних про пацієнтів. Це створює можливості для застосування предиктивної аналітики, але водночас висуває високі вимоги до методів обробки великих обсягів різнорідної інформації [1].

Окремо слід відзначити актуальність інтелектуального моніторингу патологічних процесів у кардіології, оскільки серцево-судинні захворювання залишаються однією з провідних причин смертності у світі. Традиційні методи діагностики, що базуються на періодичних оглядах, не завжди дозволяють виявити приховані зміни в роботі серцево-судинної системи. У таких умовах ключове значення має аналіз часових рядів показників, зокрема артеріального тиску, серцевого ритму та метаболічних маркерів. Саме динамічний аналіз цих даних дозволяє точніше прогнозувати ризик розвитку інфарктів та інсультів [2].

Сучасний розвиток медицини характеризується переходом від реактивного лікування до проактивного управління здоров’ям пацієнта. У цьому контексті використовується концепція P4-медицини, яка включає предиктивність, превентивність, персоналізацію та партисипативність. Предиктивність передбачає використання моделей для оцінки індивідуальних ризиків, превентивність, раннє попередження захворювань, персоналізація, адаптацію

рішень до особливостей пацієнта, а партисипативність, залучення пацієнта до контролю стану за допомогою IoT-рішень та мобільних сервісів [3].

На рисунку 1.1 зображено життєвий цикл медичних даних у концепції P4-медицини.

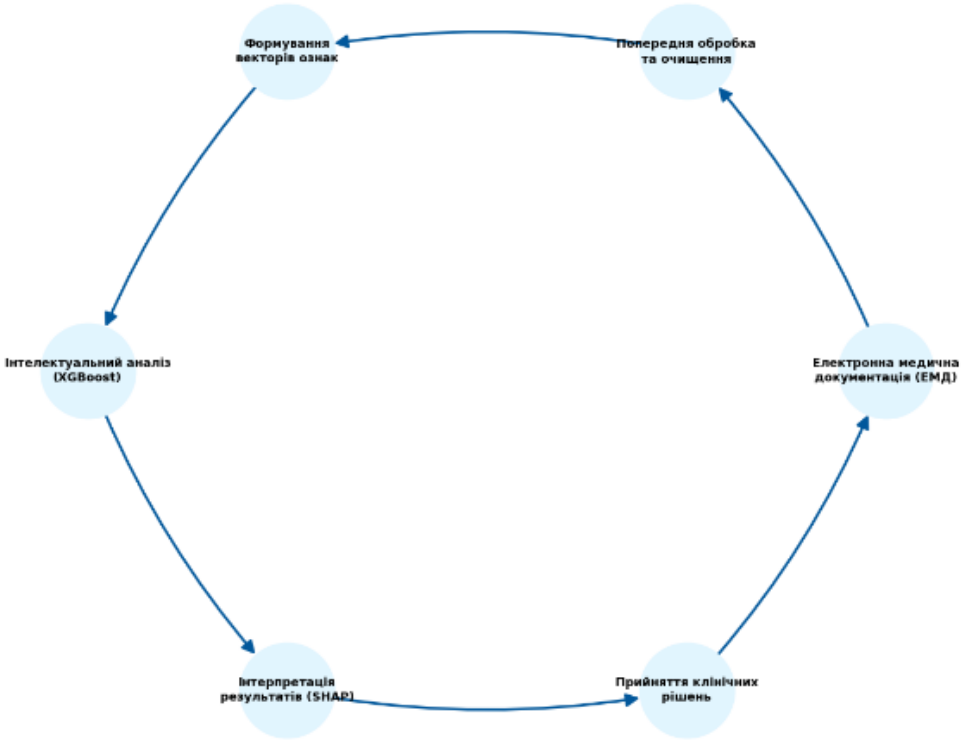

Рисунок 1.1 – Схема життєвого циклу медичних даних

Впровадження зазначеної парадигми значною мірою залежить від застосування методів штучного інтелекту та технологій обробки великих даних. Сучасні медичні інформаційні системи повністю відповідають основним ознакам Big Data, оскільки характеризуються великими обсягами інформації, високою швидкістю її надходження від різних діагностичних пристроїв та різноманітністю форматів представлення даних. При цьому ключовим фактором для медичної сфери є достовірність та якість інформації. У процесі формування електронних записів можливі втрати даних, помилки або викривлення, що зумовлює необхідність впровадження інтелектуальних механізмів контролю ще на етапі їх збору. Подальший аналіз перебігу захворювань дозволяє

трансформувати первинні дані у структуровані знання, які використовуються як основа для прийняття клінічних рішень.

Незважаючи на активний розвиток медичних інформаційних технологій, проблема створення ефективних інструментів прогнозування залишається недостатньо вирішеною. Існуючі програмні рішення часто є складними у впровадженні в повсякденну роботу медичного персоналу або мають обмежену архітектурну гнучкість, що ускладнює їх адаптацію до потреб конкретних медичних закладів. Крім того, класичні статистичні підходи до оцінювання ризиків базуються переважно на статичних моделях і не враховують складні нелінійні залежності та часову динаміку показників. Це підкреслює актуальність розробки нових програмних рішень, які поєднують точність методів машинного навчання з інтуїтивно зрозумілими веб-інтерфейсами для оперативного доступу до результатів аналізу [4].

Соціально-економічна значущість вирішення цієї задачі є досить високою. Використання інтелектуальних систем моніторингу та предиктивної аналітики дозволяє зменшити витрати на стаціонарне лікування та лікування ускладнень, які можна було б попередити за умови раннього виявлення ризиків. З точки зору якості життя пацієнтів такі цифрові інструменти підвищують рівень поінформованості та формують більш відповідальне ставлення до власного здоров’я. Це пов’язано з можливістю візуального відстеження змін у стані організму та розуміння впливу факторів ризику, таких як шкідливі звички або низька фізична активність, на ключові показники здоров’я. Розробка методів автоматизованого аналізу перебігу захворювань на основі електронних медичних даних є актуальним науково-прикладним завданням, що поєднує досягнення медицини, математичної статистики та інформаційних технологій.

Подальший розвиток цієї сфери пов’язаний зі створенням адаптивних веб-платформ, здатних обробляти складні часові ряди у режимі реального часу та водночас забезпечувати високу точність прогнозів і зрозумілу інтерпретацію результатів для лікарів [5].

1.2 Характеристика та систематизація даних електронної медичної документації як об’єкта аналізу

Ефективність застосування інструментів інтелектуального аналізу перебігу захворювань безпосередньо залежить від глибини розуміння структури та особливостей вхідних даних. У процесі кардіологічного моніторингу масиви електронних медичних записів представляють собою складну сукупність взаємопов’язаних показників, які потребують попередньої систематизації перед етапом побудови предиктивних моделей. Медична інформація за своєю природою є неоднорідною, оскільки поєднує числові параметри, описові характеристики та часові ряди, що відображають динаміку функціонування основних фізіологічних систем пацієнта [6]. На рисунку 1.2 зображено класифікацію параметрів ЕМД для інтелектуального аналізу.

Рисунок 1.2 – Класифікація параметрів ЕМД

Фундаментальну групу даних, що використовується для оцінювання кардіологічних ризиків, становлять антропометричні та фізіологічні показники пацієнта. Такі базові характеристики, як маса тіла та зріст, у сучасних аналітичних підходах зазвичай не розглядаються окремо, а використовуються у вигляді похідного інтегрального параметра, індексу маси тіла. Показник має важливе клінічне значення, оскільки пов’язаний із метаболічними порушеннями

та рівнем жирової тканини в організмі, що робить його суттєвим фактором при прогнозуванні розвитку артеріальної гіпертензії. Фізіологічні параметри, зокрема артеріальний тиск, характеризуються значною варіативністю у часі та потребують динамічного аналізу.

При цьому систолічний та діастолічний тиск відображають різні аспекти роботи серцево-судинної системи, силу серцевих скорочень та периферичний судинний опір. У межах інтелектуальних моделей ці показники доцільно розглядати у взаємозв’язку, оскільки їх різниця, тобто пульсовий тиск, часто є більш інформативним індикатором ризику розвитку судинних ускладнень, ніж окремі значення максимального чи мінімального тиску [7]. Самостійну групу параметрів становлять лабораторні показники, серед яких ключове значення мають рівні загального холестерину та глюкози в крові. На відміну від гемодинамічних характеристик, ці біохімічні маркери змінюються повільніше та відображають більш глибокі патологічні процеси, пов’язані зі станом судинної стінки та тканин організму. Підвищений рівень глюкози часто є ознакою порушення інсулінової чутливості, що може призводити до ушкодження ендотелію та формування умов для розвитку гострих кардіологічних ускладнень. У процесі інтелектуального аналізу такі змінні обов’язково підлягають нормалізації, оскільки суттєві відмінності у масштабах значень порівняно з іншими ознаками можуть викликати викривлення результатів моделі [8].

Окрім числових показників, важливу роль відіграють поведінкові та анамнестичні фактори, які здебільшого представлені у вигляді бінарних або категоріальних змінних. До них належать факт тютюнопаління, рівень фізичної активності, спадкова схильність та наявність супутніх захворювань, зокрема цукрового діабету. Такі чинники суттєво впливають на загальний рівень клінічного ризику, часто посилюючи негативний ефект інших параметрів, зокрема артеріальної гіпертензії.

Для використання в інтелектуальних моделях ця інформація потребує попереднього числового кодування, що забезпечує коректну інтерпретацію

алгоритмом внеску кожного фактора у фінальний прогноз [9]. На рисунку 1.3 наведено загальну схему технологічного процесу обробки медичних даних.

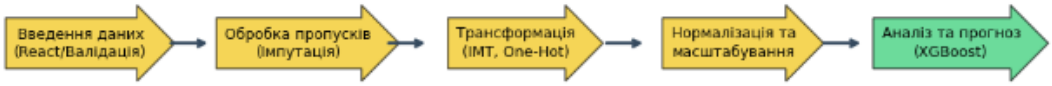

Рисунок 1.3 – Схема обробки медичних даних

Суттєвою проблемою при роботі з медичними даними є їхня нерівномірність у часовому вимірі. Оскільки пацієнт не перебуває під безперервним медичним спостереженням, інформація в електронних системах накопичується фрагментарно, переважно під час візитів до лікаря або проходження лабораторних досліджень. У результаті формуються розріджені часові ряди з великими проміжками відсутніх значень, що ускладнює їх подальший аналіз.

Для коректної роботи аналітичних моделей це потребує застосування методів відновлення пропущених даних або використання спеціалізованих підходів, які враховують нерівномірність часових інтервалів між спостереженнями.

Додатково на етапі збору даних важливе значення має оперативна перевірка їхньої коректності. З огляду на чутливість алгоритмів до шуму та некоректних значень, клієнтська частина системи повинна забезпечувати автоматичну валідацію форматів введення та виявлення аномальних показників, що виходять за межі фізіологічних норм [10].

Окремо слід відзначити проблему надмірної кореляції між ознаками. У медичних даних багато показників є взаємопов’язаними, наприклад маса тіла може корелювати з артеріальним тиском, а рівень глюкози з ліпідним профілем. Тому одним із завдань етапу підготовки даних є виявлення та усунення надлишкових залежностей, що дозволяє зменшити складність моделі, підвищити

її обчислювальну ефективність і покращити інтерпретованість результатів для практикуючого лікаря.

1.3 Огляд наукової літератури та сучасних досліджень у сфері медичного Data Mining

Комплексний аналіз сучасних вітчизняних та зарубіжних наукових публікацій свідчить, що інтелектуальна обробка медичних даних сформувалася як самостійний міждисциплінарний напрям. У дослідженнях останніх років чітко простежується тенденція переходу від простих експертних систем із жорстко заданими правилами до більш гнучких моделей, здатних виявляти складні нелінійні залежності у багатовимірних наборах медичних ознак [11]. У фундаментальних роботах підкреслюється, що основним обмежувальним фактором впровадження штучного інтелекту в клінічну практику є не стільки точність алгоритмів, скільки інтерпретованість отриманих результатів. Особливо це актуально для кардіології, де ціна помилки є критично високою, а отже рішення моделі повинні бути зрозумілими для лікаря. Це стало причиною активного розвитку напрямку пояснюваного штучного інтелекту (XAI). У межах цього підходу дослідники пропонують використовувати інструменти теорії ігор, зокрема значення Шеплі, для декомпозиції прогнозу. Такий метод дозволяє представити загальний ризиковий показник як суму внесків окремих ознак і кількісно оцінити вплив кожного медичного параметра на кінцевий результат [12].

Значна частина сучасних наукових робіт присвячена порівнянню різних архітектур машинного навчання. У дослідницькому середовищі досі ведеться обговорення щодо того, які підходи є ефективнішими, класичні ансамблеві методи чи моделі глибокого навчання. Переважає позиція, що для структурованих табличних даних із електронних медичних карт алгоритми градієнтного бустингу забезпечують найкращий баланс між точністю прогнозу та обчислювальною ефективністю. Додатковою перевагою таких алгоритмів є

краща придатність до інтерпретації результатів порівняно з більш складними нейромережевими моделями. Для медичної практики цей аспект має принципове значення, оскільки лікар повинен не лише отримати прогноз, а й розуміти, які саме показники вплинули на його формування.

У той же час при аналізі процесів, представлених у вигляді часових рядів, у науковій літературі все частіше розглядаються гібридні моделі. Такі підходи передбачають поєднання градієнтного бустингу з рекурентними нейронними мережами або архітектурами довгої короткочасної пам’яті. Подібна інтеграція дає можливість враховувати як поточні, так і історичні показники пацієнта, що сприяє виявленню прихованих негативних змін ще на ранніх, доклінічних етапах розвитку захворювання [13].

Для більш системного розуміння напрямів сучасних досліджень доцільно узагальнити основні методи аналізу, які найчастіше використовуються у профільних наукових публікаціях. В таблиці 1.1 наведено порівняльну характеристику методів інтелектуального аналізу медичних показників.

Таблиця 1.1 – Порівняльна характеристика методів інтелектуального аналізу медичних показників

Категорія методів	**Типові алгоритми**	**Переваги в клінічній практиці**	**Обмеження та наукові виклики**	**Роль у розроблюваній системі**
1	**2**	**3**	**4**	**5**
Класичні статистичні	Логістична регресія	Максимальна прозорість	Нездатність моделювати складні нелінійні патології.	Використовуються для валідації та як базові моделі.
Ансамблеві методи	XGBoost, LightGBM	Найвища точність на даних з пропусками.	Складність ручного налаштування	Основний двигун для розрахунку кардіо-ризиків.

Продовження таблиці 1.1

1	2	3	4	5
Пояснюваний ШІ (XAI)	SHAP, LIME, Anchor	Забезпечують довіру лікаря через візуалізацію причин	Додаткове навантаження на обчислювальну систему	Інструмент візуалізації для інтерфейсу лікаря
Методи кластеризації	K-means, DBSCAN	Виявлення нових підгруп пацієнтів зі схожими симптомами	Складність інтерпретації отриманих груп	Сегментація бази пацієнтів для диспансеризації

Суттєвим напрямом сучасних наукових досліджень є проєктування архітектури програмних систем. У роботах останніх років простежується тенденція поступового переходу від традиційного десктопного програмного забезпечення до веборієнтованих рішень. Зокрема, у медичній сфері доведено ефективність використання сучасних технологічних стеків для створення односторінкових застосунків. Автори підкреслюють, що така архітектура забезпечує режим миттєвого оновлення інтерфейсу: при введенні клінічних показників реактивна клієнтська частина дозволяє оновлювати візуалізацію ризиків без перезавантаження сторінки [14]. Це підвищує зручність роботи користувача та дає змогу виконувати сценарне моделювання, коли медичний фахівець змінює вхідні параметри (наприклад, поведінкові фактори пацієнта) і одразу бачить вплив цих змін на прогнозовані ризики [15].

Узагальнення аналізу наукових джерел показує, що попри значний розвиток окремих алгоритмічних рішень, все ще бракує комплексних систем, які б поєднували збір і перевірку даних, предиктивне моделювання та підтримку прийняття клінічних рішень в єдиній архітектурі. Саме тому актуальність даного дослідження полягає у створенні інтегрованого програмного рішення, що базується на методах градієнтного бустингу та сучасних вебтехнологіях.

1.4 Огляд та порівняльний аналіз існуючих програмних засобів аналізу

На етапі розвитку медичних цифрових технологій програмні засоби для аналізу та прогнозування захворювань пройшли суттєву еволюцію. Їх становлення відбувалося від простих експертних систем, що ґрунтувалися на фіксованих логічних правилах, до складних когнітивних платформ, які використовують методи машинного навчання.

Узагальнений аналіз існуючих рішень дозволяє визначити поточний рівень розвитку галузі та виявити ключові технологічні обмеження, що обумовлюють необхідність створення більш гнучких веборієнтованих систем. На сьогодні на ринку умовно можна виділити три основні групи таких рішень: глобальні аналітичні платформи, національні інформаційні системи та спеціалізовані діагностичні інструменти, кожна з яких відрізняється підходами до організації даних та архітектури [16].

Найбільш технологічно розвиненими вважаються глобальні інтелектуальні платформи, серед яких значне місце займали рішення компанії IBM. Подібні системи базуються на складних алгоритмах когнітивного аналізу та автоматичної обробки текстових даних, що дозволяє виділяти важливі ознаки з неструктурованих медичних записів і наукових публікацій. Основною перевагою таких платформ є здатність інтегрувати різнорідні джерела інформації, включаючи генетичні дані та результати інструментальних досліджень. Водночас практичний досвід їх використання виявив низку суттєвих недоліків, зокрема складність адаптації до локальних медичних стандартів, обмежену прозорість алгоритмів через закритий характер систем, а також високу вартість впровадження та підтримки. У результаті це знижує їхню доступність для медичних закладів регіонального та місцевого рівня [17].

Поряд із глобальними платформами значного розвитку набули національні та регіональні медичні інформаційні системи. Їх архітектура, як правило, орієнтована на централізоване ведення електронної медичної документації та

автоматизацію адміністративних процесів, зокрема реєстрації звернень пацієнтів, обліку наданих медичних послуг і формування електронних призначень. Хоча такі системи накопичують великі обсяги даних про стан здоров'я населення, їх аналітична функціональність зазвичай обмежується стандартними звітами та статистичними зведеннями [18]. Окрему категорію становлять спеціалізовані діагностичні вебсервіси та онлайн-калькулятори оцінки ризиків, які базуються на класичних клінічних шкалах. Такі рішення відзначаються простотою використання та доступністю, однак з точки зору сучасного інтелектуального аналізу їх можливості є обмеженими [19]. Статичні алгоритми, на яких вони побудовані, не враховують складні нелінійні залежності між фізіологічними показниками та не аналізують їх зміну в часі. Крім того, відсутність механізмів збереження історичних даних унеможливлює відстеження динаміки стану пацієнта та автоматичне коригування прогнозів відповідно до змін лікування або способу життя [20]. На рисунку 1.4 зображено гістограму порівняння точності прогнозування існуючих засобів.

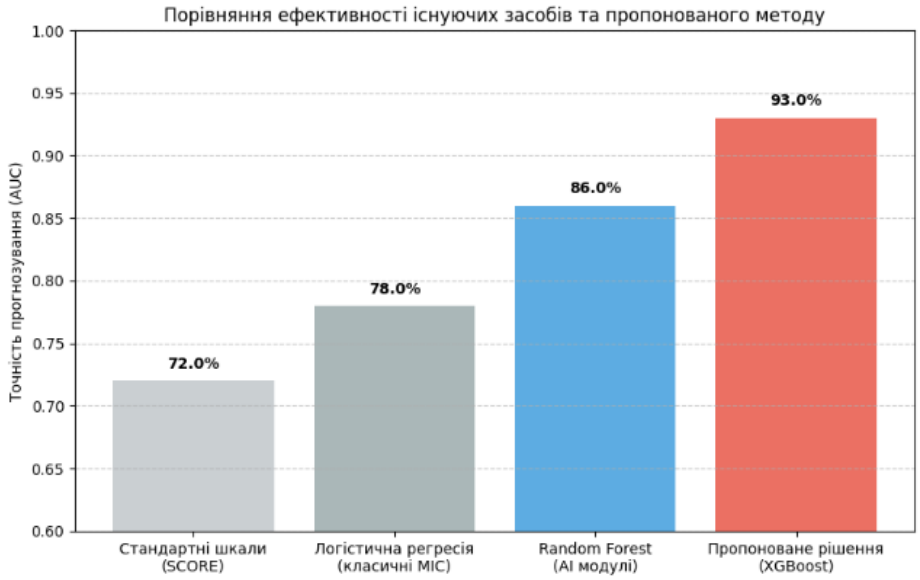

Рисунок 1.4 – Гістограма порівняння точності прогнозування існуючих засобів

Для узагальнення результатів виконаного огляду та аргументації вибору технологічного стеку для розроблюваної системи доцільно здійснити

порівняльний аналіз за основними критеріями, які визначають як ефективність побудови прогностичної моделі, так і практичну зручність її використання в реальних умовах експлуатації. Такий аналіз дозволяє оцінити не лише математичні можливості наявних рішень, а й їхню придатність до впровадження у медичних закладах, де важливими є швидкість роботи, простота інтерфейсу, захист даних та можливість адаптації до конкретних потреб лікаря.

Окрему увагу необхідно приділити тому, чи підтримує програмне рішення роботу з електронною медичною документацією, чи забезпечує збереження історії пацієнта та чи має механізми пояснення отриманих прогнозів.

Для інтелектуальних медичних систем важливим є не тільки факт розрахунку ризику, а й зрозуміле подання результату, оскільки саме інтерпретованість підвищує довіру медичного персоналу до автоматизованого аналізу. В таблиці 1.2 наведена порівняльна характеристика існуючих програмних рішень.

Таблиця 1.2 – Порівняльна характеристика існуючих програмних рішень для аналізу захворювань

Критерій порівняння	**Глобальні AI-платформи**	**Популярні МІС**	**Статичні калькулятори**	**Пропонована система**
1	**2**	**3**	**4**	**5**
Метод аналізу даних	Глибоке навчання	Описова статистика та фільтри	Регресійні статистичні формули	Машинне навчання
Обробка динаміки показників	Висока	Обмежена	Відсутня	Побудова прогнозів
Валідація вхідних даних	Пост-обробка на стороні сервера	Базова перевірка	Відсутня	Миттєва інтелектуальна валідація

Продовження таблиці 1.2

1	2	3	4	5
Архітектурна модель	Хмарна закрита екосистема	Клієнт-серверний моноліт	Односторінкова веб-форма	Мікросервісна веб-платформа
Інтерпретованість результатів	Низька (складна математика)	Відсутня (тільки цифри)	Висока (лінійна залежність)	Середня
Доступність	Дуже низька	Висока	Висока	Висока

Проведений аналіз свідчить про наявність незаповненої технологічної ніші для програмного рішення, яке поєднує високу точність сучасних методів машинного навчання з гнучкістю веборієнтованих платформ. Однією з ключових переваг запропонованого підходу є використання реактивних бібліотек при розробці користувацького інтерфейсу, що забезпечує контроль і валідацію вхідних даних у режимі реального часу. Це дозволяє мінімізувати проблему накопичення некоректної інформації, яка негативно впливає на якість результатів предиктивного аналізу.

Застосування мікросервісної архітектури для організації взаємодії між серверною частиною та аналітичним модулем створює ефективну та масштабовану систему, орієнтовану на інтелектуальну підтримку клінічних рішень у реальному часі, на відміну від великих універсальних медичних платформ. Таким чином, розробка подібного рішення є логічним продовженням розвитку цифрової медицини та спрямована на підвищення якості ранньої діагностики й індивідуалізацію підходів до лікування.

На основі проведеного аналізу сучасного стану проблеми та огляду існуючих програмних рішень можна зробити висновок, що побудова ефективної системи прогнозування захворювань потребує поєднання сучасних методів машинного навчання з гнучкою веборієнтованою архітектурою. Виявлені обмеження наявних систем, зокрема недостатній рівень валідації даних на етапі введення та складність інтерпретації результатів для медичних фахівців,

обґрунтовують необхідність розробки власного підходу. Такий підхід має забезпечувати одночасно якісну обробку вхідних даних через клієнтську частину та застосування потужних математичних моделей для виявлення нелінійних залежностей у динаміці медичних показників пацієнта. Об’єктом дослідження є процеси інтелектуального аналізу та прогнозування змін стану здоров’я на основі електронної медичної документації. Предметом дослідження виступають методи машинного навчання, алгоритми попередньої обробки медичних даних і програмні засоби їх реалізації у вигляді вебплатформи. Основна гіпотеза роботи полягає в тому, що використання градієнтного бустингу разом із інтелектуальною валідацією даних на стороні клієнта дозволяє підвищити точність раннього виявлення серцево-судинних ризиків у порівнянні з класичними статистичними підходами.

Магістерська робота спрямована на проєктування та дослідження інтелектуальної платформи прогнозування захворювань, яка на основі методів машинного навчання автоматизує оцінювання індивідуальних клінічних ризиків, забезпечує візуалізацію динаміки ключових біомаркерів та підтримує процес прийняття лікарських рішень. Реалізація проєкту передбачає створення веборієнтованого програмного забезпечення, що поєднує високу продуктивність обчислень із захищеністю та конфіденційністю медичних даних.

Для досягнення поставленої мети спочатку формується модель багатовимірного представлення ознак на основі електронних медичних записів, яка включає етапи нормалізації даних та обробки пропущених значень. Далі виконується порівняльне дослідження ансамблевих алгоритмів з метою вибору найбільш стабільної моделі, здатної ефективно виявляти клінічні аномалії. Окрему увагу приділено проєктуванню архітектури системи з розподілом функцій між клієнтською частиною та аналітичним ядром. Завершальним етапом є практична перевірка розробленого підходу на реальних наборах даних для підтвердження його ефективності в умовах медичного моніторингу.

1.5 Висновок до першого розділу

У першому розділі кваліфікаційної роботи освітнього рівня «Магістр» розглянуто теоретичні основи та здійснено комплексний аналіз сучасного стану інтелектуальних систем у сфері обробки медичних даних. Показано, що в умовах глобальної цифрової трансформації системи охорони здоров’я та переходу до проактивної моделі медицини методи машинного навчання відіграють ключову роль у виявленні патологій на ранніх стадіях. Аналіз структури електронних медичних записів дав змогу виділити основні проблеми їх використання, серед яких неоднорідність показників, розрідженість даних та складні взаємозв’язки між фізіологічними характеристиками пацієнта і ризиком розвитку захворювань.

Порівняння існуючих програмних рішень і математичних підходів засвідчило, що класичні статистичні моделі оцінювання ризиків мають обмежену ефективність при роботі з динамічними медичними процесами. Натомість доведено переваги ансамблевих методів, зокрема градієнтного бустингу, які завдяки механізмам контролю складності забезпечують високу точність та стабільність прогнозування. У результаті сформовано обґрунтування вибору технологічної основи системи, що поєднує обчислювальні можливості спеціалізованих бібліотек із сучасними вебтехнологіями, створюючи базу для розробки надійної платформи моніторингу стану здоров’я.

2 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ АРХІТЕКТУРИ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ

2.1 Методологічний аналіз стратегій розв’язання задач інтелектуального прогнозування в медицині

Перед розробкою інтелектуальної системи прогнозування захворювань необхідно визначити методологічні підходи, які найбільшою мірою відповідають специфіці електронної медичної документації. Медичні записи, що формуються у реальних клінічних умовах, зазвичай не є повністю однорідними та впорядкованими. Вони можуть містити пропущені значення, шум, різні формати подання інформації, а також складні взаємозв’язки між клінічними, антропометричними та лабораторними показниками.

Тому вибір моделі прогнозування повинен ґрунтуватися не лише на показниках точності, а й на здатності алгоритму працювати з неповними даними, виявляти нелінійні залежності та забезпечувати зрозумілу інтерпретацію результатів для медичного фахівця.

Традиційно у задачах медичної діагностики широко застосовувалися статистичні методи. Перевагою є відносна простота математичного апарату та можливість пояснення отриманих результатів. Одним із найбільш поширених підходів є логістична регресія, яка дозволяє оцінити ймовірність виникнення певного патологічного стану на основі лінійної комбінації вхідних ознак.

Проте ефективність такого підходу суттєво знижується у випадках, коли між медичними параметрами існують складні нелінійні взаємодії. У практиці аналізу біомедичних процесів часто спостерігається ситуація, коли кілька факторів ризику не просто додаються, а взаємно підсилюють один одного, формуючи більш складну картину розвитку захворювання.

Крім того, класичні статистичні моделі висувають досить жорсткі вимоги до структури та якості вибірки. Для коректного застосування таких методів бажаною є відсутність мультиколінеарності, достатня повнота даних та

наближеність ознак до певних розподілів. У реальних медичних інформаційних системах ці умови виконуються не завжди, оскільки дані накопичуються у процесі повсякденної роботи лікаря, а не в межах спеціально організованого експерименту. Саме тому використання лише класичних статистичних підходів не забезпечує достатньої гнучкості при аналізі електронних медичних записів [21].

Подальший розвиток машинного навчання дозволив розширити можливості медичного прогнозування. На відміну від статистичних методів, багато алгоритмів машинного навчання не потребують суворих припущень щодо розподілу даних і здатні адаптуватися до складної структури вибірки. Наприклад, метод найближчих сусідів базується на пошуку подібності між об’єктами у багатовимірному просторі ознак.

Такий підхід дає змогу враховувати складні межі між класами, однак має низку обмежень. Зокрема, він є чутливим до надлишкових або нерелевантних параметрів, а також потребує значних обчислювальних ресурсів при збільшенні обсягу даних.

Більш придатними для медичної діагностики є деревовидні моделі, оскільки вони формують послідовність правил, близьку до логіки клінічного мислення. Наприклад, модель може послідовно аналізувати рівень артеріального тиску, вік пацієнта, індекс маси тіла, рівень глюкози та інші показники, поступово звужуючи область можливих рішень. Такий підхід є зручним для інтерпретації, однак окремі дерева рішень мають істотний недолік, а саме високу схильність до перенавчання.

У випадку надмірного пристосування до навчальної вибірки модель втрачає здатність коректно узагальнювати результати для нових пацієнтів, що стало передумовою розвитку ансамблевих методів, у яких підсумкове рішення формується не однією моделлю, а сукупністю взаємопов’язаних алгоритмів [22].

У межах даного дослідження як основний метод прогнозування обрано градієнтний бустинг. Його принцип полягає у послідовному додаванні слабких моделей, кожна з яких коригує помилки, допущені попередніми етапами

навчання. На відміну від випадкового лісу, де дерева будуються незалежно, бустинг формує ансамбль поступово, з урахуванням залишкових похибок. Завдяки цьому алгоритм концентрується на складних для класифікації випадках і поступово підвищує якість прогнозування.

Для медичних даних така властивість має особливе значення, оскільки клінічні записи часто містять нетипові комбінації симптомів, пропущені значення або суперечливі показники. Важливою перевагою градієнтного бустингу є можливість поєднання високої точності прогнозування з інструментами пояснення результатів. У медичній сфері результат роботи моделі повинен бути не лише формально правильним, а й зрозумілим для лікаря. Саме тому особливого значення набувають методи інтерпретації, зокрема підхід на основі значень Шеплі. Такий підхід дозволяє оцінити внесок кожної ознаки у фінальний прогноз і визначити, які саме клінічні параметри найбільше вплинули на розрахований рівень ризику.

Отже, методологічна основа розроблюваної системи повинна поєднувати кілька взаємодоповнювальних складових. До них належать статистичний аналіз, алгоритми машинного навчання, ансамблеве моделювання та засоби інтерпретації результатів. Таке поєднання дозволяє перейти від простого математичного розрахунку до формування більш змістовної аналітичної оцінки, яка може бути використана у процесі підтримки прийняття клінічних рішень.

Окрім вибору алгоритму, важливим етапом є визначення технологічної основи системи. Програмне рішення для прогнозування захворювань повинно забезпечувати швидку обробку великих обсягів медичної інформації, збереження конфіденційності даних, масштабованість та зручність використання.

У сучасних інтелектуальних системах доцільно виділяти кілька основних рівнів, зокрема рівень збереження та попередньої обробки даних, аналітичний модуль і рівень представлення результатів користувачеві. Кожен із цих рівнів потребує застосування відповідних технологічних засобів. Для реалізації аналітичної підсистеми доцільним є використання мови Python, яка має

розвинену екосистему бібліотек для аналізу даних, математичного моделювання та машинного навчання.

Наявність інструментів для роботи з табличними даними, побудови моделей градієнтного бустингу та виконання статистичних обчислень робить Python придатним для створення інтелектуального ядра системи. Додатковою перевагою є підтримка паралельних обчислень, що дозволяє прискорити навчання моделей на великих наборах клінічної інформації та оперативно оновлювати прогнози у разі надходження нових медичних даних [23].

Суттєве значення має також вибір архітектурного підходу до побудови програмного комплексу. Для розроблюваної системи доцільним є використання сервіс-орієнтованої або мікросервісної архітектури, оскільки вона дозволяє відокремити аналітичний модуль від клієнтського інтерфейсу та серверної бізнес-логіки. Такий підхід створює можливість незалежного масштабування окремих компонентів системи залежно від рівня навантаження, а також спрощує подальше оновлення або заміну окремих модулів.

Сервісна модель дає змогу поєднувати різні технології в межах одного програмного середовища без жорсткої прив’язки всіх компонентів до єдиної платформи. Серверна частина може бути реалізована із застосуванням Node.js або Python, що забезпечує швидку обробку запитів, виконання необхідних обчислень та організацію взаємодії між базою даних, аналітичним модулем і клієнтською частиною. Такий підхід є доцільним для медичних систем, де важливо не лише отримати прогноз, а й забезпечити стабільну передачу даних між різними рівнями програмного комплексу.

Клієнтський інтерфейс доцільно створювати за допомогою сучасних JavaScript-фреймворків, які підтримують динамічне оновлення даних і зручну візуалізацію результатів прогнозування. Завдяки такій організації користувач отримує можливість працювати з медичними записами, переглядати аналітичні показники та отримувати результати прогнозу без повного перезавантаження сторінки.

Окремою перевагою є можливість поступового оновлення або заміни окремих модулів системи без порушення роботи інших її частин. У результаті формується архітектура, за якої зміни або збої в окремому компоненті не порушують стабільність функціонування всього програмного комплексу. Така структура підвищує надійність системи, спрощує її супровід та створює умови для подальшого масштабування у разі збільшення кількості користувачів або обсягу медичних даних. В таблиці 2.1 наведено порівняльний аналіз методів та технологій прогнозування в медицині.

Таблиця 2.1 – Порівняльний аналіз методів та технологій прогнозування в медицині

Методологія / Критерій	**Статистична**	**Метрична**	**Ансамблева**	**Нейромережева**
Здатність до інтерпретації	Дуже висока	Середня	Висока	Низька
Робота з нелінійністю	Обмежена	Висока	Дуже висока	Максимальна
Стійкість до пропусків	Низька	Чутлива	Дуже висока	Середня
Обчислювальні витрати	Мінімальні	Високі	Помірні	Дуже високі
Ризик перенавчання	Низький	Високий	Контрольований	Дуже високий

Під час розробки інтелектуальної медичної системи недостатньо зосередитися лише на виборі алгоритму прогнозування. Не менш важливим є спосіб організації даних, оскільки саме від якості їх збереження та узгодженості залежить коректність подальшого аналізу.

Клінічні записи містять персональну та діагностичну інформацію, тому база даних повинна забезпечувати контроль зв’язків між об’єктами, захист від

дублювання та збереження достовірності кожного запису. Для цього доцільно використовувати реляційну модель у поєднанні з ORM-засобами, які дозволяють пов’язати програмну логіку зі структурою сховища даних без прямої роботи з кожним SQL-запитом.

У запропонованій системі база даних, серверна частина, аналітичний модуль і користувацький інтерфейс виконують різні функції, але працюють як єдина технологічна схема. Сховище даних відповідає за впорядковане збереження інформації про пацієнтів та результати обстежень, серверний рівень координує обмін між компонентами, а модуль машинного навчання формує прогноз на основі підготовленого набору ознак. Завдяки такому розподілу система зберігає логічну цілісність і не перетворюється на набір ізольованих програмних частин.

Обрані технології взаємно доповнюють одна одну. Реляційна база даних забезпечує структурованість і контроль цілісності, ORM спрощує роботу з медичними сутностями, а алгоритми класифікації виконують аналітичну обробку підготовлених записів. У результаті підвищується стабільність роботи системи, зменшується ризик помилок під час обробки інформації та створюються умови для більш точного прогнозування у практичному медичному середовищі [24].

Окремим етапом є перевірка якості роботи прогнозної моделі. Оцінювання результатів потрібне не лише для визначення загальної точності, а й для розуміння того, наскільки система здатна коректно працювати з реальними клінічними даними.

У межах такого підходу процес аналізу охоплює послідовні етапи: отримання інформації, її попередню підготовку, побудову моделі, перевірку ефективності та пояснення отриманого прогнозу.

Саме така послідовність створює основу для подальшого проєктування бази даних, програмних модулів і функціональної структури системи прогнозування.

2.2 Математична формалізація моделі машинного навчання на основі градієнтного бустингу

Побудова прогнозної моделі для медичної системи потребує вибору такого алгоритмічного підходу, який здатний працювати з клінічними даними не як з набором ізольованих чисел, а як зі складною сукупністю взаємопов’язаних показників. У випадку аналізу стану пацієнта значення мають не лише окремі параметри, наприклад артеріальний тиск або рівень глюкози, а й характер їх поєднання. Саме тому для задачі прогнозування доцільно використовувати методи, здатні виявляти нелінійні залежності та поступово уточнювати межу між нормальним станом і підвищеним ризиком.

Одним із таких підходів є ансамблеве навчання. Його сутність полягає у тому, що підсумковий прогноз формується не однією моделлю, а сукупністю простіших алгоритмів. Кожна окрема модель може мати обмежену точність, однак об’єднання багатьох часткових рішень дозволяє отримати більш стабільний результат. Для медичних даних така властивість має важливе значення, оскільки реальні клінічні записи нерідко містять шум, пропуски, нетипові комбінації симптомів або похибки вимірювання.

У медичній діагностиці ансамблеві моделі дають змогу зменшити залежність результату від випадкових коливань у вибірці. Якщо окремий алгоритм неправильно реагує на аномальний або нетиповий запис, сукупна робота ансамблю частково компенсує таку похибку. У результаті модель краще узагальнює закономірності, що характерні не лише для навчального набору, а й для нових пацієнтів. Саме здатність до узагальнення є однією з головних вимог до інтелектуальних систем медичного прогнозування.

Серед ансамблевих методів у межах даної роботи основну увагу приділено градієнтному бустингу. На відміну від підходів типу bagging, де окремі дерева створюються незалежно, бустинг використовує послідовне навчання. Кожне нове дерево не повторює роботу попередніх, а доповнює вже побудовану модель, зменшуючи помилки, які залишилися після попередніх ітерацій. Завдяки такій

логіці алгоритм поступово зміщує фокус на ті клінічні випадки, які складніше класифікувати.

З математичної точки зору процес бустингу можна розглядати як послідовне наближення до мінімального значення функції втрат. Кожна нова модель додається у напрямку, який зменшує похибку прогнозування. У практичному сенсі алгоритм поступово уточнює рішення, аналізуючи залишкові помилки та коригуючи попередній результат. Для задач раннього виявлення патологічних ризиків така властивість є особливо корисною, оскільки модель отримує змогу реагувати не лише на очевидні відхилення, а й на менш виражені комбінації показників [25].

Під час формування прогнозної задачі важливо визначити формат цільової змінної. У межах даного дослідження використовується бінарна класифікація, за якої система оцінює належність пацієнта до одного з двох станів: умовно нормального або такого, що пов’язаний із підвищеним патологічним ризиком. Для медичного скринінгу подібна постановка є найбільш практичною, оскільки першочергове завдання полягає у своєчасному виявленні пацієнтів, які потребують додаткової уваги лікаря. Багатокласова класифікація могла б застосовуватися для деталізації окремих стадій захворювання, однак такий варіант потребує великої кількості якісно розмічених прикладів для кожного класу. У реальних медичних наборах даних розподіл записів між категоріями часто є нерівномірним. Частина патологічних станів трапляється рідше, що ускладнює навчання моделі та може призводити до нестабільних результатів. Бінарна модель зменшує вплив такої проблеми, оскільки концентрується на відокремленні групи ризику від загальної сукупності пацієнтів.

Для бінарної класифікації доцільно використовувати логістичну функцію втрат. Завдяки такій постановці результат прогнозування подається у вигляді числової ймовірності в інтервалі від 0 до 1. Для клінічного використання подібний формат є зручнішим, ніж просте присвоєння класу. Лікар отримує не лише категоріальний висновок, а й кількісну оцінку ризику.

У випадку багатокласових моделей, побудованих на Softmax-нормалізації, імовірність розподіляється між кількома категоріями. У практичній медицині подібний формат може ускладнювати оцінювання критичності стану, особливо коли необхідно швидко визначити, чи належить пацієнт до групи ризику. Тому для розроблюваної системи бінарна класифікація є більш доцільною, оскільки забезпечує зрозумілий та оперативний результат для прийняття клінічного рішення. Математична основа XGBoost базується на оптимізації цільової функції, яка поєднує дві складові. Перша складова відповідає за помилку навчання та показує, наскільки прогноз моделі відрізняється від фактичного значення. Друга складова виконує роль регуляризації, тобто обмежує складність моделі та зменшує ризик перенавчання. Для медичних даних поєднання точності й контролю складності має принципове значення, оскільки модель не повинна надмірно пристосовуватися до поодиноких нетипових записів.

На кожній ітерації алгоритм не створює модель з нуля. До наявного ансамблю додається нова функція f_t, яка зменшує залишкову похибку попередньої сукупності дерев. Пряма оптимізація функції втрат у функціональному просторі є складною обчислювальною задачею, тому в XGBoost застосовується наближення за допомогою розкладу Тейлора другого порядку. Завдяки такому перетворенню оптимізація стає більш керованою, оскільки враховується не лише напрямок зміни помилки, а й локальна кривизна функції втрат. Фундаментом для побудови кожного нового дерева виступає наступне математичне вираження:

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2$$

У наведеній формулі g_i позначає градієнт функції втрат для і-го об’єкта навчальної вибірки. Градієнт визначає напрямок корекції поточного прогнозу моделі з метою зменшення помилки. Величина h_i позначає гессіан, тобто другу похідну функції втрат для і-го об’єкта. Гессіан характеризує локальну кривизну

функції помилки та дає змогу точніше визначити інтенсивність оновлення прогнозу. Функція $f_t(x_i)$ відображає прогноз, сформований t-м деревом рішень для об'єкта x_i. Дерево додається до вже побудованого ансамблю та забезпечує поступове уточнення загального результату моделі.

Регуляризаційна частина формули містить параметр γ, що задає штраф за кількість листкових вузлів дерева T. Такий механізм обмежує надмірне розгалуження дерева та знижує ризик перенавчання моделі.

Параметр λ відповідає за L2-регуляризацію ваг листків w_j. Обмеження вагових коефіцієнтів запобігає надмірному впливу окремих аномальних або випадкових спостережень на підсумковий прогноз.

Для медичної діагностики контроль таких впливів є особливо важливим. У клінічних записах можуть траплятися технічні похибки, неточності введення або поодинокі значення, що не відображають стійкої закономірності. Без регуляризації модель могла б сприйняти такі фрагменти як значущі сигнали. Використання штрафних параметрів змушує алгоритм спиратися на більш узгоджену сукупність ознак і формувати прогноз на основі кількох клінічно важливих факторів, а не одного випадкового відхилення [26].

Побудова дерева рішень у XGBoost передбачає пошук оптимального розщеплення у кожному вузлі. Для цього алгоритм аналізує можливі порогові значення ознак і визначає поділ, який найбільше покращує якість моделі. Вибір між лівим L та правим R піддеревами ґрунтується на критерії приросту якості (Gain). У математичному вираженні такий критерій враховує співвідношення сум градієнтів і гессіанів у новостворених частинах дерева.

У результаті дерево не просто поділяє дані механічно, а формує ієрархію ознак відповідно до їх прогностичної ваги. На верхніх рівнях зазвичай розміщуються параметри, що дають найбільший внесок у розмежування класів. Для задачі серцево-судинного ризику такими ознаками можуть бути артеріальний тиск, вік, рівень глюкози, холестерин або індекс маси тіла. На нижчих рівнях уточнюються складніші комбінації параметрів, які можуть бути важливими лише для частини пацієнтів.

Механізм навчання на залишках забезпечує поступове вдосконалення ансамблю. Якщо попередні дерева помилилися на пацієнті з нетиповою комбінацією симптомів, відповідний приклад отримує більший вплив на наступних ітераціях. Наступне дерево більше орієнтується на виправлення такої помилки, а не на повторення вже правильно класифікованих випадків. Завдяки такій логіці XGBoost краще виявляє приховані нелінійні зв’язки, які складно врахувати за допомогою простих лінійних моделей.

Після визначення структури дерева необхідно розрахувати оптимальні значення у його листкових вузлах. Кожен листок містить ваговий коефіцієнт, який показує внесок відповідної групи об’єктів у фінальний прогноз. Оптимальна вага визначається через співвідношення сумарного градієнта, сумарного гессіана та регуляризаційного коефіцієнта. Такий спосіб обчислення дозволяє враховувати не лише величину помилки, а й надійність статистичного сигналу в конкретному вузлі.

Якщо у певному листку накопичено мало спостережень, регуляризація зменшує його вплив. У результаті модель не робить надто категоричних висновків на основі недостатньої кількості даних. Для медичного прогнозування така властивість є критично важливою, оскільки помилкове підсилення рідкісного або випадкового шаблону може призвести до некоректного визначення ризику.

Додатковий контроль стабільності забезпечує механізм shrinkage. Його сутність полягає у тому, що внесок кожного нового дерева масштабується коефіцієнтом швидкості навчання η\etaη. Завдяки такому обмеженню ансамбль оновлюється поступово, а жодне окреме дерево не може різко змінити підсумковий результат. Менший крок навчання зазвичай потребує більшої кількості дерев, однак підвищує стійкість моделі та знижує ризик перенавчання.

Підсумковий прогноз формується шляхом послідовного додавання внесків усіх дерев ансамблю. Кожен елемент моделі робить часткову корекцію, а загальна оцінка перетворюється у ймовірність належності пацієнта до групи ризику.

У межах розроблюваної системи такий результат може використовуватися для подальшої візуалізації, формування рекомендацій та підтримки прийняття рішень лікарем.

Таким чином, математична структура XGBoost поєднує кілька рівнів контролю: ітераційне виправлення помилок, оптимізацію за градієнтами й гессіанами, регуляризацію складності дерев та поетапне оновлення ансамблю.

Для задачі аналізу електронної медичної документації така комбінація є доцільною, оскільки забезпечує стійкість до неповних даних, здатність працювати з нелінійними залежностями та достатню точність для превентивного моніторингу стану пацієнта.

2.3 Архітектурне проектування та структурно-функціональний аналіз системи

Архітектурне проектування інтелектуальної системи прогнозування медичних станів передбачає визначення такої структури програмного комплексу, яка здатна забезпечити швидку обробку клінічних даних, захищене передавання інформації та стабільну взаємодію між усіма функціональними модулями.

Для медичної сфери архітектура має особливе значення, оскільки система працює не лише з технічними запитами, а з персональними даними пацієнтів, результатами обстежень та прогнозними висновками, що можуть бути використані під час прийняття клінічних рішень.

З урахуванням специфіки задачі та математичного апарату градієнтного бустингу, архітектуру системи доцільно будувати за багаторівневим сервіс-орієнтованим принципом.

Такий підхід передбачає логічне розділення клієнтського інтерфейсу, серверної частини, бази даних та аналітичного модуля машинного навчання.

Завдяки такій організації кожен компонент виконує окрему функцію, а обмін інформацією між ними відбувається через визначені програмні інтерфейси.

На рисунку 2.1 зображено архітектуру системи.

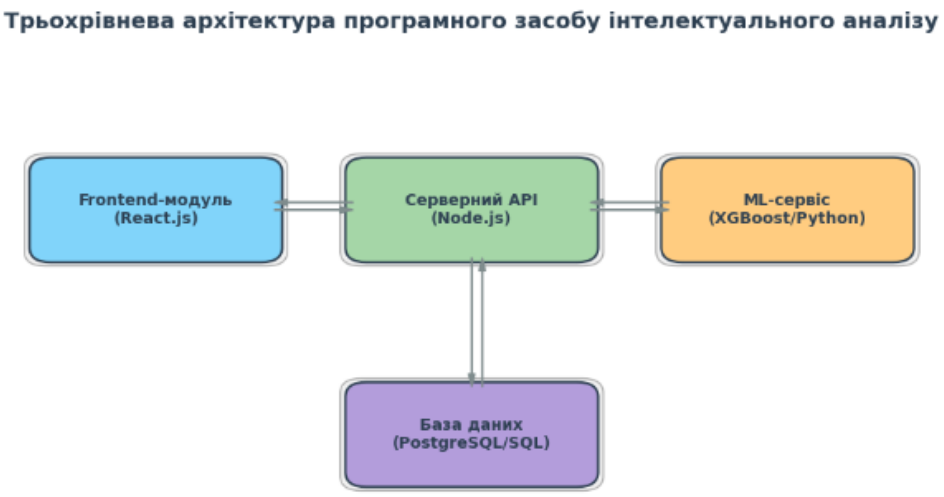

Рисунок 2.1 – Архітектура системи

Запропонована архітектура реалізує принцип розподілення відповідальності, за якого окремі частини системи не дублюють функції одна одної. Клієнтський рівень відповідає за взаємодію з користувачем, серверний рівень координує обробку запитів і роботу з даними, а аналітичний мікросервіс виконує ресурсоємні обчислення. Для медичної інформаційної системи такий розподіл є доцільним, оскільки оновлення інтерфейсу або модернізація моделі машинного навчання можуть виконуватися без повної зупинки програмного комплексу та без порушення цілісності медичних записів [27].

Рівень представлення реалізується у вигляді односторінкового вебзастосунку на основі React.js. Вибір такої технології пов’язаний із потребою у швидкому та інтерактивному інтерфейсі, який дозволяє лікарю працювати з карткою пацієнта, формами введення клінічних показників, історією обстежень і результатами прогнозування без перезавантаження сторінки. Для медичного персоналу швидкість оновлення інтерфейсу має практичне значення, оскільки під час прийому важливо швидко отримувати доступ до потрібних даних і не витрачати час на зайві переходи між сторінками.

Клієнтська частина виконує не лише функцію відображення інформації. У межах розроблюваної системи вона бере участь у первинній підготовці даних,

контролі коректності введення та візуальному поданні результатів аналітичного модуля. Механізм керування станом забезпечує узгоджену роботу різних елементів інтерфейсу, зокрема картки пацієнта, форми прогнозування, таблиці історії обстежень та графічних компонентів. Завдяки такій організації зміна даних в одному блоці інтерфейсу може автоматично відображатися в інших пов’язаних елементах.

Окремим завданням клієнтського рівня є валідація вхідних медичних показників. Перед відправленням запиту на сервер такі параметри, як артеріальний тиск, рівень глюкози, індекс маси тіла та інші числові значення, перевіряються на відповідність допустимим типам і фізіологічним межам. Попередня перевірка зменшує ймовірність передавання помилкових або некоректно введених значень до серверної частини. Для задачі прогнозування така перевірка має важливе значення, оскільки якість вхідного вектора безпосередньо впливає на достовірність результату роботи моделі.

Результати, отримані від аналітичного модуля, потребують зручного подання для медичного фахівця. Модель машинного навчання формує числові значення, зокрема ймовірність ризику та внески окремих ознак. Без належної візуалізації така інформація є складною для швидкої клінічної інтерпретації. Тому на рівні React-інтерфейсу передбачаються компоненти для відображення індикаторів ризику, графіків динаміки, діаграм та пояснювальних елементів. Компонентний підхід дозволяє реалізувати кожен візуальний блок як окремий модуль, що оновлюється відповідно до зміни стану системи.

Обмін між клієнтською та серверною частинами здійснюється через асинхронні HTTP-запити із використанням захищеного протоколу HTTPS. Асинхронна модель взаємодії дає змогу не блокувати роботу інтерфейсу під час виконання прогнозування. Поки сервер обробляє запит і передає дані до аналітичного модуля, користувач може продовжувати роботу з іншими елементами системи. Для медичного закладу такий режим роботи підвищує ергономічність програмного комплексу та скорочує час доступу до результатів аналізу [28].

Серверна частина є центральним координаційним рівнем системи. Її реалізація на основі Node.js обґрунтовується подійно-орієнтованою моделлю виконання та неблокуючим механізмом введення-виведення. Така технологічна основа дає змогу обробляти одночасні запити від кількох користувачів без створення окремого потоку для кожного підключення. Для медичної інформаційної системи, де паралельно можуть виконуватися операції перегляду пацієнтів, оновлення даних і запуску прогнозування, така властивість має суттєве значення.

У межах запропонованої архітектури сервер виконує роль API-шлюзу. До його функцій належить приймання запитів від клієнтського інтерфейсу, перевірка вхідних параметрів, взаємодія з реляційною базою даних, формування вектора ознак і передавання підготовлених даних до Python-мікросервісу. Після отримання відповіді від аналітичного ядра сервер виконує постобробку результату, перетворює числову оцінку у клінічно зрозумілий формат і зберігає прогноз у базі даних.

Важливим елементом серверної частини є проміжне програмне забезпечення. Кожен запит проходить послідовну обробку на кількох рівнях, що дозволяє забезпечити безпеку, контроль коректності даних та фіксацію дій користувача. На першому рівні виконується автентифікація та авторизація за допомогою JSON Web Token. Такий механізм дає змогу перевірити особу користувача, визначити його роль у системі та обмежити доступ до медичних даних відповідно до наданих повноважень.

Наступним рівнем є журналювання операцій. Фіксація часу запиту, IP-адреси, типу дії та службових параметрів створює основу для подальшого аудиту. У разі помилки, несанкціонованої спроби доступу або некоректної роботи окремого модуля журнали дозволяють відновити послідовність подій і визначити причину порушення. Для систем, що працюють із медичною інформацією, контрольованість операцій має не лише технічне, а й організаційне значення.

Серверна валідація виконує роль другого рівня контролю після перевірки на стороні клієнта. Усі дані повторно перевіряються на відповідність очікуваним типам, форматам і допустимим діапазонам. Такий підхід потрібний через те, що клієнтська перевірка може бути обійдена або порушена внаслідок некоректної роботи браузера чи стороннього втручання. Серверна обробка також зменшує ризик SQL-ін'єкцій, передачі помилкових структур запиту та потрапляння некоректних параметрів до моделі машинного навчання [29].

Взаємодія з реляційною базою даних здійснюється через ORM-рівень. Завдяки такому підходу медичні сутності, наприклад пацієнти, клінічні записи та прогнози, обробляються як об'єкти програмної логіки. ORM спрощує роботу з базою даних, зменшує кількість ручного SQL-коду та допомагає підтримувати узгодженість між програмними моделями й таблицями сховища. Для розробки медичної системи такий спосіб взаємодії підвищує зручність супроводу та знижує ймовірність помилок у запитах.

Під час запуску процедури прогнозування серверна частина виконує агрегацію даних. До підготовленого набору можуть входити поточні клінічні показники, демографічні характеристики пацієнта, результати попередніх обстежень і супутні медичні параметри. Після об'єднання інформації формується уніфікований JSON-об'єкт, який передається до аналітичного мікросервісу. Така схема дозволяє стандартизувати вхідні дані для моделі та забезпечити однаковий формат прогнозування для всіх пацієнтів.

Після завершення роботи ML-модуля сервер отримує відповідь у вигляді числової оцінки ризику та додаткових пояснювальних даних. На цьому етапі виконується перетворення результату у формат, придатний для збереження та подальшого відображення в інтерфейсі. Ймовірнісна оцінка може бути інтерпретована як низький, середній або високий рівень ризику. Після обробки результат записується до бази даних із дотриманням принципів транзакційності ACID, що запобігає збереженню неповних або некоректних прогнозів у разі збою під час обміну даними. Для підвищення продуктивності серверний рівень може використовувати кешування відповідей. Тимчасове збереження результатів у

оперативній пам’яті є корисним у ситуаціях, коли лікар повторно відкриває результати одного пацієнта протягом короткого проміжку часу. Завдяки кешуванню зменшується кількість повторних звернень до аналітичного модуля, скорочується час відповіді системи та знижується загальне обчислювальне навантаження.

Централізована обробка помилок забезпечує стабільність роботи серверної частини. У випадку внутрішнього збою користувач отримує зрозуміле повідомлення, а технічні деталі зберігаються в журналах для подальшого аналізу адміністратором. Такий підхід дозволяє уникнути ситуацій, коли медичний працівник бачить лише технічний код помилки та не може зрозуміти, на якому етапі виникла проблема. Для практичного використання медичного програмного забезпечення зрозумілість повідомлень має важливе значення.

Аналітичний рівень є найбільш обчислювально складною частиною системи. У запропонованій архітектурі він реалізується як окремий Python-мікросервіс із використанням FastAPI. Виокремлення аналітичного ядра в самостійний сервіс дозволяє не перевантажувати Node.js-сервер математичними обчисленнями та зберігати стабільність основної бізнес-логіки. Така організація також спрощує подальше оновлення моделі, оскільки аналітичний компонент може змінюватися незалежно від клієнтської частини та серверного API.

Використання Python для реалізації ML-модуля обґрунтовується наявністю розвиненої екосистеми бібліотек для машинного навчання та наукових обчислень. NumPy і Pandas забезпечують підготовку та обробку табличних даних, а XGBoost використовується для реалізації прогнозної моделі на основі градієнтного бустингу. FastAPI, у свою чергу, забезпечує швидку обробку запитів і підтримує асинхронну модель взаємодії, що зменшує затримки під час обміну даними між сервером і аналітичним ядром.

Внутрішня логіка аналітичного мікросервісу побудована як послідовний конвеєр обробки даних. Після отримання JSON-об’єкта від серверної частини виконується перевірка структури, нормалізація числових параметрів та обробка пропущених значень. Для медичних записів такий етап є обов’язковим, оскільки

реальні клінічні дані часто мають нерівномірну заповненість. Підхід Sparsity-aware Split Finding, реалізований у XGBoost, дозволяє моделі коректно працювати з розрідженими наборами ознак без втрати аналітичної стійкості.

Наступним етапом є використання попередньо навченої моделі, яка зберігається у серіалізованому вигляді та завантажується в оперативну пам’ять мікросервісу. Такий режим дозволяє уникнути повторної ініціалізації моделі під час кожного запиту. Для медичної системи швидке завантаження прогнозної моделі має практичне значення, оскільки результат повинен бути доступним у межах короткого часу, придатного для роботи під час прийому пацієнта.

Під час прогнозування модель виконує проходження через ансамбль дерев рішень. Кожен вузол перевіряє певну умову щодо вхідних клінічних параметрів, а підсумкова оцінка формується шляхом агрегування внесків усіх дерев. Результатом роботи аналітичного модуля є не лише формальна класифікація, а й імовірнісне значення ризику. Такий формат краще відповідає потребам клінічної практики, оскільки дозволяє оцінити ступінь вираженості ризику, а не лише факт належності до певного класу.

Додатковою складовою аналітичного рівня є пояснення прогнозу. Для цієї мети можуть використовуватися SHAP-значення, які показують внесок кожної ознаки у підсумкову оцінку. Завдяки такій інтерпретації лікар отримує можливість зрозуміти, які саме параметри найбільше вплинули на результат, наприклад рівень глюкози, артеріальний тиск, індекс маси тіла або вік пацієнта. Пояснюваність моделі підвищує довіру до системи та робить її придатнішою для застосування у підтримці клінічних рішень.

Ізоляція аналітичного рівня також спрощує масштабування. У разі зростання кількості користувачів або збільшення обсягу прогнозних запитів ML-сервіс може бути розгорнутий окремо та масштабований незалежно від основного сервера. Контейнеризація, зокрема Docker, дає змогу переносити аналітичний модуль між різними середовищами та автоматизувати процес розгортання. Такий підхід забезпечує гнучкість програмного комплексу та створює умови для подальшої заміни або донавчання моделі.

У результаті запропонована архітектура поєднує користувацький інтерфейс, серверну логіку, базу даних та аналітичний мікросервіс у цілісну функціональну структуру. Розподіл системи на окремі рівні забезпечує стабільність обробки запитів, контроль якості вхідних даних, захист медичної інформації та можливість подальшого розвитку. Така архітектурна модель створює технічну основу для реалізації інтелектуального аналізу електронної медичної документації та підтримки прийняття клінічних рішень.

2.4 Проектування бази даних та структури збереження інформації

Проєктування бази даних для інтелектуальної медичної системи є окремим етапом розробки, оскільки якість організації інформації безпосередньо впливає на коректність подальшого прогнозування. У межах розроблюваного програмного комплексу сховище даних повинно забезпечувати збереження персональної інформації пацієнтів, клінічних показників, результатів роботи моделі машинного навчання та службових відомостей про користувачів системи. Для медичної сфери така структура має особливе значення, оскільки будь-яке порушення зв’язку між пацієнтом, обстеженням і прогнозом може призвести до некоректної інтерпретації результатів.

Для реалізації інформаційного забезпечення обрано реляційну модель даних. Такий підхід є доцільним для систем, у яких необхідно підтримувати чіткі логічні зв’язки між окремими сутностями та гарантувати посилальну цілісність. Медична інформаційна система повинна не просто зберігати окремі записи, а формувати впорядковану історію взаємодії з пацієнтом, включаючи його обстеження, результати аналізів і сформовані прогнозні висновки. Реляційна структура дає змогу уникнути дублювання даних і забезпечити однозначне зіставлення кожного клінічного запису з відповідним профілем пацієнта.

Використання СУБД PostgreSQL або MySQL дозволяє застосовувати механізми зовнішніх ключів, індексів та транзакційної обробки. Зовнішні ключі

забезпечують контроль зв’язків між таблицями та запобігають появі клінічних або прогнозних записів, які не мають відповідного об’єкта в основних сутностях.

Транзакційність відповідно до принципів ACID гарантує узгодженість бази даних у випадках збою під час запису або оновлення інформації. Для медичних систем така властивість є критичною, оскільки збереження неповного або помилкового прогнозу може вплинути на подальший аналіз стану пацієнта [30]. Завдяки атомарності транзакцій усі пов’язані операції виконуються як єдине ціле: або повністю завершуються, або скасовуються у разі виникнення помилки. Це особливо важливо тоді, коли одночасно зберігаються клінічні показники пацієнта, результат прогнозування та службова інформація про дату створення запису. У разі технічного збою система не повинна залишати частково оновлені дані, оскільки це може призвести до некоректного відображення історії спостережень або помилкової інтерпретації результатів аналізу.

Продуктивність бази даних також має суттєве значення, оскільки аналітичний модуль може часто звертатися до історичних записів пацієнта. Для швидкого отримання даних за ідентифікатором, датою візиту або іншими ключовими полями доцільно застосовувати індексування. Така організація пришвидшує виконання запитів навіть за умов поступового збільшення кількості пацієнтів і накопичення великого обсягу медичних записів [31]. Крім того, індекси дають змогу ефективніше виконувати вибірки за часовими проміжками, що є важливим для аналізу динаміки змін показників пацієнта. Раціональне проєктування індексів дозволяє зменшити навантаження на сервер бази даних і забезпечити стабільну роботу системи під час одночасного доступу кількох користувачів. При цьому індексування має застосовуватися до тих полів, які найчастіше використовуються у фільтрації, пошуку та зв’язуванні таблиць.

Центральною сутністю бази даних є таблиця Patients, яка зберігає базову інформацію про пацієнта. До цієї таблиці вносяться відомості, що мають відносно стабільний характер і використовуються для ідентифікації особи в системі. Винесення персональної інформації в окрему сутність дозволяє не

дублювати її у кожному клінічному записі та забезпечує більш раціональну організацію сховища.

Розмежування персональних і клінічних даних має практичне значення для побудови історії спостережень. Дані про пацієнта зберігаються в одному місці, тоді як результати обстежень накопичуються в окремих таблицях.

Завдяки такій структурі система може зберігати тривалу історію змін медичних показників без надмірного збільшення обсягу основної таблиці профілів. Схему моделі в базі даних для сутності пацієнта можна побачити на рисунку 2.2.

Рисунок 2.2 – Схема таблиці для сутності пацієнт

Сутність пацієнта містить набір полів, необхідних для однозначної ідентифікації та первинного опису особи в системі. Поле id виконує роль унікального ідентифікатора та використовується для зв’язку з іншими таблицями. Поле full_name призначене для збереження прізвища, імені та по батькові пацієнта. Демографічні характеристики представлені полями age та gender, а контактна інформація зберігається у полі phone. Поле created_at фіксує дату та час створення профілю, що дозволяє визначити момент реєстрації пацієнта в системі.

Наступною сутністю є ClinicalData, яка відповідає за збереження медичних показників, отриманих під час обстежень. На відміну від таблиці Patients, дана

сутність має динамічний характер, оскільки для одного пацієнта може створюватися багато записів у різні дати.

Така організація дозволяє аналізувати не лише поточний стан, а й зміну клінічних параметрів у часі. Зв’язок із пацієнтом забезпечується через зовнішній ключ patient_id. Схему моделі в базі даних для сутності клінічні дані можна побачити на рисунку 2.3.

Рисунок 2.3 – Схема таблиці для сутності клінічні дані

Сутність клінічних даних містить показники, які використовуються для подальшого аналітичного опрацювання та формування вектора ознак моделі машинного навчання. Поле id ідентифікує окремий запис обстеження. Поле patient_id встановлює зв’язок із відповідним пацієнтом. До основних медичних параметрів належать blood_pressure, cholesterol, glucose та bmi. Збереження цих значень у структурованому вигляді дає змогу виконувати як поточний аналіз, так і подальше порівняння показників у динаміці. Поле visit_date визначає дату та час проведення обстеження, що є необхідним для побудови часової історії стану пацієнта.

Окрему роль у базі даних виконує таблиця Predictions, призначена для збереження результатів роботи інтелектуального модуля. Прогноз не повинен

існувати ізольовано від вхідних медичних показників, тому кожен запис у цій таблиці прив’язується до конкретного клінічного обстеження. Такий підхід дозволяє встановити, на основі яких саме параметрів було сформовано прогноз, і забезпечує відтворюваність аналітичного висновку. Схему моделі в базі даних для сутності прогноз можна побачити на рисунку 2.4.

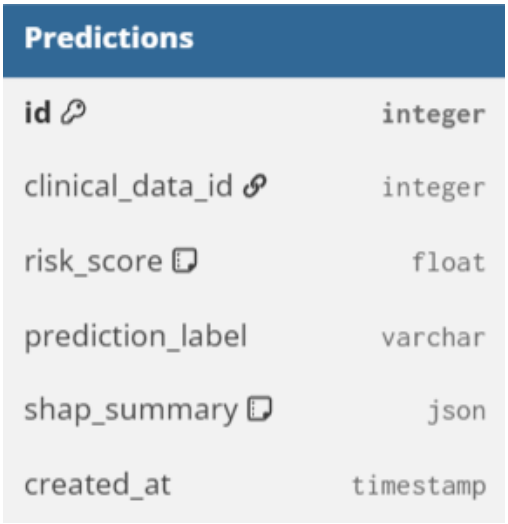

Рисунок 2.4 – Схема таблиці для сутності прогноз

Сутність прогнозу фіксує результат обробки клінічних даних моделлю машинного навчання. Поле id використовується для унікальної ідентифікації прогнозного запису. Поле clinical_data_id встановлює зв’язок із обстеженням, на основі якого було виконано розрахунок. Числовий результат зберігається у полі risk_score та відображає ймовірність розвитку патологічного стану. Для зручності клінічного сприйняття передбачено поле prediction_label, де результат подається у текстовій формі, наприклад як низький, середній або високий рівень ризику. Поле created_at фіксує час створення прогнозу та дає змогу аналізувати історію аналітичних висновків.

Для організації доступу до системи передбачено таблицю Users. У цій сутності зберігаються дані про користувачів, які мають право працювати з програмним комплексом. Наявність окремої таблиці користувачів дозволяє реалізувати автентифікацію, розмежування ролей і контроль доступу до медичної інформації. У медичних системах такий механізм є необхідним,

оскільки різні категорії користувачів можуть мати різні повноваження щодо перегляду, редагування або адміністрування даних. Схему моделі для сутності користувач можна побачити на рисунку 2.5.

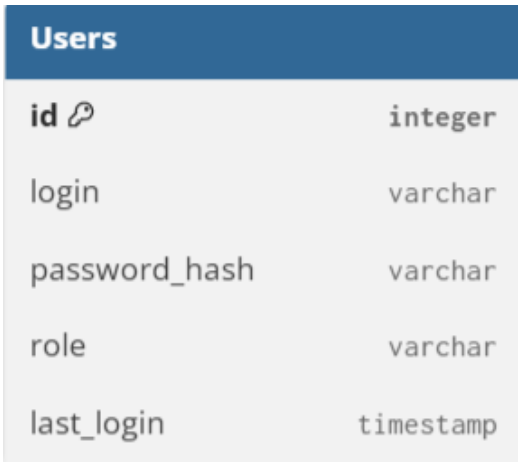

Рисунок 2.5 – Схема таблиці для сутності користувач

Сутність користувача містить поля, необхідні для ідентифікації та контролю доступу. Поле id є унікальним ідентифікатором облікового запису. Поле login використовується для входу в систему. Для забезпечення безпеки пароль не зберігається у відкритому вигляді, замість цього застосовується поле password_hash, у якому розміщується хешоване значення. Поле role визначає тип користувача, наприклад лікар або адміністратор, та використовується для обмеження доступу до окремих функцій системи. Поле last_login може застосовуватися для фіксації останнього входу та контролю активності користувача.

Після визначення основних сутностей виконується побудова логічної моделі бази даних. На цьому етапі встановлюються зв’язки між таблицями, визначаються первинні й зовнішні ключі та перевіряється відповідність структури принципам нормалізації. Основне завдання полягає у тому, щоб забезпечити збереження даних без дублювання та водночас не втратити можливість швидко отримувати необхідну інформацію для роботи аналітичного модуля. Зв’язок між Patients і ClinicalData дозволяє формувати повну історію клінічних спостережень для кожного пацієнта. Один профіль може мати багато

записів обстежень, що відповідає реальній медичній практиці, де стан пацієнта оцінюється не одноразово, а протягом певного періоду. Зв'язок між ClinicalData і Predictions забезпечує прив'язку кожного прогнозу до конкретного набору вхідних показників. Така організація підвищує прозорість роботи системи та дозволяє повторно перевірити підстави для сформованого висновку [32].

Правильний вибір типів полів, індексів і зв'язків між сутностями зменшує час доступу до даних і підвищує ефективність функціонування всієї системи. Після визначення та опису усіх сутностей системи, можна розробити повну діаграму бази даних. На рисунку 2.6 зображено загальну діаграму бази даних інтелектуальної системи.

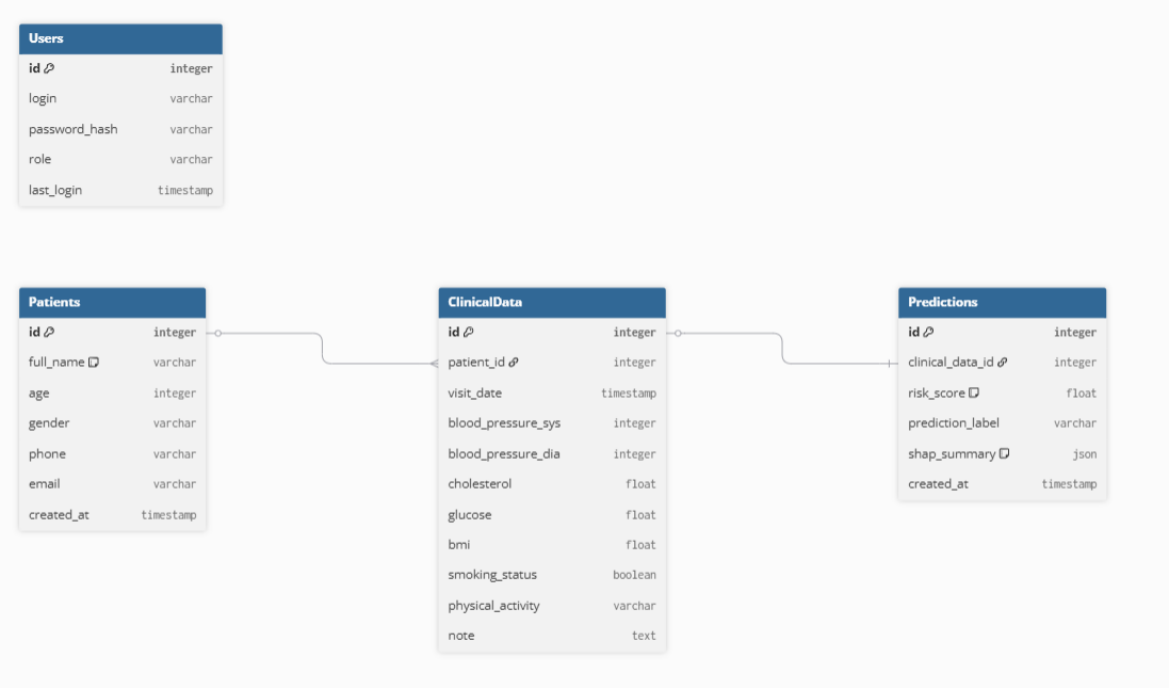

Рисунок 2.6 – Загальна діаграма бази даних інтелектуальної системи

Загальна діаграма бази даних відображає чотири ключові сутності та логіку їх взаємодії. Центральне місце займає таблиця Patients, оскільки з нею пов'язуються всі медичні записи пацієнта. Між Patients і ClinicalData встановлюється зв'язок типу «один-до-багатьох», що дозволяє зберігати множину обстежень для одного пацієнта. Така структура відповідає задачі

аналізу динаміки захворювань, оскільки забезпечує накопичення показників у часовому розрізі.

Між ClinicalData та Predictions реалізується зв’язок типу «один-до-одного». Кожен прогноз формується на основі конкретного запису клінічних даних, тому результат аналітичного модуля не відокремлюється від джерела вхідної інформації. Такий підхід забезпечує логічну прозорість і дає змогу простежити, які саме медичні параметри були використані під час розрахунку ризику.

Таблиця Users забезпечує облік медичного персоналу та адміністраторів, які працюють із системою. Через цю сутність реалізується контроль доступу до інформації та розмежування функціональних можливостей. У результаті база даних підтримує не лише збереження медичних записів, а й організацію безпечної роботи користувачів із конфіденційною інформацією.

2.5 Висновок до другого розділу

У другому розділі було сформовано математичну, алгоритмічну та архітектурну основу інтелектуальної системи прогнозування динаміки захворювань за даними електронної медичної документації. Основну увагу приділено вибору методів, здатних працювати з неоднорідними, частково неповними та складно структурованими медичними даними. На основі проведеного аналізу обґрунтовано доцільність використання ансамблевого підходу, зокрема алгоритму XGBoost, який поєднує високу точність класифікації, стійкість до шуму та можливість подальшої інтерпретації результатів.

У межах математичної частини розглянуто принципи функціонування градієнтного бустингу, механізм послідовного додавання дерев рішень, роль функції втрат, регуляризації, градієнтів та гессіанів у процесі оптимізації моделі. Окремо обґрунтовано використання бінарної класифікації для задачі раннього виявлення пацієнтів із підвищеним рівнем ризику. Такий формат прогнозування

є практично зручним для медичної сфери, оскільки результат подається у вигляді ймовірнісної оцінки, придатної для подальшого аналізу лікарем.

Також у розділі розроблено архітектурну модель програмного комплексу. Запропонована структура передбачає поділ системи на клієнтський рівень, серверну бізнес-логіку, реляційне сховище даних та окремий аналітичний мікросервіс. Така організація забезпечує незалежність компонентів, спрощує подальше масштабування та дозволяє модернізувати окремі модулі без порушення роботи всієї системи. Особливу увагу приділено взаємодії React-інтерфейсу, Node.js-сервера та Python-модуля машинного навчання.

У процесі проєктування бази даних визначено основні сутності, необхідні для збереження персональної, клінічної та прогнозної інформації. Структура бази даних передбачає таблиці для пацієнтів, клінічних показників, результатів прогнозування та користувачів системи. Завдяки встановленню логічних зв’язків між цими сутностями забезпечується збереження медичного анамнезу, прив’язка кожного прогнозу до конкретного обстеження та контроль доступу медичного персоналу до інформації.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ПРОГНОЗУВАННЯ МЕДИЧНИХ РИЗИКІВ

3.1 Розробка моделі машинного навчання

Розробка інтелектуального модуля прогнозування серцево-судинних ризиків розпочинається з підготовки медичних даних до формату, придатного для машинного навчання. Клінічні показники, введені користувачем через інтерфейс системи, не можуть безпосередньо використовуватися алгоритмом у початковому вигляді, оскільки модель XGBoost працює з числовими ознаками, поданими у впорядкованому багатовимірному просторі. Тому перед виконанням прогнозування всі вхідні параметри проходять етап формалізації.

Під час проєктування моделі було враховано, що якість прогнозу залежить не лише від кількості вхідних показників, а й від їхньої інформативності. Саме тому до вектора ознак включаються не тільки безпосередні результати вимірювань, а й похідні параметри, які краще відображають фізіологічний стан пацієнта. Такий підхід дає змогу врахувати складні зв’язки між антропометричними, гемодинамічними, лабораторними та поведінковими характеристиками, що мають значення для оцінювання кардіологічного ризику.

Одним із важливих етапів підготовки даних є вторинна генерація ознак, або feature engineering. У межах розробленої системи частина таких ознак формується автоматично під час передавання даних із клієнтського інтерфейсу. Зокрема, на основі введених значень зросту та маси тіла розраховується індекс маси тіла BMI. Використання такого показника є більш доцільним порівняно з окремим аналізом зросту або ваги, оскільки BMI узагальнює інформацію про фізичний стан пацієнта та дозволяє моделі врахувати фактор ожиріння, який має суттєвий вплив на ймовірність розвитку серцево-судинних ускладнень [33].

Гемодинамічні показники представлені у моделі окремими значеннями систолічного та діастолічного артеріального тиску. Систолічний тиск

характеризує навантаження на судини під час скорочення серця, тоді як діастолічний тиск пов’язаний із периферичним судинним опором.

Лабораторні параметри, зокрема рівень глюкози та загального холестерину, у системі подаються у категоріальному форматі. Такий спосіб подання особливо корисний для медичних даних, де незначні відхилення в результатах лабораторного обладнання не завжди мають самостійне клінічне значення. В таблиці 3.1 продемонстровано опис показників для аналізу.

Таблиця 3.1 – Значення показників та їх опис

Категорія показників	**Технічна назва**	**Тип даних (Python/ML)**	**Опис логічної ролі ознаки**
Антропометричні	height, weight, bmi	float	Оцінка фізичного статусу та ризиків, пов'язаних з ожирінням.
Гемодинамічні	ap_hi, ap_lo	int	Прямі показники роботи серцево-судинної системи та тиску на судини.
Лабораторні	cholesterol, gluc	category	Біохімічні маркери стану крові та метаболічних процесів.
Поведінкові	smoke, active	uint8	Врахування способу життя пацієнта.

Для демонстрації процесу навчання модельного модуля було використано структурований набір кардіологічних даних. До вхідних параметрів входили антропометричні характеристики пацієнта, показники артеріального тиску, рівень холестерину та глюкози, а також поведінкові фактори, зокрема факт

тютюнопаління та фізична активність. Цільова змінна мала бінарний характер і визначала належність запису до одного з двох класів: відсутність підвищеного серцево-судинного ризику або наявність такого ризику.

Перед навчанням моделі всі дані було уніфіковано та подано у числовому форматі, придатному для обробки алгоритмом XGBoost. На цьому етапі виконувалася перевірка значень на наявність помилок, аномалій або невідповідностей очікуваним діапазонам. Антропометричні параметри використовувалися не лише як окремі ознаки, а й як основа для розрахунку індексу маси тіла BMI. Лабораторні показники, зокрема рівень холестерину та глюкози, подавалися у категоріальному вигляді, що дозволяло врахувати їх клінічну градацію. Поведінкові характеристики, такі як куріння та фізична активність, кодувалися як бінарні змінні. У результаті було сформовано узгоджений вхідний вектор, придатний для подальшого навчання моделі.

Навчальний датасет складався з 600 записів, кожен із яких містив 8 вхідних ознак. Для побудови та перевірки моделі вибірку було поділено на дві частини. До навчальної частини увійшло 480 записів, які використовувалися для формування ансамблю дерев XGBoost. Інші 120 записів було відокремлено для оцінювання якості прогнозування. Такий підхід дозволяє перевірити не лише здатність моделі працювати з уже відомими даними, а й її можливість коректно класифікувати нові записи, які не використовувалися під час навчання.

Поділ вибірки на навчальну та тестову частини є важливим етапом експериментальної перевірки, оскільки він дає змогу оцінити узагальнювальну здатність моделі. Навчальна частина використовується для побудови внутрішніх правил класифікації, тоді як тестова частина виконує роль незалежного контрольного набору. Завдяки цьому можна визначити, наскільки стабільно модель розпізнає закономірності у даних і чи не обмежується її робота простим запам’ятовуванням навчальних прикладів.

Для оцінювання результатів було використано кілька метрик бінарної класифікації. Accuracy показує загальну частку правильно визначених записів. Precision дозволяє оцінити, наскільки точними були прогнози у випадках, коли

модель відносила пацієнта до групи ризику. Recall характеризує здатність моделі знаходити пацієнтів, які дійсно належать до ризикової групи. F1-score використовується як узагальнений показник, що поєднує precision і recall та дає змогу оцінити баланс між ними.

Після підготовки вибірок було виконано навчання XGBoost-моделі та перевірку її якості на тестових даних. У результаті програмний сценарій вивів основні параметри експерименту: загальну кількість записів у датасеті, кількість використаних ознак, обсяг навчальної та тестової вибірок, а також значення accuracy, precision, recall і F1-score. Такий формат подання результатів дозволяє простежити повний шлях оцінювання моделі, від структури вхідних даних до кількісної характеристики якості прогнозування. Отримані результати наведено на рисунку 3.1.

```
        STARTING XGBOOST MODEL TRAINING
============================================================
Dataset records:      600
Number of features:   8
Train samples:        480
Test samples:         120
------------------------------------------------------------
Training model...
Training completed successfully.
Testing model on test sample...
------------------------------------------------------------
        MODEL EVALUATION RESULTS
------------------------------------------------------------
Accuracy:             90.00%
Precision:            91.67%
Recall:               88.71%
F1-score:             90.16%
------------------------------------------------------------
Model saved successfully: xgboost_model.joblib
============================================================

Process finished with exit code 0
```

Рисунок 3.1 – Результати навчання та оцінювання якості моделі XGBoost

Як видно з рисунка, модель була навчена на 480 записах, а оцінювання виконувалося на 120 тестових записах. Отримані значення accuracy, precision, recall та F1-score свідчать про здатність моделі виконувати бінарну класифікацію

пацієнтів за рівнем серцево-судинного ризику. Значення accuracy 90% підтверджує достатню якість моделі для подальшого використання у складі предиктивного модуля.

Для зменшення ризику потрапляння некоректної інформації до аналітичного модуля у системі передбачено багаторівневу перевірку даних. На рівні клієнтського інтерфейсу React-компоненти форми контролюють введення значень за допомогою обробників подій onChange. Такий механізм не допускає передавання нечислових символів у поля, призначені для клінічних або антропометричних параметрів. Якщо введене значення виходить за встановлені межі, відповідне поле позначається як помилкове, а відправлення запиту на сервер блокується.

Попередня валідація має не лише технічне, а й аналітичне значення. Модель машинного навчання формує прогноз на основі вхідного вектора, тому помилки у значеннях тиску, глюкози, холестерину або BMI можуть змінити підсумкову оцінку ризику. Контроль коректності на етапі введення дозволяє зменшити обсяг інформаційного шуму та підвищити стабільність роботи XGBoost-модуля.

Завершальним кроком підготовки вхідних даних є додавання бінарних ознак, які описують поведінкові фактори. До таких параметрів належать наявність шкідливих звичок і рівень фізичної активності. Попри певну суб’єктивність цих даних, у поєднанні з клінічними показниками вони підвищують здатність моделі виявляти приховані ризики. Наприклад, пацієнт може мати відносно нормальні лабораторні результати, однак наявність куріння або низької фізичної активності може змінювати загальну оцінку прогнозу.

Після формування повного набору ознак створюється багатовимірний вектор, який передається до модуля машинного навчання. У межах розроблюваної системи для реалізації предиктивного аналізу використано алгоритм XGBoost. Вибір цього методу пов’язаний із його здатністю ефективно працювати зі структурованими табличними даними, враховувати нелінійні

залежності та забезпечувати високу точність класифікації у задачах із неоднорідними вхідними параметрами.

Після оцінювання якості моделі доцільно розглянути, за рахунок яких алгоритмічних механізмів XGBoost забезпечує таку якість класифікації. На відміну від простого фіксованого правила, модель формує прогноз як результат роботи ансамблю дерев рішень, кожне з яких поступово уточнює попередню оцінку.

Принцип роботи XGBoost базується на побудові ансамблю дерев рішень. Кожне окреме дерево виконує часткову оцінку, а підсумковий прогноз формується через поєднання результатів усіх дерев. На відміну від окремого дерева рішень, ансамбль має вищу стійкість до випадкових відхилень у даних та краще узагальнює закономірності, виявлені під час навчання. Для медичного прогнозування така властивість є важливою, оскільки реальні клінічні записи часто містять пропуски, неточності або нетипові комбінації показників.

Алгоритмічна логіка XGBoost базується на послідовному уточненні прогнозу. Початкове дерево формує базову оцінку, після чого кожна наступна структура додається з урахуванням помилок попередніх етапів. Оптимізація виконується через мінімізацію функції втрат, яка у даній задачі відповідає логіці бінарної класифікації. Такий підхід дозволяє моделі поступово уточнювати межу між групами пацієнтів із низьким і підвищеним ризиком, концентруючись на складних для класифікації випадках.

Медичні набори даних можуть містити нетипові записи або випадкові коливання показників, тому без додаткового обмеження складності модель може занадто точно пристосуватися до навчальної вибірки. Регуляризація зменшує ймовірність побудови надмірно глибоких дерев і перешкоджає формуванню правил, які описують лише поодинокі випадки без достатнього клінічного обґрунтування.

Структурно модель можна розглядати як систему послідовних умов, які перевіряють значення вхідних ознак. На верхніх рівнях дерева зазвичай розташовуються параметри з найбільшим внеском у розділення класів.

Наприклад, підвищений артеріальний тиск може спрямовувати подальший аналіз до гілки, де додатково враховуються вік пацієнта, рівень холестерину або глюкози. На етапі налаштування було підібрано ключові гіперпараметри, зокрема max_depth та learning_rate. Максимальна глибина дерева визначає складність окремих правил, а швидкість навчання контролює внесок кожного нового дерева до ансамблю. Підбір цих параметрів дозволив збалансувати точність прогнозування та обчислювальну ефективність.

Для збереження навченої моделі застосовано формат серіалізації joblib. Такий спосіб дозволяє швидко завантажувати предиктивне ядро під час обробки запитів від серверної частини. Завдяки попередньому збереженню навченої моделі відпадає потреба у повторному навчанні або повній ініціалізації алгоритму під час кожного звернення, що зменшує час відповіді системи. Це також забезпечує стабільність використання однакової версії моделі під час багаторазового виконання прогнозів.

Окремою вимогою до медичних інтелектуальних систем є пояснюваність отриманих результатів. Прогноз не повинен бути лише числовим значенням, оскільки лікарю необхідно розуміти, які саме показники вплинули на підсумкову оцінку ризику. Для підвищення прозорості роботи моделі у межах дослідження виконано візуалізацію структури одного з дерев рішень, що входить до ансамблю XGBoost [34]. Такий підхід дає змогу наочно простежити логіку прийняття рішення моделлю та визначити роль окремих ознак у процесі класифікації. Крім того, візуальне подання дерева рішень полегшує перевірку коректності роботи моделі на етапі аналізу отриманих результатів. У практичному застосуванні це спрощує інтеграцію моделі з серверною частиною, оскільки аналітичний сервіс працює вже з готовим об’єктом прогнозування. Крім того, серіалізація дозволяє зберігати не лише параметри моделі, а й її внутрішню структуру, сформовану під час навчання. Це забезпечує відтворюваність результатів за однакових вхідних даних. У свою чергу, візуалізація дерева рішень може використовуватися як додатковий засіб контролю, що підтверджує логічність побудови моделі та відповідність її роботи поставленому завданню.

На рисунку 3.2 продемонстровано логічну архітектуру одного з дерев рішень, згенерованого на основі підготовленого вектора ознак.

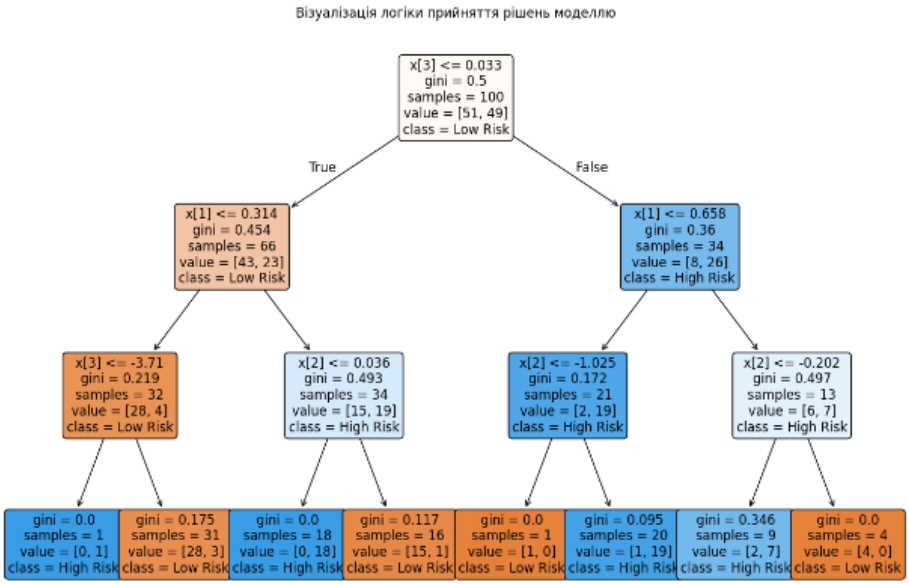

Рисунок 3.2 – Візуалізація логічної структури дерева рішень предиктивного модуля

Аналіз візуалізованого дерева рішень дає змогу простежити порядок використання ознак під час класифікації. Кореневий вузол містить параметр, який має найбільшу здатність розділяти пацієнтів за рівнем ризику. Подальші вузли формують послідовність умов, що перевіряють значення клінічних показників, зокрема артеріального тиску, віку або лабораторних маркерів. Кожне розгалуження відповідає одному з можливих варіантів виконання умови та веде до наступного етапу оцінювання.

У візуальному поданні дерева використовується колірне кодування та показник неоднорідності Gini impurity. Вузли, пов’язані з нижчим ризиком, позначаються теплими відтінками, а вузли з підвищеною ймовірністю патології, холодними. Насиченість кольору відображає ступінь впевненості моделі. Чим нижчим є значення Gini, тим більш однозначним вважається розподіл пацієнтів у відповідному вузлі.

Листкові вузли містять підсумкові вагові значення, які надалі враховуються під час формування загального прогнозу ансамблю. Така структура демонструє, що алгоритм не обмежується механічним присвоєнням класу, а виконує послідовне оцінювання медичних параметрів.

Фінальний етап розробки інтелектуального модуля передбачав програмну реалізацію предиктивного ядра. Основною метою було не лише отримання математичного класифікатора, а створення програмного компонента, здатного генерувати практично придатний результат для лікаря. Тому модуль поєднує розрахунок ймовірності ризику, визначення клінічного статусу та формування рекомендацій залежно від отриманої оцінки.

Центральною функцією модуля є run_diagnostic. Вона координує приймання вхідного вектора, запуск моделі, отримання ймовірнісної оцінки та підготовку результату до передавання у серверну частину. Побудований ансамбль зі 100 дерев рішень не обмежується віднесенням пацієнта до одного з класів, а формує показник Risk Prediction у відсотковому вигляді. Такий формат дозволяє деталізувати рівень небезпеки навіть у межах однієї категорії ризику.

Для розрахунку ймовірності використовується метод predict_proba. Завдяки такому підходу модель формує не жорстке рішення, а розподіл імовірностей між класами. Для медичної практики така форма результату є корисною, оскільки дозволяє розрізняти прикордонні випадки та ситуації з вираженим ризиком. Наприклад, значення 51 % і 89 % можуть належати до однієї категорії, але мають різний рівень клінічної терміновості.

Метод predict_proba повертає для кожного вхідного запису масив імовірностей, що відповідають класам моделі. Оскільки в межах даної роботи використовується бінарна класифікація, результат має два значення: імовірність належності пацієнта до класу без підвищеного ризику та ймовірність належності до класу підвищеного ризику. Для подальшої інтерпретації системою використовується саме друге значення, тобто ймовірність позитивного класу.

Наприклад, якщо метод predict_proba повертає значення [0.18; 0.82], перше число означає 18 % імовірності належності до класу низького ризику, а друге

число, 82 % імовірності належності до класу підвищеного ризику. Саме значення 0,82 перетворюється у відсотковий формат і відображається в системі як Risk Prediction = 82 %. Таким чином, модель не змінює постановку задачі як бінарної класифікації, а лише надає кількісну оцінку впевненості у належності пацієнта до ризикового класу.

Отримана ймовірність використовується для визначення рівня пріоритетності результату. Низькі значення інтерпретуються як low risk, проміжні, як medium risk, а високі, як high risk. Завдяки такому підходу лікар отримує не лише факт класифікації, а й числовий показник, який дозволяє оцінити ступінь вираженості ризику та порівнювати пацієнтів у межах однієї категорії.

Для підвищення прикладної цінності системи до предиктивного модуля додано базу клінічних рекомендацій. Вона активується залежно від отриманого рівня ризику [35]. У системі передбачено три рівні пріоритетності: low risk, medium risk та high risk. Для низького ризику формуються поради щодо підтримання здорового способу життя. Для середнього ризику акцент робиться на корекції харчування, регулярному контролі показників і підвищенні фізичної активності. Для високого ризику генерується рекомендаційний протокол із необхідністю консультації кардіолога та проходження додаткових обстежень, зокрема ЕКГ і ліпідного профілю.

Додатково реалізовано механізм контекстних попереджень. Якщо окремий показник виходить за критичну межу, система формує спеціальне повідомлення незалежно від загального інтегрального ризику. Наприклад, при систолічному тиску понад 140 мм рт. ст. генерується попередження щодо необхідності контролю гемодинамічних параметрів. Такий механізм запобігає ситуації, коли окреме клінічно важливе відхилення залишається непоміченим через усереднений результат моделі.

Після навчання та оцінювання якості моделі було виконано контрольний запуск аналітичного Python-модуля на прикладі окремого пацієнта. Такий запуск демонструє не процес навчання, а практичне використання вже навченої моделі

для формування індивідуального прогнозу. До вхідного набору були включені вік 55 років, артеріальний тиск 155/95, підвищена маса тіла та наявність тютюнопаління. На основі цих параметрів модуль сформував відповідь із відсотковим значенням ризику, клінічним статусом і набором рекомендацій.

На рисунку 3.3 подано результат контрольного запуску аналітичного Python-модуля, який є частиною загальної інтелектуальної системи. Після обробки вхідного вектора ознак модуль формує структуровану відповідь, що надалі передається на сервер Node.js у форматі JSON та використовується для відображення фінального результату у веб-інтерфейсі.

```
          CARDIOVASCULAR RISK ANALYSIS REPORT
================================================
Risk Prediction: 89.09%
Clinical Status: HIGH RISK
Patient BMI:     31.83
------------------------------------------------
STATUS: URGENT MEDICAL ATTENTION REQUIRED
 1. Immediate cardiologist consultation for comprehensive diagnostics.
 2. Mandatory ECG and lipid profile blood tests.
 3. Strict sodium (salt) intake reduction and BP monitoring twice daily.
 4. Pharmacological intervention according to clinical guidelines.

[!] ALERT: High Systolic Pressure (155) is a primary risk factor.
================================================
```

Рисунок 3.3 – Результат контрольного запуску аналітичного Python-модуля

Отриманий результат підтверджує коректність роботи предиктивного модуля в режимі обробки одиничного запиту. Система не лише розраховує відсоткову ймовірність ризику, а й визначає клінічний статус пацієнта, активує відповідний блок рекомендацій та формує контекстне попередження щодо критичного значення систолічного артеріального тиску. Формат JSON, у якому повертається результат, спрощує інтеграцію Python-модуля із серверною частиною на Node.js. Завдяки такій організації забезпечується зв’язок між клієнтським інтерфейсом, backend-рівнем та інтелектуальним ядром системи.

3.2 Розробка серверної частини системи на базі платформи Node.js та інтеграція з ML-модулем

Розробка серверного рівня системи розпочалася з налаштування середовища Node.js, яке забезпечує виконання JavaScript-коду поза межами браузера. Такий вибір обумовлений потребою в організації швидкої обробки HTTP-запитів, взаємодії з базою даних та координації обміну між клієнтською частиною і модулем машинного навчання. Для керування залежностями проєкту використано менеджер пакетів NPM, що дозволило зафіксувати перелік необхідних бібліотек і їх версій у файлі package.json. Така організація спрощує повторне розгортання системи в іншому середовищі та забезпечує стабільність конфігурації.

Основним фреймворком для побудови API обрано Express.js. Застосування пояснюється невеликою кількістю надлишкових компонентів, гнучкістю маршрутизації та достатньою продуктивністю для обробки запитів у режимі реального часу. Для медичної інформаційної системи швидкість відповіді серверної частини має важливе значення, оскільки затримки під час отримання прогнозу або роботи з реєстром пацієнтів можуть ускладнювати практичне використання програмного комплексу [36].

Взаємодія з реляційною базою даних MySQL організована через ORM Sequelize. Використання такого інструмента дозволяє працювати з таблицями бази даних на рівні програмних моделей, не формуючи кожен SQL-запит вручну.

Завдяки цьому зменшується навантаження на СУБД, а серверна частина може стабільно обробляти кілька паралельних звернень [37]. Серверна частина побудована за модульним принципом.

Окремі функціональні блоки відповідають за підключення до бази даних, опис моделей, маршрутизацію запитів, виконання бізнес-логіки та взаємодію з ML-ядром. Такий розподіл полегшує супровід проєкту, оскільки зміни в одному модулі не потребують повної перебудови інших частин системи. Крім того, подібна структура спрощує тестування й дає змогу масштабувати окремі

елементи залежно від навантаження. В таблиці 3.2 наведено опис ієрархії каталогів та призначення функціональних модулів.

Таблиця 3.2 – Опис ієрархії каталогів та призначення функціональних модулів

Каталог / Файл	**Призначення в системі**	**Технологічний стек**
Config/db.js	Налаштування зв'язку з MySQL та параметри пулу	Sequelize / mysql2
Models/diagnosis.js	Опис SQL-схеми та типів даних результатів аналізу	Sequelize Models
Controllers/	Бізнес-логіка, запуск ML-ядра та обробка помилок	Express Controllers
routes/	Визначення API-маршрутів та рівнів доступу (JWT)	Express Router
ml_core/	Розміщення навченої моделі та предиктивних скриптів	Python / XGBoost

Ключову роль у серверній частині відіграє рівень контролерів. Саме контролери приймають запити від клієнтського застосунку, перевіряють вхідні параметри, запускають необхідні операції та формують відповідь для інтерфейсу користувача. Для реалізації асинхронної логіки застосовується механізм async/await, який дозволяє виконувати операції введення-виведення без блокування основного циклу подій Node.js. Завдяки такій організації сервер може одночасно працювати з кількома запитами, зокрема обробляти звернення до реєстру пацієнтів і запускати процедури прогнозування.

Найбільш важливим елементом цього рівня є PredictionController, оскільки саме через нього здійснюється зв’язок між вебзастосунком і Python-модулем машинного навчання. Спочатку контролер отримує від клієнтської частини вектор клінічних та антропометричних показників. Далі виконується перевірка повноти й коректності параметрів, що дозволяє відсіяти помилкові запити до запуску ресурсоємного прогнозування. Після проходження перевірки сервер ініціює окремий дочірній процес, у межах якого запускається Python-скрипт із моделлю XGBoost.

Передавання даних до Python-середовища реалізовано через механізм spawn. Такий спосіб дозволяє запускати зовнішній процес і передавати йому необхідні параметри без зупинки основного серверного потоку. Після завершення обчислень ML-модуль повертає результат у форматі JSON, який містить оцінку ризику та набір рекомендацій. Контролер зчитує відповідь асинхронно, обробляє її та передає на наступний етап, де результат зберігається в базі даних.

Збереження прогнозу виконується через метод Diagnosis.create(). Така операція створює постійний запис у реляційній базі даних і прив’язує результат аналізу до відповідного пацієнта або клінічного обстеження. Завдяки такій логіці кожен сформований прогноз може бути повторно переглянутий, використаний для побудови історії спостережень або включений до аналітичного звіту.

Окремий контролер UserController відповідає за автентифікацію та контроль доступу до системи. Під час реєстрації медичного працівника пароль не зберігається у відкритому вигляді. Перед записом у базу даних пароль проходить криптографічне перетворення за допомогою bcrypt.

У результаті в таблиці користувачів зберігається лише хешоване значення, що значно знижує ризик компрометації облікових даних.

Під час авторизації введені користувачем дані порівнюються із хешем, який зберігається в базі. У разі успішної перевірки сервер формує JSON Web Token. Такий токен містить службову інформацію про користувача та строк дії сесії. Подальші звернення до API виконуються з передаванням токена, що

дозволяє серверу визначати особу користувача без повторного введення пароля. Такий механізм є зручним для роботи лікаря із захищеними сторінками системи та водночас підтримує контроль доступу до медичних даних.

HistoryController призначений для роботи з історією прогнозів і клінічних записів. Через можливості Sequelize контролер формує запити до бази даних, виконує пошук, фільтрацію та сортування результатів. Наприклад, медичний працівник може отримати записи за певний період, переглянути пацієнтів із конкретним рівнем ризику або проаналізувати останні обстеження. Для цього використовується метод findAll() із визначенням умов у параметрі where.

Крім отримання окремих записів, HistoryController може виконувати агрегацію даних для формування аналітичних звітів. Така функціональність потрібна для відстеження змін стану пацієнта в часі та перегляду динаміки результатів прогнозування.

Сортування записів за датою створення у спадному порядку забезпечує першочерговий доступ до найновіших результатів, що є зручним під час роботи лікаря з актуальними медичними даними.

Додаткові контролери відповідають за налаштування профілю, системні параметри та оновлення службової інформації. Під час виконання таких операцій важливо зберігати узгодженість даних і не допускати дублювання записів. Для цього застосовується валідація на рівні серверної логіки та бази даних. У складних операціях доцільним є використання транзакцій, які дозволяють скасувати зміни у разі помилки та повернути систему до попереднього стабільного стану.

Взаємодія серверної частини з базою даних MySQL організована через ORM Sequelize, що забезпечує більш структурований та безпечний спосіб роботи з реляційними даними. Замість безпосереднього формування SQL-запитів у коді, система оперує програмними моделями, які відповідають окремим таблицям бази даних. Такий підхід спрощує реалізацію операцій створення, читання, оновлення та видалення записів, а також зменшує ймовірність помилок під час роботи з медичною інформацією.

Використання Sequelize також дозволяє чітко описати типи полів, зв’язки між сутностями та правила валідації даних на рівні серверної логіки. Для медичної системи такий механізм має важливе значення, оскільки результати прогнозування повинні зберігатися у впорядкованому вигляді та бути пов’язаними з відповідними клінічними записами пацієнтів. Крім того, ORM-рівень підвищує зручність подальшого супроводу програмного продукту, оскільки зміни у структурі бази даних можна узгоджувати з моделями застосунку без повного переписування логіки доступу до даних. В таблиці 3.3 наведено опис атрибутів сутності "Result" у базі даних.

Таблиця 3.3 – Опис атрибутів сутності "Result" у базі даних

Атрибут (Column)	**Тип даних (SQL)**	**Призначення**
Id	int (Primary Key)	Унікальний ідентифікатор запису
Patient_age	int	Вік пацієнта
Risk_score	float	Розрахований відсоток ймовірності захворювання
Risk_status	varchar (20)	Категорія (Low, Medium, High)
Recommendations	text	Масив клінічних порад у серіалізованому форматі

Під час реалізації моделі було передбачено автоматичне створення часових міток createdAt. Наявність таких полів дозволяє відстежувати момент формування прогнозу та аналізувати зміну клінічного стану пацієнта протягом тривалого періоду. Для системи прогнозування медичних ризиків часова прив’язка має важливе значення, оскільки результати окремих обстежень набувають цінності саме в контексті динаміки.

Інтеграція Node.js-сервера з Python-модулем побудована на використанні дочірніх процесів. Така архітектурна схема дозволяє винести обчислення XGBoost за межі основного циклу подій Node.js. Завдяки цьому сервер не припиняє обробку інших HTTP-запитів під час виконання прогнозування. Для системи, яка може одночасно обслуговувати кількох користувачів, така ізоляція має суттєве значення.

Міжпроцесна взаємодія між Node.js та Python-ядром належить до ключових технічних аспектів реалізації. Node.js працює в асинхронному однопотоковому середовищі, тому тривалі математичні операції не повинні виконуватися безпосередньо в основному потоці. Для вирішення цього завдання використано модуль child_process, який забезпечує запуск окремого процесу для виконання Python-скрипта [38].

Застосування spawn відрізняється від синхронного запуску зовнішніх команд тим, що обмін даними організовується через потоки. Сервер може передати Python-скрипту необхідні параметри, продовжити обслуговування інших користувацьких запитів і отримати результат після завершення обчислень. Така схема зменшує ризик блокування серверної частини та підтримує стабільність роботи системи під час виконання ресурсоємних прогнозів.

Процедура предиктивного аналізу починається з отримання даних від клієнтського інтерфейсу. До вхідного набору входять вік, стать, зріст, маса тіла, систолічний і діастолічний артеріальний тиск, рівень глюкози, рівень холестерину, а також інформація про фізичну активність і шкідливі звички. Серверна частина об’єднує ці параметри в уніфікований вектор ознак, який відповідає структурі, очікуваній моделлю машинного навчання.

Далі сформований набір параметрів передається Python-інтерпретатору у вигляді аргументів командного рядка sys.argv. Після запуску скрипт завантажує попередньо навчену модель XGBoost, виконує необхідну підготовку вхідних значень і формує прогноз. Результатом роботи аналітичного модуля є ймовірність розвитку серцево-судинного ризику та набір рекомендацій, сформованих у форматі JSON [39].

Отримання результату з Python-середовища організовано через стандартний потік виводу stdout. Оскільки відповідь може складатися з кількох фрагментів, сервер поступово збирає отримані частини в єдину структуру. Одночасно контролюється потік stderr, у якому можуть з'являтися повідомлення про помилки, наприклад відсутність файлу моделі, некоректний формат параметрів або збій під час виконання Python-скрипта. Такий контроль дозволяє швидко виявляти технічні проблеми та не передавати користувачу некоректний результат.

Для підвищення надійності взаємодії з ML-модулем реалізовано механізм обмеження часу виконання. Якщо Python-процес працює довше за допустимий інтервал, сервер ініціює примусове завершення за допомогою сигналу SIGTERM і методу pythonProcess.kill(). Такий механізм захищає систему від зависання дочірніх процесів, накопичення зайвих обчислювальних задач і потенційного перевантаження сервера під час пікової активності користувачів [40].

Після успішного завершення обчислень результат повертається до основного контролера. Далі виконується фінальна перевірка отриманої структури, після чого ініціюється операція збереження через Sequelize. Дані записуються до таблиці diagnoses або пов'язаної сутності результатів із прив'язкою до відповідного пацієнта. Використання зовнішнього ключа забезпечує логічний зв'язок між прогнозом і медичним записом, на основі якого було виконано аналіз.

Завершальним етапом є формування HTTP-відповіді для клієнтської частини. Після успішного запису результатів у базу даних сервер повертає статус 200 OK та передає оброблені дані до React-інтерфейсу. На стороні клієнта отриманий прогноз використовується для візуалізації рівня ризику, відображення рекомендацій та подальшої роботи лікаря з результатами аналізу

3.3 Розробка клієнтської частини системи на базі React.js

Клієнтська частина інтелектуальної системи розроблена як односторінковий вебзастосунок на основі бібліотеки React.js. Такий формат

обрано з урахуванням характеру роботи медичного персоналу, оскільки лікар повинен швидко переходити між реєстром пацієнтів, аналітичною панеллю та модулем прогнозування без повного перезавантаження сторінок. Для системи, що працює з електронними медичними записами та результатами прогнозної моделі, швидкість оновлення інтерфейсу має не лише технічне, а й практичне значення, оскільки впливає на зручність прийняття клінічних рішень [41].

React.js використовується як основа для побудови компонентного інтерфейсу. Кожен функціональний елемент клієнтської частини виділяється в окремий компонент, що спрощує підтримку коду, повторне використання окремих блоків і подальше розширення системи. Така структура є доцільною для медичного вебзастосунку, оскільки різні сторінки використовують схожі елементи, зокрема панелі навігації, таблиці, форми введення даних, індикатори ризику та графіки.

Для організації вихідного коду застосовано ієрархію каталогів, у якій окремі папки відповідають за компоненти інтерфейсу, сторінки, хуки, стилі та основні файли застосунку. Подібний поділ дозволяє швидко орієнтуватися у структурі проєкту та зменшує ризик змішування логіки відображення з логікою обробки даних. Для подальшого супроводу системи така організація є важливою, оскільки зміни в окремому модулі не потребують перегляду всієї клієнтської частини. Для організації вихідного коду використана чітка ієрархія каталогів що зображена на рисунку 3.4.

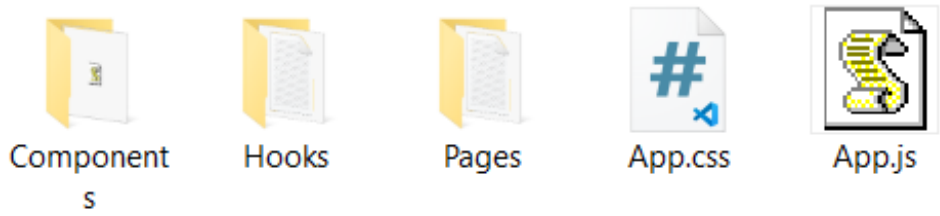

Рисунок 3.4 – Каталог клієнтської частини

Каталог Components містить набір елементів, з яких формується візуальна частина системи. До базових інтерфейсних компонентів належать Navbar.jsx і

Sidebar.jsx. Navbar.jsx відповідає за верхню панель застосунку, де розміщуються пошук пацієнтів, блок сповіщень та інформація про авторизованого користувача. Sidebar.jsx забезпечує навігацію між ключовими розділами, зокрема Dashboard, реєстром пацієнтів, аналітикою та сторінкою прогнозування ризику. Завдяки поділу навігації на окремі компоненти інтерфейс зберігає єдину структуру на різних сторінках системи.

AnalyticsChart.jsx відповідає за побудову графіків і діаграм, які допомагають оцінювати статистику роботи системи та динаміку прогнозів. Використання таких компонентів підвищує наочність результатів і дає змогу лікарю швидше сприймати великі обсяги інформації.

Функціональний блок прогнозування реалізується через компоненти PredictorForm.jsx і RiskIndicator.jsx. PredictorForm.jsx забезпечує введення антропометричних, лабораторних і поведінкових параметрів пацієнта, а також виконує попередню перевірку коректності значень перед передаванням інформації на сервер. RiskIndicator.jsx призначений для візуального подання розрахованого рівня ризику. Значення ризику відображається у відсотковому форматі та супроводжується відповідним кольоровим оформленням, що полегшує швидке розпізнавання критичних випадків.

Каталог Context використовується для розміщення логіки керування станом і взаємодії з даними. Хук usePrediction відповідає за запуск запитів до ML-модуля, обробку стану завантаження та фіксацію можливих помилок під час отримання відповіді від сервера. Хук useAuth контролює автентифікацію користувача через JWT-токен і обмежує доступ до захищених сторінок. Функція formatRiskData виконує перетворення числового прогнозу у формат, зручний для відображення в інтерфейсі. Каталог Pages об’єднує основні сторінки системи. До них належать сторінка аналітичної панелі Dashboard, модуль роботи з реєстром пацієнтів та сторінка прогнозування ризику захворювань [42]. На рівні сторінок окремі компоненти поєднуються у завершені користувацькі сценарії. Наприклад, сторінка прогнозування об’єднує пошук пацієнта, коротку картку підтвердження, форму введення клінічних даних і блок відображення результатів

аналізу. Описана структура клієнтської частини забезпечує логічний поділ інтерфейсу на незалежні функціональні блоки. Компонентний підхід спрощує повторне використання елементів, підтримує єдиний стиль оформлення та полегшує розширення системи новими модулями.

Кожна сторінка системи побудована як комбінація React-компонентів, контекстних хуків і API-запитів. Така організація забезпечує швидку взаємодію з медичними записами, доступ до інструментів прогнозування та динамічне оновлення результатів без перезавантаження сторінки. Для медичного персоналу такий режим роботи є зручним, оскільки основні операції виконуються в межах одного робочого середовища.

Нижче наведено візуалізацію ключових екранних форм програмного продукту та виконано аналіз їх функціональних можливостей. На рисунку 3.5 зображено головну аналітичну панель системи Dashboard, яка використовується як центральний екран для моніторингу активності системи та перегляду узагальнених показників.

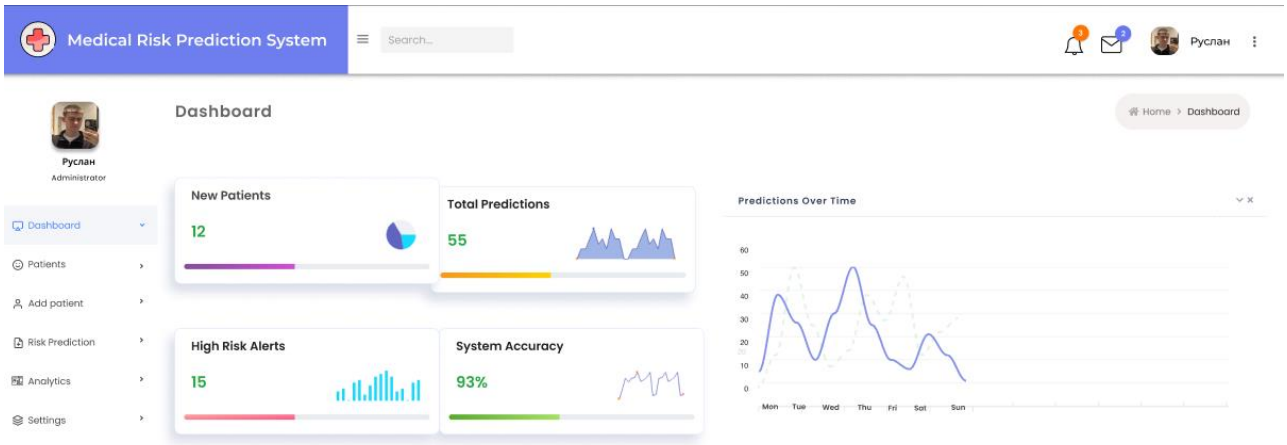

Рисунок 3.5 – Головна аналітична панель системи

Аналітична панель Dashboard виконує роль стартового інформаційного простору для лікаря. У верхній частині інтерфейсу розміщено статистичні віджети, які показують кількість нових пацієнтів, загальну кількість проведених аналізів, рівень активності системи та перелік випадків із підвищеним ризиком.

Такий формат подання дозволяє швидко оцінити поточне навантаження та визначити, які записи потребують першочергової уваги [43]. Центральна частина Dashboard містить інтерактивний графік, призначений для відображення динаміки обстежень і прогнозів. За допомогою графічного подання медичний персонал може простежувати активність системи протягом певного періоду, виявляти дні з підвищеною кількістю звернень і оцінювати загальні тенденції. Дані на панелі оновлюються через асинхронні HTTP-запити, тому користувач отримує актуальну інформацію без повного перезавантаження сторінки.

Окрім статистичних віджетів і графіків, у загальну структуру інтерфейсу інтегровано доступ до реєстру пацієнтів. Такий модуль забезпечує перехід від загальної аналітики до конкретних медичних записів. Лікар може швидко знайти потрібного пацієнта, переглянути основні відомості та перейти до історії попередніх обстежень або запуску прогнозування. У межах інтерфейсу така організація забезпечує зв’язок між аналітичним оглядом і практичною роботою з конкретними клінічними даними. На рисунку 3.6 зображено сторінку управління реєстром пацієнтів

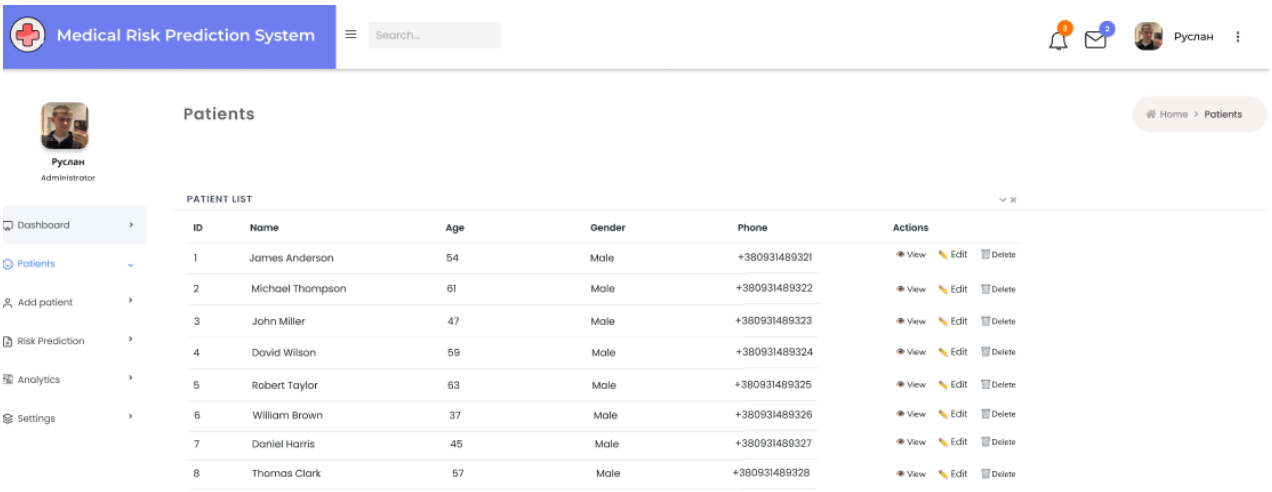

Рисунок 3.6 – Сторінка управління реєстром пацієнтів

Сторінка реєстру пацієнтів реалізована у вигляді інтерактивної таблиці. У ній подаються основні атрибути електронної медичної картки, зокрема унікальний ідентифікатор, прізвище та ім’я, вік, стать і контактна інформація.

Дані надходять із серверної частини через асинхронні запити до Node.js API, яке взаємодіє з реляційною базою даних. У правій частині таблиці передбачено блок дій для роботи з окремими записами. Розміщення основних операцій безпосередньо в таблиці скорочує кількість переходів між сторінками та робить роботу з реєстром швидшою. Пошук і фільтрація записів виконуються динамічно за допомогою механізмів керування станом React [44].

Функціональним центром системи є сторінка Risk Prediction. Саме в межах цієї сторінки відбувається повний сценарій прогнозування, від вибору пацієнта до отримання результату аналітичного модуля. Сторінка забезпечує зв'язок між фронтенд-рівнем і Python-сервісом через серверну частину, яка приймає дані, передає їх до ML-моделі та повертає результат у форматі, придатному для візуального подання. Початковим етапом роботи з модулем прогнозування є вибір пацієнта. Для цього використовується спеціалізований компонент інтерактивного пошуку. На рисунку 3.7 зображено компонент пошуку пацієнтів.

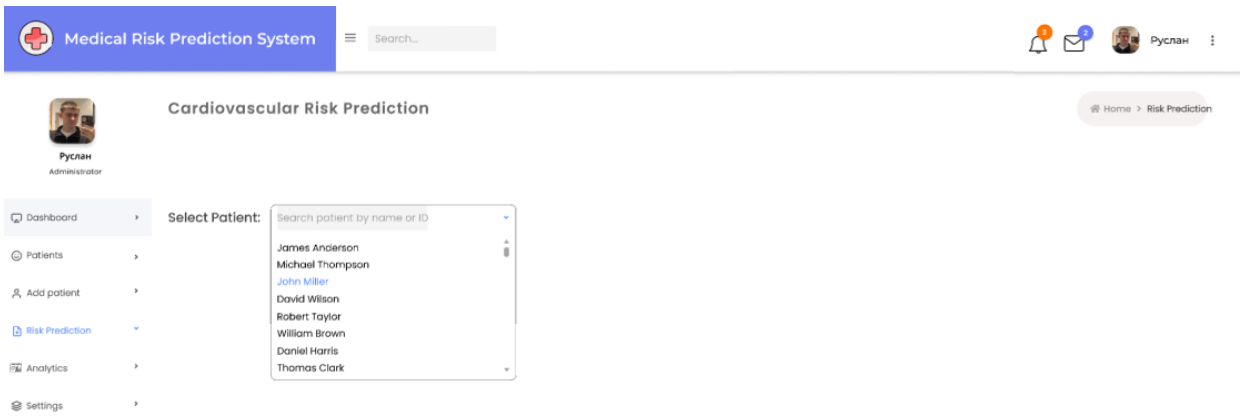

Рисунок 3.7 – Реалізація компонента інтерактивного пошуку пацієнта

Компонент пошуку реалізований за принципом Smart Search Dropdown. Під час введення перших символів прізвища або ідентифікаційного номера виконується фільтрація записів і формується список релевантних результатів.

Такий механізм скорочує час пошуку потрібного пацієнта та зменшує потребу у ручному перегляді великої кількості записів.

Точність вибору пацієнта має важливе значення для подальшого прогнозування. Помилковий вибір профілю може призвести до аналізу неправильних клінічних даних і формування некоректного результату. Застосування інтерактивного пошуку знижує такий ризик, оскільки користувач бачить релевантні записи вже під час введення пошукового запиту.

Після вибору пацієнта активується блок Patient Summary. У ньому відображається коротка картка з основними персональними та медичними даними. Такий етап використовується для остаточного підтвердження правильності вибраного профілю перед введенням клінічних показників і запуском процедури прогнозування. На рисунку 3.8 представлено візуалізацію картки підтвердження обраного пацієнта.

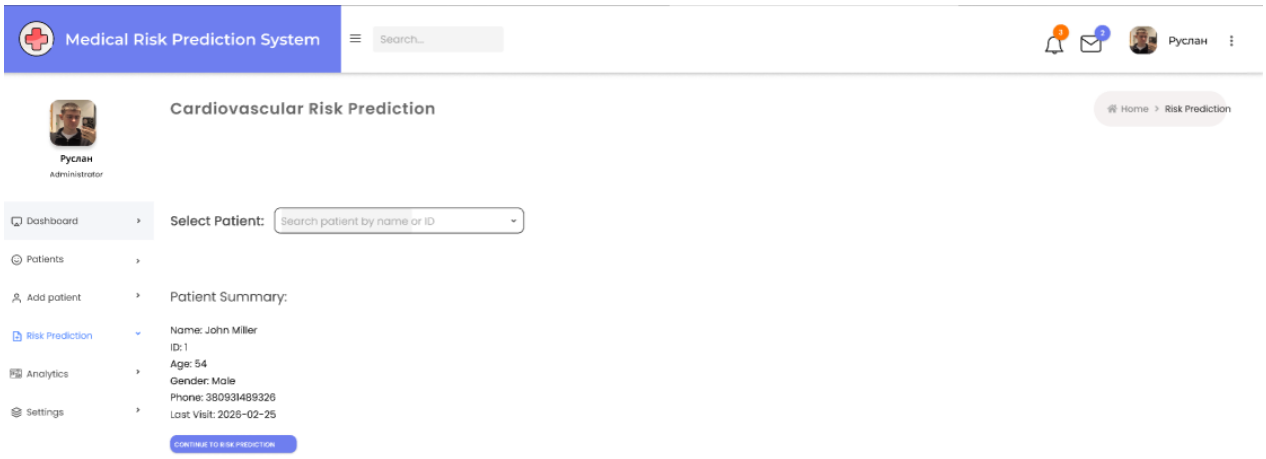

Рисунок 3.8 – Візуалізація картки підтвердження обраного пацієнта

Після підтвердження пацієнта користувач переходить до заповнення цифрового протоколу Clinical Data. Ця форма призначена для введення клінічних і фізіологічних показників, які надалі використовуються для формування вектора

ознак. До таких параметрів належать вік, стать, зріст, маса тіла, артеріальний тиск, рівень холестерину, рівень глюкози, факт куріння та фізична активність.

Під час проєктування форми основна увага приділялася коректності вхідної інформації. Кожне поле має власну логіку перевірки, що враховує тип даних і допустимий діапазон значень. Артеріальний тиск поділяється на два окремі показники, систолічний і діастолічний, що дозволяє точніше описати гемодинамічний стан пацієнта. Для лабораторних параметрів, зокрема глюкози та холестерину, застосовуються обробники onChange, які контролюють введення в режимі реального часу [45].

У разі введення некоректного значення система активує візуальне попередження та не дозволяє передати дані на сервер до усунення помилки. Така перевірка зменшує ймовірність потрапляння інформаційного шуму до ML-модуля та підвищує стабільність результатів прогнозування. На рисунку 3.9 наведено реалізацію форми введення клінічних діагностичних показників.

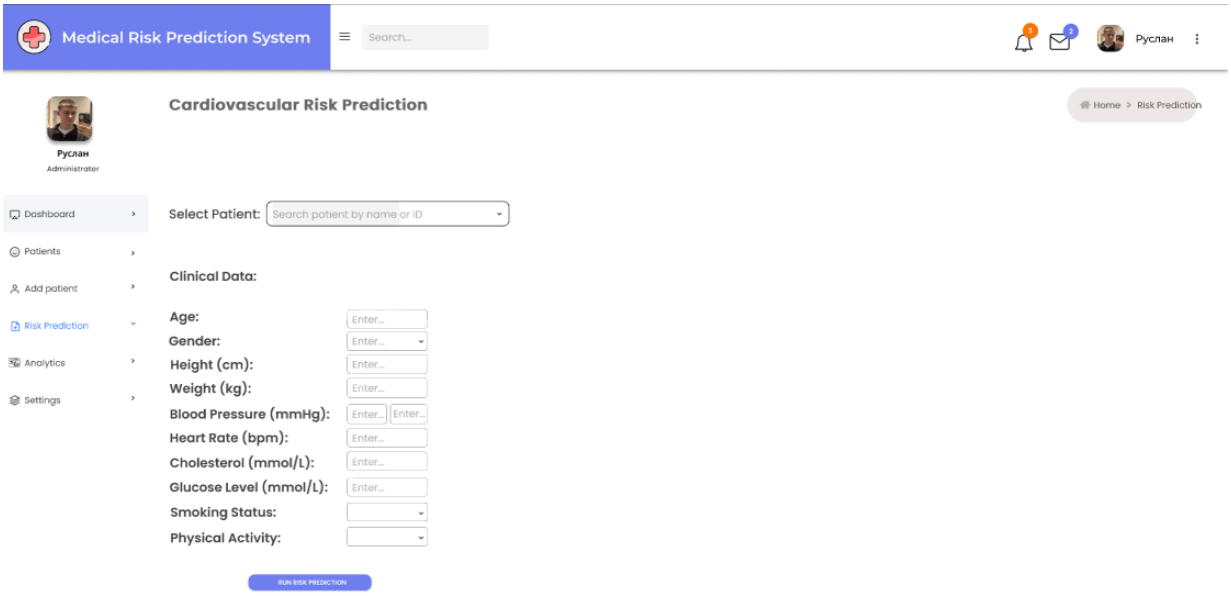

Рисунок 3.9 – Реалізація форми введення клінічних діагностичних показників

Після підтвердження введених даних формується JSON-об’єкт, який передається до серверної частини. Backend-рівень приймає набір параметрів,

виконує додаткову перевірку та передає дані до Python-сервісу, де завантажена модель машинного навчання виконує прогнозування. Така послідовність забезпечує розмежування функцій між інтерфейсом, сервером і аналітичним ядром.

На рисунку 3.10 зображено результат аналізу з розрахунком ймовірності ризику.

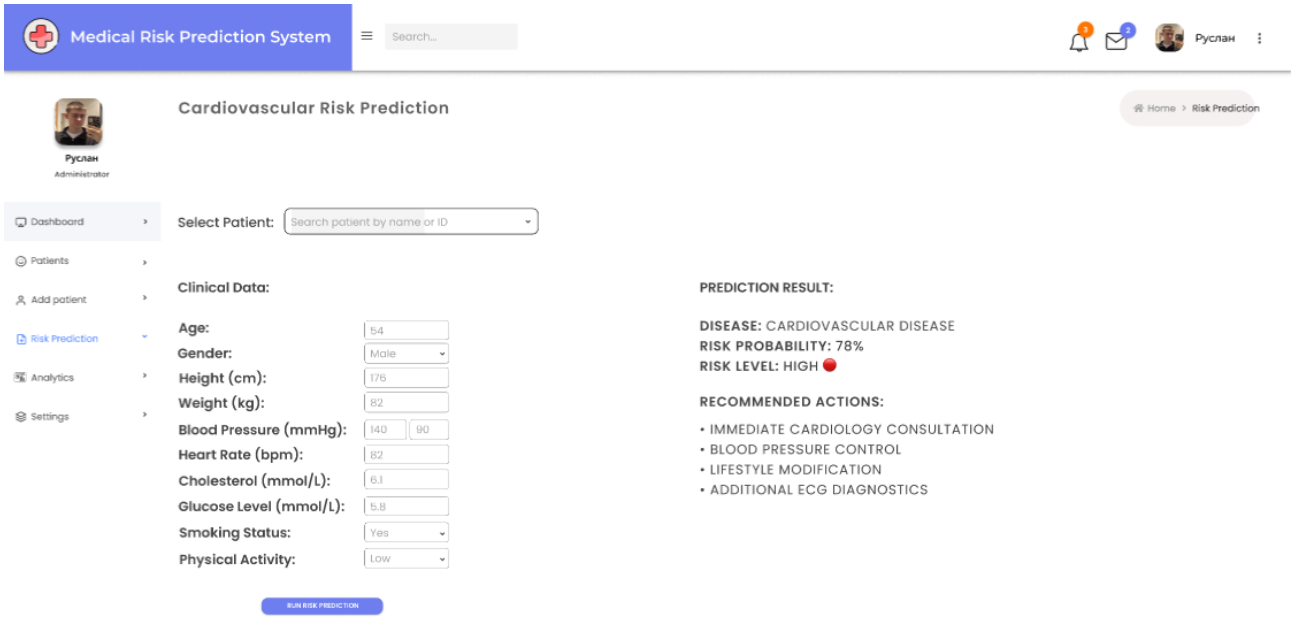

Рисунок 3.10 – Результат аналізу з розрахунком ймовірності ризику

Результат прогнозування подається у вигляді звіту, який містить відсоткову оцінку ризику та персоналізовані рекомендації. Такий формат дозволяє лікарю швидко визначити рівень потенційної небезпеки та прийняти рішення щодо подальших дій. Відсоткове подання є зручним для клінічного сприйняття, оскільки дає змогу порівнювати пацієнтів не лише за категоріями, а й за ступенем вираженості ризику.

Візуальна частина звіту доповнюється індикаторами, які змінюють вигляд залежно від рівня ризику. Для низького, середнього та високого значення можуть застосовуватися різні кольорові позначення, що прискорює сприйняття результату. Текстові рекомендації формуються на основі аналізу відхилень

окремих фізіологічних параметрів від нормативних значень. У разі підвищеного артеріального тиску система може запропонувати консультацію кардіолога або додаткове обстеження. При високому рівні глюкози формується попередження щодо можливих метаболічних порушень. Завдяки такій логіці звіт не обмежується числовим прогнозом, а доповнюється практично корисними поясненнями для подальшої роботи лікаря.

3.4 Висновки до третього розділу

У третьому розділі кваліфікаційної роботи було розглянуто практичну реалізацію інтелектуальної системи прогнозування медичних ризиків. Описано процес підготовки вхідних медичних даних, формування вектора ознак, розробку предиктивного модуля на основі алгоритму XGBoost, реалізацію серверної частини на Node.js та створення клієнтського інтерфейсу з використанням React.js. Основна увага приділялася не лише окремим програмним компонентам, а й організації їхньої взаємодії в межах єдиного програмного комплексу.

У процесі розробки моделі було використано структурований датасет кардіологічного спрямування, що містив антропометричні, гемодинамічні, лабораторні та поведінкові ознаки пацієнтів. Оцінювання якості моделі виконувалося за метриками accuracy, precision, recall та F1-score. Отримане значення accuracy понад 90 % підтвердило придатність XGBoost для задачі бінарної класифікації серцево-судинного ризику.

Серверна частина системи забезпечила взаємодію між базою даних MySQL, клієнтським інтерфейсом і Python-модулем машинного навчання. Для запуску аналітичного ядра використано механізм дочірніх процесів, що дозволило ізолювати ресурсоємні обчислення від основного циклу подій Node.js. Результати роботи ML-модуля передаються у форматі JSON, зберігаються у базі даних і надалі використовуються для відображення у веб-інтерфейсі.

Під час контрольного запуску на прикладі пацієнта з підвищеним ризиком система коректно сформувала відсоткову оцінку ризику та надала рекомендації.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Нормативні вимоги до організації робочого місця користувача медичної інформаційної системи

Організація безпечних умов праці під час використання медичної інформаційної системи має важливе значення, оскільки робота користувача пов’язана з тривалим застосуванням комп’ютерної техніки, аналізом електронної медичної документації та переглядом результатів інтелектуального прогнозування. У межах розробленої системи основним користувачем виступає лікар або медичний працівник, який взаємодіє з реєстром пацієнтів, формами введення клінічних показників, аналітичними панелями та звітами про рівень ризику.

Нормативною основою для опису умов праці є ДСТУ 2293:2014 «Охорона праці. Терміни та визначення основних понять», який унормовує українські терміни у сфері охорони праці та призначений для застосування під час розроблення і перевірки нормативних документів у цій сфері [46]. Для роботи з екранними пристроями основним документом є НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями», затверджений наказом Міністерства соціальної політики України від 14.02.2018 № 207 [55]. Також під час аналізу умов роботи враховуються ДСН 3.3.6.042-99 щодо мікроклімату виробничих приміщень та ДСанПіН 3.3.2.007-98, який поширювався на умови й організацію праці при роботі з візуальними дисплейними терміналами.

Робоче місце користувача медичної інформаційної системи повинно бути організоване так, щоб забезпечити зручне положення тіла, стабільну роботу обладнання та зменшення зорового навантаження. Оскільки лікар працює з таблицями, графіками, формами введення даних та кольоровими індикаторами ризику, неправильне розташування обладнання може призвести до швидкої втоми, зниження концентрації та підвищення ймовірності помилок під час роботи з медичною інформацією [56].

До основних параметрів робочого середовища користувача медичної інформаційної системи належать:

– відстань від очей до екрана, рекомендоване значення становить 50-70 см. Дотримання такої відстані відповідає вимогам безпечної роботи з екранними пристроями та дозволяє зменшити зорове напруження під час перегляду електронних медичних записів, графіків і результатів прогнозування;

– освітленість робочої зони, оптимальним є рівень 300-500 **лк**. Достатнє освітлення необхідне для комфортної роботи з комп'ютером, особливо під час аналізу таблиць, дрібного тексту, медичних показників та кольорових індикаторів ризику;

– температура повітря, рекомендований діапазон становить 20-24° С відповідно до вимог ДСН 3.3.6.042-99 щодо мікроклімату виробничих приміщень. Підтримання стабільної температури сприяє збереженню працездатності користувача протягом робочого дня;

– відносна вологість повітря, оптимальним є рівень 40-60 %, що також узгоджується з вимогами ДСН 3.3.6.042-99. Недостатня вологість може сприяти сухості очей і підвищенню втоми під час тривалої роботи з екраном;

– рівень шуму не вище 50 дБ. Помірний шумовий фон є важливою умовою для підтримання концентрації під час роботи з медичною інформацією та прийняття рішень на основі результатів аналітичної системи;

– тривалість безперервної роботи з екраном, рекомендовано обмежувати інтервалом 45-60 хв, після чого доцільно робити коротку перерву. Такий режим спрямований на профілактику зорової та загальної перевтоми;

– перерва під час тривалої роботи, рекомендована тривалість становить 10-15 хв**.** У цей час доцільно змінити положення тіла, виконати вправи для очей, кистей, шиї та плечового поясу, що дозволяє зменшити статичне навантаження.

Крім параметрів робочого середовища, важливе значення має правильне розміщення комп'ютерного обладнання. Монітор повинен розташовуватися перед користувачем, без повороту корпусу або голови вбік. Верхня межа екрана має бути на рівні очей або трохи нижче. Клавіатура та миша повинні

розміщуватися так, щоб кисті рук перебували у природному положенні, без надмірного згинання зап’ясть. Робоче крісло повинно мати підтримку попереку та можливість регулювання висоти.

Особливу увагу необхідно приділяти організації простору навколо робочого місця. Кабелі живлення та мережеві дроти не повинні створювати перешкод для пересування працівника. Системний блок, монітор, клавіатура, миша та інші периферійні пристрої мають бути розташовані так, щоб користувач міг виконувати основні дії без зайвих рухів і перенапруження. У разі появи запаху гару, перегріву обладнання, іскріння або нестабільної роботи електропристроїв експлуатацію необхідно припинити та повідомити відповідальну особу.

4.2 Профілактика професійних захворювань під час тривалої роботи з екранними пристроями

Тривала робота з комп’ютеризованими медичними системами може супроводжуватися впливом факторів, що негативно позначаються на стані здоров’я користувача. Медичний працівник під час використання розробленої системи переглядає електронні картки пацієнтів, вводить клінічні показники, аналізує результати прогнозування та приймає рішення щодо подальших дій. Такий вид діяльності потребує високої концентрації уваги, точності введення даних і постійної взаємодії з екраном.

Відповідно до НПАОП 0.00-7.15-18, під час роботи з екранними пристроями необхідно враховувати вимоги безпеки та захисту здоров’я працівників [57]. Для розробленої медичної системи зазначені вимоги мають практичне значення, оскільки користувач може працювати з інтерфейсом протягом тривалого часу, переглядаючи таблиці, аналітичні графіки, кольорові індикатори ризику та текстові рекомендації.

До основних професійних ризиків під час роботи з комп’ютеризованою медичною системою належать:

– зорове перенапруження, виникає внаслідок тривалого перегляду дрібного тексту, таблиць, графіків та результатів прогнозування. Для профілактики необхідно налаштовувати яскравість і контрастність монітора, уникати відблисків та періодично переводити погляд на віддалені об’єкти;

– синдром сухого ока, може виникати через зменшення частоти моргання під час зосередженої роботи з екраном. Зменшенню ризику сприяють короткі перерви, підтримання вологості повітря на рівні 40-60 **%** та правильне розташування монітора;

– біль у шиї, плечах і спині, пов’язаний зі статичною позою та неправильним положенням тіла. Профілактичними заходами є використання ергономічного крісла, правильна висота робочої поверхні та періодична зміна положення тіла;

– втома кистей і передпліч, може виникати через постійне введення даних, роботу з мишею та заповнення форм. Для зменшення навантаження слід забезпечити правильне положення рук, використовувати зручну клавіатуру та виконувати короткі вправи для кистей;

– психоемоційне напруження, пов’язане з роботою з медичними ризиками, відповідальністю за правильність введення даних і необхідністю швидкої інтерпретації результатів. Зниженню такого навантаження сприяють логічно структурований інтерфейс, автоматичні підказки, валідація введених значень і зрозумілі повідомлення системи;

– помилки через втому, можуть виникати при тривалій концентрації на великій кількості медичних записів. Профілактика передбачає використання автоматичної перевірки даних, системних попереджень та чіткого розмежування критичних повідомлень.

Профілактика порушень опорно-рухового апарату передбачає правильну організацію робочої пози. Спина повинна мати опору, плечі мають залишатися розслабленими, а стопи повинні стояти на підлозі або спеціальній підставці. Робоча поверхня має бути достатньою для розміщення клавіатури, миші та

необхідних документів. Небажаним є тривале перебування в нахиленому положенні або постійне напруження м’язів шиї.

До основних профілактичних заходів для зниження статичного навантаження належать:

– підтримання прямого положення спини під час роботи;

– використання крісла з підтримкою попереку;

– розташування клавіатури та миші на рівні, зручному для природного положення рук;

– виконання коротких розминок для шиї, плечей, кистей і спини;

– уникнення тривалої роботи в одній позі.

Особливе значення має також програмна організація інтерфейсу. Розроблена система зменшує навантаження на користувача за рахунок структурованих форм введення, автоматичної перевірки показників, кольорової індикації рівня ризику та зрозумілого формування рекомендацій.

Завдяки такій організації частина контролю переноситься на програмний рівень, що зменшує ймовірність помилок, спричинених втомою або перевантаженням користувача.

4.3 Безпека в надзвичайних ситуаціях під час експлуатації інтелектуальної медичної системи

Під час експлуатації інтелектуальної медичної системи необхідно враховувати ризики, що можуть виникати не лише у штатному режимі роботи, а й за умов надзвичайних ситуацій. Відповідно до Кодексу цивільного захисту України, питання безпеки у надзвичайних ситуаціях охоплюють захист населення, майна, територій та організацію реагування на небезпечні події [49].

Для розробленого програмного комплексу такі ситуації можуть бути пов’язані як із фізичною небезпекою для працівників, так і з порушенням доступу до електронної медичної документації, відмовою серверного обладнання або пошкодженням інформаційних ресурсів.

Нормативною основою для розгляду цього питання є ДСТУ 3891:2013

“Безпека у надзвичайних ситуаціях. Терміни та визначення основних понять”, який установлює термінологічну базу у сфері безпеки під час надзвичайних ситуацій [58].

Також у межах підрозділу враховано положення ДСТУ 2272:2006 “Пожежна безпека. Терміни та визначення основних понять” і Правил пожежної безпеки в Україні, затверджених наказом МВС України від 30.12.2014 № 1417, які визначають загальні вимоги щодо пожежної безпеки об’єктів, обладнання та приміщень [59].

У випадку виникнення пожежі, задимлення, перегріву електрообладнання або загрози ураження електричним струмом першочерговим завданням є захист життя та здоров’я працівників.

Медичний персонал повинен припинити роботу з комп’ютерною технікою, за можливості відключити живлення обладнання, повідомити відповідальних осіб і діяти згідно з планом евакуації. Робочі місця, серверне обладнання та мережеві пристрої мають розміщуватися таким чином, щоб не перешкоджати вільному пересуванню працівників і не блокувати евакуаційні виходи.

До основних надзвичайних ситуацій, які можуть вплинути на роботу інтелектуальної медичної системи, належать:

– Відключення електроенергії, може призвести до втрати незбережених даних, переривання запиту до сервера або тимчасової зупинки роботи системи. Для зменшення ризику необхідно використовувати джерела безперебійного живлення, механізми автозбереження та резервне живлення для критичних елементів інфраструктури;

– Пожежа або задимлення у приміщенні, може створити безпосередню загрозу для працівників і призвести до пошкодження комп’ютерної техніки, серверного обладнання та носіїв інформації. У такій ситуації необхідно припинити роботу, відключити електроживлення за можливості, повідомити відповідальних осіб і виконати евакуацію згідно з установленим порядком;

– Збій серверної частини або ML-модуля, може унеможливити формування прогнозу або призвести до помилки під час обробки даних. У такому випадку

система не повинна формувати неповний результат, користувач має отримати зрозуміле повідомлення, а технічна інформація про помилку повинна записуватися в журнал;

– Кіберінцидент, пов’язаний із несанкціонованим доступом, зміною або витоком персональних медичних даних. Для захисту інформації необхідно застосовувати HTTPS, JWT-автентифікацію, хешування паролів, розмежування ролей доступу та журналювання дій користувачів;

– Перевантаження сервера, може спричинити затримку відповідей або тимчасову недоступність системи. Для зменшення такого ризику доцільно використовувати моніторинг навантаження, кешування результатів і масштабування серверних компонентів.

Особливу увагу слід приділяти збереженню електронної медичної документації. У разі аварійного завершення роботи системи не повинні втрачатися записи пацієнтів, результати обстежень або сформовані прогнози.

Для цього необхідно використовувати транзакційне збереження даних, регулярне резервне копіювання бази, контроль цілісності записів і журнали подій. Така організація дозволяє відновити роботу системи після технічного збою без втрати критично важливої інформації [60].

Для забезпечення інформаційної стійкості системи доцільно передбачити такі заходи:

– регулярне створення резервних копій бази даних;

– перевірку можливості відновлення даних із резервних копій;

– використання транзакційного збереження результатів прогнозування;

– контроль цілісності записів після аварійного завершення роботи;

Після усунення наслідків надзвичайної ситуації необхідно виконати послідовну перевірку працездатності системи. Насамперед перевіряється доступність бази даних, коректність авторизації користувачів, робота серверної частини та можливість взаємодії з Python-аналітичним ядром.

4.4 Висновок до четвертого розділу

У четвертому розділі було розглянуто питання безпеки в надзвичайних ситуаціях та основи охорони праці під час експлуатації інтелектуальної медичної інформаційної системи. Визначено, що робота з такою системою пов’язана не лише з використанням комп’ютерної техніки, а й з підвищеним інформаційним навантаженням, оскільки користувач опрацьовує електронні медичні записи, клінічні показники, результати прогнозування та рекомендації щодо ризиків.

У межах розділу проаналізовано нормативні вимоги до організації робочого місця користувача системи з урахуванням положень ДСТУ, НПАОП, ДСН та ДСанПіН. Описано основні параметри безпечного робочого середовища, зокрема рекомендовану відстань від очей до екрана, рівень освітленості, температурний режим, вологість повітря, допустимий рівень шуму та режим праці й відпочинку. Дотримання цих вимог дозволяє зменшити зорове, статичне та психоемоційне навантаження на медичного працівника.

Окрему увагу приділено профілактиці професійних захворювань, які можуть виникати під час тривалої роботи з екранними пристроями. Розглянуто ризики зорового перенапруження, синдрому сухого ока, болю у шиї, плечах і спині, втоми кистей та психоемоційного виснаження. Обґрунтовано необхідність регламентованих перерв, правильної організації робочої пози, налаштування параметрів екрана та використання ергономічного робочого місця. Також у розділі визначено основні загрози, що можуть виникати під час надзвичайних ситуацій при експлуатації медичної інформаційної системи. До таких загроз належать відключення електроенергії, пожежа, збій бази даних, порушення роботи серверної частини, відмова ML-модуля та кіберінциденти. Для зменшення їх наслідків запропоновано заходи резервного копіювання, використання джерел безперебійного живлення, журналювання подій, захищеної автентифікації та контрольованого відновлення працездатності системи.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи освітнього рівня «Магістр» було досліджено підходи до інтелектуального аналізу динаміки захворювань на основі даних електронної медичної документації. У роботі розглянуто проблему раннього виявлення серцево-судинних ризиків, яка є актуальною в умовах цифровізації медичної галузі та зростання обсягів клінічних даних. Встановлено, що традиційні методи оцінювання ризиків не завжди забезпечують достатню гнучкість під час роботи з різнорідними, неповними та динамічними медичними показниками. Саме тому доцільним є використання методів машинного навчання, здатних виявляти складні залежності між параметрами стану пацієнта.

У першому розділі кваліфікаційної роботи освітнього рівня «Магістр» було розкрито теоретичні основи застосування інтелектуальних систем у медицині. Проаналізовано концепцію P4-медицини, роль електронної медичної документації у формуванні цифрового профілю пацієнта та значення предиктивної аналітики для раннього виявлення патологічних змін. Також було розглянуто особливості медичних даних, зокрема їхню неоднорідність, наявність пропущених значень, залежність показників у часі та потребу в попередній обробці перед використанням у моделях машинного навчання. На основі огляду існуючих рішень визначено необхідність створення веб-орієнтованої системи, яка поєднує зручний інтерфейс із потужним аналітичним ядром.

У другому розділі було обґрунтовано вибір математичного апарату та технологічної архітектури системи. Основним алгоритмом прогнозування обрано XGBoost, оскільки він ефективно працює зі структурованими медичними даними, підтримує обробку пропущених значень і дозволяє досягати високої точності класифікації. Описано процес формування вектора ознак на основі антропометричних, фізіологічних та лабораторних параметрів пацієнта. Крім того, розглянуто підходи до валідації даних, нормалізації показників та побудови трирівневої архітектури, що включає клієнтську частину, серверну логіку та окремий аналітичний модуль на Python.

У третьому розділі було реалізовано програмний прототип інтелектуальної системи. Клієнтська частина створена з використанням React і забезпечує роботу з реєстром пацієнтів, введення клінічних даних, запуск прогнозування та візуалізацію результатів. Серверна частина на Node.js відповідає за обробку запитів, взаємодію з базою даних і передачу даних до Python-модуля, у якому виконується робота моделі XGBoost. У межах апробації проведено тестування системи на контрольних наборах даних, що підтвердило її здатність швидко формувати прогноз ризику та надавати лікарю структуровані рекомендації. Отримана точність прогнозування перевищила 90%, що свідчить про ефективність обраного підходу.

Практична цінність дослідження полягає у створенні програмного рішення, яке може бути використане в медичних установах для автоматизації первинного аналізу клінічних показників і підтримки прийняття лікарських рішень. Запропонована система дозволяє скоротити час обробки інформації про пацієнта, зменшити ймовірність помилок під час інтерпретації даних та підвищити оперативність реагування на потенційно небезпечні зміни стану здоров’я. Завдяки веб-орієнтованій архітектурі система може бути масштабована та адаптована до потреб різних медичних закладів.

У результаті виконання роботи було підтверджено доцільність використання алгоритмів машинного навчання, зокрема XGBoost, для задач прогнозування серцево-судинних ризиків на основі електронної медичної документації. Розроблена система поєднує інструменти предиктивної аналітики, механізми валідації даних і зручний веб-інтерфейс, що створює основу для подальшого розвитку інтелектуальних медичних сервісів, орієнтованих на ранню діагностику, персоналізований моніторинг та підвищення якості медичної допомоги.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Topol E. J. High-performance medicine: the convergence of human and artificial intelligence. Nature Medicine. 07.01.2019. URL: https://www.nature.com/articles/s41591-018-0300-7.pdf.

2. WHO. Global strategy on digital health 2020-2025. World Health Organization. 2021. URL: https://www.who.int/docs/defaultsource/documents/ gs4dhdaa2a97010b744a39a848a601345564f.pdf.

3. Haleem A., Javaid M., Singh R. P., Suman R. Telemedicine for healthcare: Capabilities, features, barriers, and applications. Sensors International. 2021. URL: https://doi.org/10.1016/j.sintl.2021.100117.

4. Dash S., Shakyawar S. K., Sharma M., Kaushik S. Big data in healthcare: management, analysis and future prospects. Journal of Big Data. 19.06.2019. URL: https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0217-0.

5. Meskó B., deBronkart D. Patient Design: The Next Step in User-Centered Healthcare. Journal of Medical Internet Research. 2022. URL: https://www.jmir.org/2022/9/e39434/.

6. Sutton R. T., Pincock D., Baumgart D. C. An overview of clinical decision support systems: benefits, risks, and strategies for success. NPJ Digital Medicine. 06.02.2020. URL: https://www.nature.com/articles/s41746-020-0221-y.pdf.

7. Zikos D., DeLellis N. CDSS Evaluation Framework: A Systematic Review. Health Policy and Technology. 2018. URL: https://doi.org/10.1016/j.hlpt.2018.10.003.

8. Rajpurkar P., Chen E., Banerjee O., Topol E. J. AI in health and medicine. Nature Medicine. 20.01.2022. URL: https://www.nature.com/articles/s41591-021-01614-0.pdf.

9. Wasylewicz A. T., Scheepers-Hoeks A. Clinical Decision Support Systems. Fundamentals of Clinical Data Science. 2019. URL: https://link.springer.com/chapter/10.1007/978-3-319-99713-1_13.

10. Miller R. A. Clinical Decision Support Systems: A 25-Year Retrospective and a 25-Year Prospective. Yearbook of Medical Informatics. 2016. URL: https://pubmed.ncbi.nlm.nih.gov/27494165/.

11. Jiang F., Jiang Y., Zhi H. Artificial intelligence in healthcare: past, present and future. Stroke and Vascular Neurology. 2017. URL: https://svn.bmj.com/content/svneurol/2/4/230.full.pdf.

12. Kaur P., Sharma M., Mittal M. Big Data and Machine Learning Algorithms for Health-Care Analysis. In: Scalable Computing: Practice and Experience. 2018. URL: https://www.scpe.org/index.php/scpe/article/view/1344.

13. Wiens J., Saria S., Sendak M. Do no harm: a roadmap for responsible machine learning for health. Nature Medicine. 2019. URL: https://www.nature.com/articles/s41591-019-0548-6.

14. Shailaja K., Seetharamulu B., Jabbar M. A. Machine Learning in Healthcare: A Review. International Conference on Electronics, Communication and Aerospace Technology. 2018. URL: https://ieeexplore.ieee.org/document/8474914.

15. Kourou K., Exarchos T. P., Fotiadis D. I. Machine learning applications in cancer prognosis and prediction. Computational and Structural Biotechnology Journal. 2015. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4348437/.

16. Price W. N., Cohen I. G. Privacy in the age of medical big data. Nature Medicine. 2019. URL: https://www.nature.com/articles/s41591-018-0272-7.

17. Mittelstadt B. Principles of Biomedical Ethics in the Digital Age. Ethics and Information Technology. 2019. URL: https://link.springer.com/article /10.1007/s10676-019-09513-z.

18. Vayena E., Blasimme A., Tasioulas J. Machine learning in medicine: Addressing ethical challenges. PLOS Medicine. 2018. URL: https://journals.plos.org/plosmedicine/article?id=10.1371/journal.pmed.1002689.

19. Ghassemi M., Naumann T., Schulam P. A review of challenges and opportunities in machine learning for health. AMIA Summits on Translational Science Proceedings. 2020. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7233045/.

20. European Commission. Ethics guidelines for trustworthy AI. B-1049 Brussels. 08.04.2019. URL: https://ec.europa.eu/newsroom/dae/document.cfm?doc_id=60419.

21. Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System. KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge

Discovery and Data Mining. 13.08.2016. URL: https://arxiv.org/pdf/1603.02754.pdf.

22. Nielsen D. Tree Boosting With XGBoost - Why Does XGBoost Win Every Machine Learning Competition? NTNU Open. 2016. URL: https://ntnuopen.ntnu.no/ntnu-xmlui/bitstream/handle/11250/2433761/16128.pdf.

23. Lundberg S. M., Lee S.-I. A Unified Approach to Interpreting Model Predictions. Advances in Neural Information Processing Systems 30. 2017. URL: https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.

24. Bentéjac C., Csörgő A., Martínez-Muñoz G. A comparative analysis of gradient boosting algorithms. Springer Link. 2020. URL: https://link.springer.com/article/10.1007/s10462-020-09839-3.

25. Brownlee J. XGBoost With Python: Gradient Boosted Trees with XGBoost and Scikit-Learn. Machine Learning Mastery. 2021. URL: https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/.

26. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Stanford University. 2017. URL: https://web.stanford.edu/~hastie/ElemStatLearn/printings/ESLII_print12_toc.pdf.

27. Banks A., Porcello E. Learning React: Modern Patterns for Developing React Apps. O'Reilly Media. 2020. URL: https://www.oreilly.com/library/view/learning-react-2nd/9781492044819/.

28. Teixeira H. Node.js Design Patterns: Design and implement production-grade Node.js applications. URL: https://www.packtpub.com/product/node-js-design-patterns-third-edition/9781839214110.

29. Voron A. FastAPI - Modern Python Web Development. Leanpub. 2021. URL: https://fastapi.tiangolo.com/learn/.

30. Kleppmann M. Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems. O'Reilly Media. 2017. URL: https://dataintensive.net/.

31. Sadalage P. J., Fowler M. NoSQL Distilled: A Brief Guide to the Emerging

World of Polyglot Persistence. Pearson Education. 2013. URL: https://martinfowler.com/books/nosql.html.

32. Silberschatz A., Korth H., Sudarshan S. Database System Concepts. McGraw-Hill Education. 2019. URL: https://db-book.com/.

33. Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2016. URL: https://arxiv.org/abs/1603.02754.

34. Lundberg S. M., Lee S.-I. A Unified Approach to Interpreting Model Predictions. Advances in Neural Information Processing Systems (NeurIPS). 2017. URL: https://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf.

35. Grinberg M. Flask Web Development: Developing Web Applications with Python. 2nd Edition. O'Reilly Media. 2018. URL: https://www.oreilly.com/library/view/flask-web-development/9781491991725/.

36. Flanagan D. JavaScript: The Definitive Guide. 7th Edition. O'Reilly Media. 2020. URL: https://www.oreilly.com/library/view/javascript-the-definitive/9781491952016/.

37. Teixeira H., Casciaro M. Node.js Design Patterns: Design and implement production-grade Node.js applications. 3rd Edition. Packt Publishing. 2020. URL: https://www.packtpub.com/product/nodejsdesignpatternsthirdedition/9781839214110

38. Node.js Integration with Python Scripts via Child Processes. Node.js Official Documentation. 2025. URL: https://nodejs.org/api/child_process.html.

39. Глибовець М. М., Олєйніков О. М. Розробка веб-застосувань на базі Node.js та Express. Комп’ютерне моделювання та інформаційні технології. 2021. С. 45–52.

40. Шпортько О. В., Шпортько Г. О. Проектування та розробка баз даних у реляційній СКБД MySQL. Вісник Національного університету водного господарства та природокористування. 2022. С. 112–121.

41. Shemer M., Gudes E. Security and Privacy in Medical Information Systems. International Journal of Environmental Research and Public Health. 2022. URL:

https://www.mdpi.com/1660-4601/19/13/7714.

42. Banks A., Porcello E. Learning React: Modern Patterns for Developing React Apps. 2nd Edition. O'Reilly Media. 2020. URL: https://www.oreilly.com/library/view/learning-react-2nd/9781492044819/.

43. React Documentation. Official website. URL: https://react.dev/.

44. Osmani A. Learning JavaScript Design Patterns. 2nd Edition. O'Reilly Media. 2023. URL: https://www.oreilly.com/library/view/learning-javascript-design/9781098139865/.

45. Березська К. М., Гевко Г. В. Проектування користувацьких інтерфейсів медичних інформаційних систем. Інноваційні технології та комп'ютерна інженерія. 2023. № 1 (37). С. 24–31.

46. Meszaros G. xUnit Test Patterns: Refactoring Test Code. Addison-Wesley. 2007. URL: http://xunitpatterns.com/.

47. Fowler M. Refactoring: Improving the Design of Existing Code. 2nd Edition. Addison-Wesley Professional. 2018. URL: https://martinfowler.com/books/refactoring.html.

48. A. Zaporozhets, Y. Kuts, B. Mlynko, M. Fryz, and L. Scherbak, "EEG Signal Classification Using Linear Process Model-Based Feature Extraction and Supervised Learning," in Advanced System Development Technologies II. Studies in Systems, Decision and Control, M. Bezuglyi, N. Bouraou, V. Mykytenko, G. Tymchyk, and A. Zaporozhets, Eds., Cham: Springer Nature Switzerland, 2025, pp. 235–257. doi: 10.1007/978-3-031-82035-9_7.

49. M. Fryz, L. Scherbak, B. Mlynko, and T. Mykhailovych, "Linear RandomProcess Model-Based EEG Classification Using Machine LearningTechniques," in Proceedings of the 1st International Workshop on Computer Information Technologies in Industry 4.0 (CITI 2023), 2023, vol. 3468, pp. 126–132. [Online]. Available: https://ceur-ws.org/Vol-3468/short5.pdf

50. Бабак В.П., Куц Ю.В., Мислович М.В., Фриз М.Є., Щербак Л.М. Об'єктно-орієнтована ідентифікація стохастичних шумових сигналів. Київ: Наукова думка, 2024. 240 с. https://doi.org/10.15407/978-966-00-1883-9.

51. V. Babak, A. Zaporozhets, Y. Kuts, M. Fryz, L. Scherbak. Noise signals: Modelling and Analyses. Cham: Springer Nature Switzerland, 2025. 222 p. DOI: https://doi.org/10.1007/978-3-031-71093-3

52. Бабак В.П., Марченко Б.Г., Фриз М.Є. Теорія ймовірностей, випадкові процеси та математична статистика. – К.: Техніка, 2004. – 288 с.

53. M. Fryz, “Conditional linear random process and random coefficient autoregressive model for EEG analysis,” 2017. doi:10.1109/UKRCON.2017.8100498.

54. Fryz M., Mlynko B. Property analysis of multivariate conditional linear random processes in the problems of mathematical modelling of signals // Technol. Audit Prod. Reserv. 2022. Vol. 3, No 2(65). P. 29–32.

55. ДСТУ 2293:2014. Охорона праці. Терміни та визначення основних понять. Чинний від 2015-05-01. Київ: Мінекономрозвитку України, 2014.

56. Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями: наказ Міністерства соціальної політики України від 14.02.2018 № 207.

57. ДСН 3.3.6.042-99. Санітарні норми мікроклімату виробничих приміщень.

58. Кодекс цивільного захисту України: Закон України № 5403-VI.

59. ДСТУ 3891:2013. Безпека у надзвичайних ситуаціях. Терміни та визначення основних понять.

60. Про затвердження правил пожежної безпеки в Україні: наказ Міністерства внутрішніх справ України від 30.12.2014 № 1417.

ДОДАТКИ

Додаток А

Публікації

IX Міжнародна студентська науково - технічна конференція
"ПРИРОДНИЧІ ТА ГУМАНІТАРНІ НАУКИ. АКТУАЛЬНІ ПИТАННЯ"

УДК 004.8
Шабля Р. - ст. гр. СНнм-61
Тернопільський національний технічний університет імені Івана Пулюя

ПРОБЛЕМА НЕОДНОРІДНОСТІ ЕЛЕКТРОННОЇ МЕДИЧНОЇ ДОКУМЕНТАЦІЇ У ЗАДАЧАХ АНАЛІЗУ ЗАХВОРЮВАНЬ

Shablia R.
Ternopil Ivan Puluj National Technical University

THE PROBLEM OF HETEROGENEITY OF ELECTRONIC MEDICAL RECORDS IN DISEASE ANALYSIS TASK

У сучасних системах охорони здоров'я електронна медична документація є основним джерелом даних для побудови моделей прогнозування захворювань та підтримки клінічних рішень. Активне впровадження інформаційних систем у медичну практику призвело до накопичення значних обсягів даних про пацієнтів, що відкриває нові можливості для застосування методів машинного навчання. Проте ефективність таких моделей суттєво обмежується якістю, структурою та узгодженістю вхідних даних.

Ключовою проблемою є неоднорідність медичних даних. Інформація про пацієнта формується з різних джерел, включаючи лабораторні дослідження, результати інструментальних обстежень, текстові висновки лікарів, історії лікування та дані моніторингу стану пацієнта. Ці дані відрізняються за форматом (структуровані, напівструктуровані, неструктуровані), частотою оновлення, точністю вимірювань та рівнем деталізації. У результаті виникають труднощі при їх інтеграції в єдину аналітичну систему, що негативно впливає на якість побудованих моделей. Додатковою складністю є те, що електронна медична документація часто містить пропуски, дублювання записів, суперечливу інформацію та шум. Причинами цього можуть бути людський фактор, різні стандарти ведення документації, технічні обмеження медичних інформаційних систем, а також нерегулярність спостережень за пацієнтом. У таких умовах традиційні підходи до аналізу даних, які передбачають їх повноту та узгодженість, втрачають ефективність.

Особливо критичною є проблема обробки неструктурованих текстових медичних записів, які містять значну частину клінічно важливої інформації, зокрема опис симптомів, висновки лікарів та рекомендації щодо лікування. Такі дані не можуть бути безпосередньо використані в моделях машинного навчання без попередньої обробки. Відсутність або обмежене використання методів обробки природної мови призводить до втрати значної частини інформації, що знижує якість аналізу та точність прогнозування.

Таким чином, підвищення ефективності прогнозних моделей потребує розробки комплексних підходів до попередньої обробки та уніфікації медичних даних. До таких підходів належать методи очищення даних, заповнення пропусків, нормалізації показників, інтеграції даних із різних джерел, а також використання алгоритмів для роботи з неструктурованою інформацією. Важливим напрямом є також застосування моделей, здатних працювати з неоднорідними та часовими даними, включаючи гібридні підходи, що поєднують різні типи інформації. Неоднорідність електронної медичної документації є одним із ключових факторів, що обмежує точність та надійність моделей прогнозування захворювань. Подолання цієї проблеми потребує комплексного підходу до обробки, інтеграції та аналізу даних, а також використання сучасних методів машинного навчання, здатних ефективно працювати з різнорідною медичною інформацією.

УДК 004.8
Шабля Р. - ст. гр. СНнм-61
Тернопільський національний технічний університет імені Івана Пулюя

ПОРІВНЯННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ ПРОГНОЗУВАННЯ СЕРЦЕВО-СУДИННИХ ПОКАЗНИКІВ НА ОСНОВІ ЧАСОВИХ РЯДІВ

Shablia R.
Ternopil Ivan Puluj National Technical University

COMPARISON OF MACHINE LEARNING METHODS FOR PREDICTING CARDIOVASCULAR INDICATORS BASED ON TIME SERIES

Серцево-судинні захворювання залишаються однією з провідних причин смертності у світі. Сучасні моніторингові системи та електронні медичні записи (EMR) дозволяють збирати багатовимірні часові ряди таких показників, як артеріальний тиск, частота серцевих скорочень, рівень холестерину та глюкози. Аналіз цих даних із застосуванням методів машинного навчання відкриває можливості для раннього виявлення ризиків та прогнозування погіршення стану пацієнта.

Основною проблемою є високий рівень варіативності та шумності даних: пропуски вимірювань, нерегулярні інтервали спостережень та вплив зовнішніх факторів. Крім того, різні показники мають різні шкали та частоти оновлення, що ускладнює інтеграцію даних для багатовимірного прогнозування. Традиційні статистичні методи недостатньо ефективні для виявлення складних залежностей між показниками у часі.

Для прогнозування динаміки серцево-судинних показників використовуються класичні статистичні методи, такі як ARIMA та експоненційне згладжування. Вони ефективні для аналізу одновимірних часових рядів і дозволяють моделювати тренди та сезонні коливання показників. Рекурентні нейронні мережі (RNN, LSTM, GRU) застосовуються для моделювання довгострокових залежностей у багатовимірних часових рядах. Вони здатні враховувати попередні стани системи, що є критично важливим для прогнозування динаміки стану пацієнта.

Трансформери ефективні для роботи з багатовимірними часовими рядами з нерегулярними інтервалами вимірювань. Вони дозволяють моделювати складні взаємозв'язки між показниками та одночасно обробляти великі обсяги даних.

Попередня обробка даних включає нормалізацію показників, інтерполяцію пропусків, виявлення аномалій та синхронізацію різних параметрів. Ці кроки забезпечують якісну основу для навчання моделей та підвищують точність прогнозування.

Методики машинного навчання, особливо LSTM та трансформери, демонструють високий потенціал для прогнозування серцево-судинних показників на основі часових рядів. Вони дозволяють враховувати динаміку стану пацієнта, передбачати можливі загострення та підтримують розвиток персоналізованої медицини. Попередня обробка даних та коректне формування багатовимірних рядів є критично важливими для підвищення точності моделей.