

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
Факультет комп'ютерно-інтегрованих технологій, мехатроніки і
робототехніки
Кафедра робототехніки, електроенергетики та автоматизації
ім. проф. Б.Б. Самотокіна

ПОЯСНЮВАЛЬНА ЗАПИСКА
до кваліфікаційної роботи магістра
на тему: **«Автоматизована система прогнозування виявлення цукрового**
діабету на основі медичних даних пацієнтів»

Виконав студент 2 курсу, групи АТ-28м
спеціальності 174 “ Автоматизація,
комп'ютерно-інтегровані технології
та робототехніка”

Підопригора Сергій Вікторович

(прізвище та ініціали студента)

Керівник: к.т.н., доц. Ткачук А.Г.

(науковий ступінь, вчене звання, посада,
прізвище та ініціали)

Рецензент: д.т.н., проф. Подчашинський Ю.О.

(науковий ступінь, вчене звання, посада,
прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ “ЖИТОМИРСЬКА ПОЛІТЕХНІКА”
Факультет комп’ютерно-інтегрованих технологій, мехатроніки і
робототехніки

Кафедра робототехніки, електроенергетики та автоматизації
ім. проф. Б.Б. Самотокіна

Спеціальність 174 “Автоматизація, комп’ютерно-інтегровані технології
та робототехніка”

ЗАТВЕРДЖУЮ
Завідувач кафедри
робототехніки, електроенергетики
та автоматизації
ім. проф. Б.Б. Самотокіна
к.т.н., доц. Громовий О.А.

к.т.н., доц. Громовий О.А.
«_____» _____ 2024р.

ЗАВДАННЯ

на кваліфікаційну роботу магістра

- Студент Підпригора Сергій Вікторович
1. Тема роботи: Автоматизована система прогнозування виявлення цукрового діабету на основі медичних даних пацієнтів
Затверджена Наказом по університету від «12» листопада 2024 р. № 646/с
 2. Термін здачі закінченої роботи на кафедрі «20» грудня 2024 р.
 3. Вихідні дані до роботи матеріали звіту про переддипломну практику, технічне завдання
 4. Зміст пояснювальної записки (перелік питань, які підлягають розробці) технічне завдання, огляд літератури, методологія та матеріали, розробка моделей, розробка програмного забезпечення, економне обґрунтування
 5. Перелік презентативного матеріалу (з точним визначенням обов’язкових слайдів/креслень) блок-схема алгоритму
 6. Консультанти з кваліфікаційної роботи магістра із зазначенням розділів роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Технічна частина	Ткачук А.Г., к.т.н, доцент	11.09.2024	11.09.2024

7. Дата видачі завдання 11.09.2024

Календарний план

№ з/п	Назва етапів виконання кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1.	Підготувати звіт про переддипломну практику	1.11.2024 – 1.12.2024	виконано
2.	Збір літературних даних за темою кваліфікаційної роботи та відомих систем-аналогів	21.09.2024 – 24.09.2024	виконано
3.	Ознайомитись з основними принципами побудови систем	24.09.2024 – 11.10.2024	виконано
4.	Вибір архітектури системи	11.10.2024 – 02.11.2024	виконано
5.	Розробка та навчання моделей	02.11.2024 – 10.11.2024	виконано
6.	Розробка програмного забезпечення	10.11.2024 – 30.11.2024	виконано
7.	Економічне обґрунтування	10.11.2024 – 30.11.2024	виконано
8.	Оформлення пояснювальної записки та презентаційного матеріалу	30.11.2024 – 11.12.2024	виконано

Студент _____
(підпис, дата)

С.В. Підпригора
(прізвище та ініціали)

Керівник роботи _____
(підпис, дата)

А.Г. Ткачук
(прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота присвячена розробці автоматизованої системи прогнозування цукрового діабету на основі аналізу медичних даних пацієнтів із використанням алгоритмів машинного навчання. З метою підвищення доступності ранньої діагностики створено веборієнтовану систему з інтерактивним інтерфейсом, зручним для лікарів і пацієнтів. У дослідженні проведено порівняння алгоритмів, таких як логістична регресія, метод опорних векторів і градієнтний бустинг, із врахуванням якості обробки даних.

Результатом є система, здатна аналізувати великі обсяги медичної інформації, забезпечуючи високоточні прогнози. Вона інтегрується з існуючими медичними базами та сприяє зниженню витрат на лікування і покращенню якості медичних послуг.

ANNOTATION

The qualification work is dedicated to developing an automated system for predicting diabetes based on the analysis of patients' medical data using machine learning algorithms. To enhance the accessibility of early diagnosis, a web-based system with an interactive interface, user-friendly for both doctors and patients, has been created. The study compares algorithms such as logistic regression, support vector machines, and gradient boosting, focusing on data preprocessing quality.

The result is a system capable of analyzing large volumes of medical information, providing highly accurate predictions. It integrates with existing medical databases and contributes to reducing treatment costs while improving the quality of healthcare services.

ЗМІСТ

ВСТУП	1
1 ОГЛЯД ЛІТЕРАТУРИ.....	8
1.1 Аналіз сучасного стану дослідження цукрового діабету та методів його виявлення	8
1.2 Огляд існуючих автоматизованих систем прогнозування	9
1.3 Сучасні підходи машинного навчання у медицині.....	13
1.4 Вибір і обґрунтування напрямків дослідження	15
2. МЕТОДОЛОГІЯ ТА МАТЕРІАЛИ	20
2.1 Опис використовуваних медичних даних (джерела, обсяги, формат)	20
2.2 Підготовка даних.....	23
2.3 Обґрунтування вибору моделей машинного навчання	25
2.4 Архітектура розробленої системи	28
2.5 Використовувані технології та програмні засоби	30
3 РОЗРОБКА МОДЕЛЕЙ.....	33
3.1 Створення моделі машинного навчання на основі логістичної регресії	33
3.2 Створення моделі машинного навчання на основі методу опорних векторів (SVM)	38
3.3 Створення моделі машинного навчання на основі методу градієнтного бустингу (XGBoost).....	44
3.4 Створення моделі машинного навчання на основі методу k найближчих сусідів (K-NN)	48
Порівняльний аналіз моделей	54
4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	58
4.1 Архітектура API на Flask для взаємодії з моделями машинного навчання	58
4.2 Розробка користувацького інтерфейсу на Vue.js	65
5 ІНТЕГРАЦІЯ АПАРАТНИХ МОДУЛІВ ЧЕРЕЗ REST API ДЛЯ ВІДДАЛЕНОГО МЕДИЧНОГО АНАЛІЗУ	89

5.1 Опис протоколу HTTP та RESTful API	89
5.2 Опис модуля ESP8266 (Arduino), як основи для систем віддаленого медичного аналізу.....	91
5.3 Приклад інтеграції модуля ESP8266 (Arduino) з системою прогнозування виявлення діабету	94
6 ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ	98
6.1 Можливі схеми монетизації технології машинного навчання для діагностики цукрового діабету.....	98
6.2 Розрахунок собівартості створення та просування сервісу для прогнозування виявлення цукрового діабету	99
6.3 Мінімальної кількості клієнтів, необхідної для виходу на точку беззбитковості	102
6.4 Висновок щодо перспектив розвитку автоматизованої система прогнозування виявлення цукрового діабету	103
ВИСНОВОК	105
Узагальнення отриманих результатів	105
Внесок у вирішення проблеми автоматизованого виявлення діабету	106
Практичне значення роботи.....	106
РЕКОМЕНДАЦІЇ	108
Подальші напрямки розвитку системи.....	108
Можливості інтеграції з існуючими медичними інформаційними системами.....	109
СПИСОК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ	112
ДОДАТОК А: Функціональна схема автоматизованої системи прогнозування виявлення цукрового діабету	114
ДОДАТОК Б: Блок-схема для оцінки роботи алгоритму прогнозування виявлення цукрового діабету на основі медичних даних пацієнтів	115
ДОДАТОК В: Блок-схема алгоритму вибору найточнішого передбачення	116

ВСТУП

Актуальність дослідження

Цукровий діабет є однією з найбільш поширених хронічних хвороб у світі. Це захворювання є одним із найдавніших відомих людству, і його опис можна знайти ще у давньоєгипетському папірусі Еберса, датованому 1550 роком до нашої ери. Протягом тисячоліть цукровий діабет залишався невиліковним, аж до відкриття інсуліну у 1921 році [1].

Відомості про цукровий діабет можна знайти в працях лікарів Стародавньої Греції, Індії, Персії та Китаю. Індійські лікарі називали це захворювання "медовою сечею" через солодкий смак сечі, що приваблював мурах. [1] [2] Відновлення досліджень діабету розпочалося в епоху Відродження завдяки працям швейцарського лікаря Парацельса, який помітив, що сеча хворих містить білу речовину, що залишалася після випаровування. У XVII столітті Томас Вілліс і Метью Добсон визначили, що цукор присутній не лише в сечі, а й у крові хворих на діабет [1] [3].

У 1855 році Клод Бернар показав, що печінка відіграє ключову роль у регуляції рівня цукру в крові, відкривши поняття глікогену. Це стало основою для подальшого розвитку ендокринології. В 1889 році Йозеф фон Мерінг і Оскар Мінковський експериментально викликали діабет у собак шляхом видалення підшлункової залози, що підтвердило її зв'язок із захворюванням [4] [2].

У 1921 році канадські вчені Фредерік Бантінг і Чарльз Герберт Бест вперше виділили інсулін з підшлункової залози і успішно застосували його для лікування діабету. Це відкриття змінило прогноз для пацієнтів з цукровим діабетом, які раніше мали лише рік життя після встановлення діагнозу. З тих пір розроблено численні методи введення інсуліну та моніторингу рівня глюкози в крові. У 1979 році був синтезований людський інсулін за допомогою генної інженерії, що покращило лікування пацієнтів [3] [2].

Ця історія свідчить про значний прогрес у розумінні та лікуванні цукрового діабету, який перетворився з невиліковної хвороби на контрольовану ендокринну патологію завдяки науковим відкриттям і технологічному прогресу.

За даними Всесвітньої організації охорони здоров'я (ВООЗ), кількість хворих на діабет у світі щороку зростає, і на сьогодні налічує понад 400 мільйонів людей. Очікується, що до 2045 року цей показник перевищить 700 мільйонів. В Україні проблема діабету також є надзвичайно актуальною, адже, за статистикою Міністерства охорони здоров'я України, понад 1 мільйон людей мають встановлений діагноз, причому значна частина населення не знає про наявність у себе цього захворювання [5].

Рання діагностика цукрового діабету має вирішальне значення для запобігання розвитку ускладнень, таких як серцево-судинні захворювання, ураження нирок, нервової системи та втрати зору [6]. Проте традиційні методи діагностики часто є трудомісткими, дорогими та вимагають значних ресурсів, що обмежує їх використання, особливо у віддалених регіонах або країнах з обмеженими можливостями охорони здоров'я.

З розвитком технологій штучного інтелекту (ШІ) та машинного навчання (ML) з'явилися нові можливості для вирішення проблеми ранньої діагностики діабету. Використання моделей машинного навчання дозволяє автоматизувати аналіз великих обсягів медичних даних, виявляти закономірності та робити прогнози з високою точністю [7] [8]. Такі підходи можуть значно підвищити доступність діагностики та забезпечити своєчасну медичну допомогу пацієнтам [9].

Наукова новизна даного дослідження полягає у створенні автоматизованої системи прогнозування, яка об'єднує сучасні алгоритми машинного навчання з веборієнтованою архітектурою. Це дозволяє не лише підвищити точність прогнозування, а й зробити цю технологію доступною для широкого кола користувачів через інтерактивний вебінтерфейс.

Таким чином, тема роботи є надзвичайно актуальною, оскільки сприяє вирішенню проблеми ранньої діагностики цукрового діабету за допомогою сучасних

інформаційних технологій. Розробка подібної системи не лише сприятиме зменшенню соціально-економічного тягаря, пов'язаного з діабетом, але й забезпечить інтеграцію інноваційних технологій у медичну практику.

Мета роботи

Метою даної роботи є створення автоматизованої системи прогнозування виявлення цукрового діабету на ранніх стадіях на основі аналізу великих обсягів медичних даних пацієнтів із використанням алгоритмів машинного навчання. Важливо забезпечити високу точність та ефективність прогнозування шляхом вибору оптимального алгоритму аналізу даних, що відповідає сучасним вимогам медичної практики.

Робота також передбачає інтеграцію розробленої системи у веборієнтоване середовище, що забезпечить доступність технології для широкого кола користувачів, включаючи медичних працівників та пацієнтів. Особливу увагу буде приділено зручності користування інтерфейсом системи, її масштабованості та можливостям інтеграції з існуючими медичними інформаційними системами.

Цільовою аудиторією проекту є медичні установи, лікарі, страхові компанії, пацієнти, державні органи охорони здоров'я, фармацевтичні компанії, фітнес-технологічні компанії та дослідницькі організації. Медичні заклади та лікарі можуть використовувати систему для підвищення точності діагностики, персоналізованого лікування та контролю динаміки захворювання. Страхові компанії отримають інструменти для оцінки ризиків клієнтів, аналізу захворюваності та розробки адаптованих страхових планів. Пацієнти зможуть активно моніторити стан свого здоров'я, отримувати своєчасні рекомендації та проводити профілактику діабету. Фармацевтичні компанії, урядові організації та дослідницькі установи зможуть використовувати агреговані дані для оптимізації медичних витрат, аналізу ефективності лікувальних заходів та планування клінічних досліджень.

Об'єкт дослідження

Об'єктом дослідження є процеси розвитку та прогнозування цукрового діабету, зокрема типу 2, який є одним із найпоширеніших захворювань обміну речовин у світі. Цукровий діабет 2 типу зазвичай розвивається у людей середнього або старшого віку, однак останнім часом все частіше реєструються випадки цього захворювання навіть серед дітей та підлітків. Така тенденція викликає серйозне занепокоєння серед медичних фахівців і дослідників.

Фактори, що сприяють розвитку цукрового діабету, включають генетичну схильність, надлишкову масу тіла, нездорові харчові звички, а також малорухливий спосіб життя. Вплив цих факторів на організм часто призводить до підвищення рівня цукру в крові (гіперглікемії), яка, у разі недіагностованого або неправильно контрольованого діабету, може спричинити серйозні ускладнення. До таких ускладнень належать ретинопатія, що може призвести до сліпоти, нефропатія, яка вражає нирки, нейропатія, що ускладнює роботу нервової системи, а також серцево-судинні захворювання, включаючи інфаркти та інсульти.

Раннє виявлення цукрового діабету є критично важливим для покращення якості життя пацієнтів, запобігання розвитку важких ускладнень та збільшення тривалості їх життя. Зокрема, застосування сучасних методів прогнозування та аналізу даних, таких як алгоритми машинного навчання, відкриває нові можливості для швидкої та точної діагностики цього захворювання. Дослідження об'єкту має на меті вивчення цих процесів та створення ефективних інструментів для їх контролю та аналізу.

Предмет дослідження

Предметом даного дослідження є використання сучасних методів машинного навчання (ML) та штучного інтелекту (AI) для діагностики, скринінгу та прогнозування цукрового діабету 2 типу [9]. Особливий акцент робиться на розробці та вдосконаленні алгоритмів, здатних аналізувати великі обсяги різнорідних медичних даних, виявляти закономірності та забезпечувати точне передбачення розвитку діабету на ранніх стадіях.

У межах дослідження вивчаються методи аналізу електронних медичних записів (EHR) з різних джерел, таких як національні бази даних, лікарні, клінічні центри та персональні пристрої для моніторингу здоров'я. Ці дані включають демографічну інформацію, історію хвороби, результати лабораторних аналізів, генетичні фактори та параметри способу життя пацієнтів. На основі цих даних створюються моделі для класифікації ризиків, прогнозування ускладнень та автоматизованого виявлення преддіабетичних станів [9].

Дослідження включає використання різних алгоритмів машинного навчання, таких як логістична регресія, дерева рішень, градієнтний бустинг, метод опорних векторів, глибокі нейронні мережі та ансамблеві методи. Особлива увага приділяється порівнянню ефективності цих алгоритмів у контексті точності, чутливості, специфічності та інших метрик оцінки моделей. Додатково досліджується вплив передобробки даних, вибору гіперпараметрів та балансування класів на загальну продуктивність системи [9] [7] [8].

Інтеграція створених моделей у реальні медичні системи передбачає їх адаптацію для використання через інтуїтивно зрозумілі вебінтерфейси та мобільні додатки, що дозволяє медикам та пацієнтам ефективно використовувати їх у клінічній практиці. Застосування цих технологій сприяє підвищенню доступності, точності та оперативності діагностики, що є важливим кроком у покращенні профілактики та лікування цукрового діабету.

Завдання дослідження

Для досягнення мети дослідження необхідно провести аналіз наукової літератури та сучасних технологій для вивчення існуючих підходів до прогнозування, діагностики та скринінгу цукрового діабету 2 типу із застосуванням методів машинного навчання та штучного інтелекту. Також потрібно визначити та обрати найбільш ефективні алгоритми для аналізу медичних даних, таких як логістична регресія, дерева рішень, метод опорних векторів, градієнтний бустинг та нейронні мережі [9] [7] [8].

Необхідно зібрати медичні дані з різних джерел, включаючи електронні медичні записи, клінічні бази та національні реєстри, а також підготувати ці дані шляхом видалення пропусків, нормалізації та балансування класів для забезпечення коректного аналізу. Далі слід розробити моделі машинного навчання для прогнозування ризику розвитку цукрового діабету, виявлення преддіабетичних станів та оцінки факторів ризику. Проведення оцінки точності, чутливості, специфічності та інших ключових метрик моделей є важливим для визначення їхньої ефективності [7].

Крім цього, потрібно інтегрувати створені моделі у веборієнтовану систему шляхом розробки API для взаємодії з моделями та створення зручного вебінтерфейсу для медичних працівників і пацієнтів. Завершальним етапом є проведення комплексного тестування створеної системи, включаючи функціональні, стресові та юзабіліті-тести, для забезпечення її надійності та зручності використання. На основі отриманих результатів необхідно розробити рекомендації щодо впровадження створеної системи у клінічну практику та дослідити можливості її адаптації для вирішення інших медичних завдань.

Методи дослідження

Методи дослідження базуються на застосуванні підходів машинного навчання, яке є підгалуззю штучного інтелекту, та спрямоване на розробку алгоритмів, що дозволяють комп'ютерам навчатися та здійснювати передбачення на основі попереднього досвіду. Цей підхід тісно пов'язаний зі статистикою та аналізом даних, забезпечуючи інструменти для моделювання складних залежностей і автоматизації процесу прийняття рішень [7].

Процес навчання моделей починається зі збору даних, які можуть бути отримані з електронних медичних записів, баз даних лікарень, національних реєстрів та інших джерел. Зібрані дані проходять етап підготовки, що включає очищення, нормалізацію, видалення пропусків, виявлення аномалій та обробку викидів. Ця підготовка є критично важливою, адже якість початкових даних безпосередньо впливає на точність і надійність моделей [7] [6].

Наступним кроком є побудова моделі машинного навчання. Для цього використовуються різноманітні алгоритми, такі як логістична регресія, дерева рішень, метод опорних векторів, градієнтний бустинг та глибокі нейронні мережі. Вибір алгоритму залежить від характеру даних, цільової задачі (класифікація, регресія чи кластеризація) та вимог до продуктивності. Алгоритми аналізують зв'язки між параметрами даних та формують прогностичну модель, здатну робити точні передбачення.

Тестування є невід'ємною частиною процесу дослідження. Для оцінки моделей використовуються метрики продуктивності, такі як точність, чутливість, специфічність, F1-міра та AUC-ROC. Тестування дозволяє визначити сильні та слабкі сторони моделі, а також виявити можливості для її покращення [7].

Процес оптимізації моделі спрямований на підвищення її ефективності та стабільності. Це може включати налаштування гіперпараметрів, вибір більш релевантних ознак, застосування методів ансамблювання або збагачення даних новими джерелами інформації. Крім того, для підвищення загальної продуктивності досліджується вплив передобробки даних та методів балансування класів на результативність.

Машинне навчання знаходить широке застосування у медицині, зокрема у прогнозуванні та діагностиці захворювань, таких як цукровий діабет [7]. Алгоритми дозволяють автоматично аналізувати великі обсяги медичних даних, виділяти ключові фактори ризику, класифікувати стан пацієнтів та оцінювати ймовірність розвитку ускладнень. Це сприяє покращенню якості медичних послуг та підвищенню точності діагностики.

Таким чином, методи дослідження у даній роботі базуються на інтеграції алгоритмів машинного навчання, статистичного аналізу та підготовки медичних даних для створення ефективних інструментів прогнозування, що сприяють розвитку сучасної медицини та інформатизації охорони здоров'я.

1. ОГЛЯД ЛІТЕРАТУРИ

1.1 Аналіз сучасного стану дослідження цукрового діабету та методів його виявлення

Цукровий діабет є однією з найбільш поширених ендокринних патологій, що вимагає постійного моніторингу, своєчасної діагностики та впровадження інноваційних підходів до лікування. В умовах зростання поширеності цього захворювання, сучасні дослідження зосереджуються на вдосконаленні методів виявлення, прогнозування та управління ним.

Одним із новітніх досягнень у цій галузі є використання генетичного тестування, що дозволяє визначити схильність до цукрового діабету, особливо 1-го типу. Генетичні аналізи дають можливість виявляти ризики на ранніх стадіях, а також прогнозувати ймовірність розвитку хвороби у майбутньому [10]. Важливою складовою діагностики залишаються клінічні аналізи, такі як вимірювання рівня глюкози в крові та визначення рівня глікованого гемоглобіну. Ці тести дозволяють оцінити стан пацієнта та визначити наявність переддіабетичного стану чи самого діабету [11].

Іншим перспективним напрямком є дослідження нових біомаркерів, які асоціюються із захворюванням [10]. Вони сприяють спрощенню діагностики, забезпечують більш точні результати та скорочують час, необхідний для моніторингу стану пацієнтів. Системи неперервного моніторингу глюкози (СНМГ) також набувають широкого використання. Ці технології забезпечують автоматичний збір даних про рівень глюкози у режимі реального часу, дозволяючи пацієнтам і лікарям краще контролювати хворобу.

Сучасні технології відкривають нові можливості у лікуванні цукрового діабету. Штучні підшлункові залози (ШПЗ) є одним із таких рішень. Ця технологія автоматично регулює рівень глюкози в крові та дозування інсуліну, що значно полегшує життя пацієнтів із цукровим діабетом 1-го типу [10]. Інший напрямок досліджень — це клітинна та генна терапія, яка спрямована на відновлення функції

бета-клітин підшлункової залози. Цей підхід може потенційно змінити сучасні підходи до лікування діабету, зробивши його менш залежним від зовнішніх джерел інсуліну [10].

Системи СНМГ, що поєднують моніторинг рівня глюкози з технологіями прогнозування, допомагають пацієнтам своєчасно коригувати дози інсуліну та запобігати гіпоглікемії чи гіперглікемії. Впровадження цих систем у широку практику сприяє підвищенню якості лікування та комфорту пацієнтів [10].

Перспективи розвитку методів діагностики та лікування цукрового діабету тісно пов'язані з інтеграцією сучасних технологій у медичну практику. Розробка програмно-апаратних комплексів для моніторингу рівня глюкози, прогнозування розвитку захворювання та аналізу факторів ризику відкриває нові можливості для персоналізованої медицини [12] [13]. Такі системи дозволяють забезпечити цілодобовий контроль за станом пацієнтів, оперативно реагувати на зміни та мінімізувати ризики ускладнень.

Аналіз сучасного стану досліджень у галузі цукрового діабету демонструє активний розвиток інноваційних підходів до діагностики та лікування. Генетичні тести, дослідження нових біомаркерів, впровадження штучних підшлункових залоз і систем неперервного моніторингу глюкози є ключовими напрямками, що сприяють підвищенню ефективності лікування та якості життя пацієнтів. Подальший розвиток цих технологій дозволяє сподіватися на нові досягнення у боротьбі з цукровим діабетом [10] [12] [11].

1.2 Огляд існуючих автоматизованих систем прогнозування

Автоматизовані системи прогнозування відіграють важливу роль у сучасній науці, бізнесі та медицині, забезпечуючи ефективний аналіз даних і підтримку прийняття рішень. Наприклад, у медицині широко використовуються системи прогнозування для оцінки ризику розвитку захворювань, таких як цукровий діабет або серцево-судинні хвороби. У сфері фінансів автоматизовані системи

допомагають у прогнозуванні ринкових трендів і ризиків, сприяючи більш обґрунтованому управлінню інвестиціями. У промисловості такі системи використовуються для оптимізації виробничих процесів, управління ланцюгами постачання та запобігання технічним несправностям. Завдяки використанню сучасних алгоритмів і технологій, ці системи дозволяють автоматизувати процеси обробки інформації, виявлення закономірностей і формування точних прогнозів на основі історичних даних.

Процес автоматизованого прогнозування починається зі збору великих обсягів даних. Дані можуть надходити з різних джерел, таких як бази даних, сенсори, електронні таблиці та API. Зібрані дані підлягають очищенню, що включає видалення помилок, заповнення пропусків і нормалізацію для забезпечення коректності подальшого аналізу. Цей етап є критично важливим, оскільки від якості даних залежить точність прогнозу [14].

Наступним етапом є аналіз даних, який здійснюється з використанням різноманітних статистичних методів, таких як регресійний аналіз, кореляційний аналіз і факторний аналіз, що дозволяють виявляти зв'язки між змінними. Для кластеризації використовуються алгоритми, такі як k-means, ієрархічна кластеризація та метод DBSCAN, які забезпечують поділ даних на групи з подібними характеристиками. У поєднанні з алгоритмами машинного навчання ці методи дозволяють створювати точні прогностичні моделі та формувати цінні інсайти. На цьому етапі системи виявляють приховані закономірності, аналізують зв'язки між змінними та формують попередні гіпотези. Результати аналізу використовуються для моделювання — створення математичних моделей прогнозування. Алгоритми, такі як регресійний аналіз, часові ряди, нейронні мережі або їх комбінації, дозволяють формувати прогностичні моделі, що забезпечують високу точність і адаптивність [15].

Останній етап — це прогнозування, яке здійснюється на основі створених моделей. Сучасні автоматизовані системи підтримують роботу в режимі реального

часу, що дозволяє оперативно реагувати на зміни в даних та забезпечувати швидке прийняття рішень [16].

Методи, що застосовуються в автоматизованих системах прогнозування, можна умовно поділити на три основні групи:

- Традиційні методи прогнозування, такі як регресійний аналіз, авторегресійні моделі (AR, ARMA, ARIMA) та адаптивні моделі, відіграють важливу роль у короткостроковому аналізі часових рядів. Ці методи використовують математичні та статистичні підходи для виявлення закономірностей у даних і побудови моделей, що дозволяють робити прогнози з високою точністю. Завдяки своїй простоті та надійності вони залишаються популярними в багатьох галузях, таких як економіка, фінанси та управління запасами, де важливо передбачити зміни в короткій перспективі [15].
- Методи машинного навчання є потужними інструментами для обробки та аналізу великих обсягів даних. Вони використовують складні алгоритми, такі як випадкові ліси, градієнтний бустинг і глибокі нейронні мережі (LSTM), які дозволяють виявляти нелінійні закономірності та приховані взаємозв'язки у даних. Завдяки здатності працювати з багатовимірними наборами даних і адаптуватися до їхньої структури, ці методи широко застосовуються у сферах, де точність прогнозування та виявлення аномалій мають вирішальне значення. Наприклад, у медицині вони використовуються для діагностики захворювань та прогнозування ризиків, а у фінансах — для управління портфелями та виявлення шахрайства. Завдяки своїй універсальності та ефективності, методи машинного навчання є основою сучасних автоматизованих систем прогнозування [16] [17].
- Методи нечіткої логіки є важливим інструментом для роботи з даними, що характеризуються високою невизначеністю або неточністю. Ці методи дозволяють створювати прогностичні моделі, які можуть адаптуватися до динамічних змін у даних, враховуючи нечіткість і неоднозначність інформації. Такий підхід особливо ефективний у нестабільних умовах, де традиційні

математичні моделі можуть виявитися недостатньо гнучкими. Наприклад, у медицині методи нечіткої логіки використовуються для оцінки ризиків захворювань або прийняття рішень в умовах обмеженої кількості даних. Завдяки своїй здатності враховувати контекст і нюанси ситуації, ці методи знаходять широке застосування в управлінні, прогнозуванні та розробці складних систем [15].

Процеси автоматизованого прогнозування надають суттєві переваги перед традиційними підходами. Вони дозволяють значно скоротити час, необхідний для збору, обробки та аналізу інформації, що робить побудову прогнозів швидшою та ефективнішою. Завдяки цьому можна обробляти великі обсяги даних, оптимізуючи роботу з ними та підвищуючи точність прийняття рішень. Особливо це актуально у сферах, де швидкість прогнозування є ключовим фактором досягнення успіху.

Автоматизація дозволяє суттєво зменшити час, який необхідний для збору, обробки та аналізу даних, що значно прискорює побудову прогнозів. Завдяки автоматизованим процесам можна ефективно опрацьовувати великі обсяги інформації, оптимізуючи роботу з даними та забезпечуючи швидкість і точність у прийнятті рішень. Це особливо актуально для сучасних галузей, де своєчасність прогнозів є ключовим фактором для досягнення успіху.

Автоматизовані системи прогнозування суттєво знижують ризик помилок, пов'язаних із людським фактором, завдяки повній автоматизації процесу обробки даних. Це дозволяє забезпечити високу об'єктивність результатів, що є важливим у галузях, де точність прогнозів критично важлива. Використання потужних алгоритмів для аналізу великих масивів даних гарантує формування достовірних висновків, які лягають в основу ефективних управлінських рішень.

Ці системи відзначаються високою здатністю до адаптації, що дозволяє їм успішно справлятися із зростаючими обсягами даних і все більш складними завданнями. Завдяки масштабованим алгоритмам та гнучким програмним платформам вони залишаються ефективними навіть у динамічних середовищах. Така універсальність забезпечує їхнє широке застосування в різних галузях,

включаючи медицину, фінанси, енергетику та промисловість, де їхня продуктивність сприяє досягненню стратегічних цілей.

Автоматизовані системи сучасного покоління підтримують роботу в реальному часі, забезпечуючи оперативну обробку та аналіз даних. Це особливо важливо для таких галузей, як медицина, де затримки у прогнозуванні можуть негативно вплинути на своєчасність надання допомоги пацієнтам, або фінанси, де швидкість реагування на ринкові зміни визначає успіх прийнятих рішень. Можливість працювати в режимі реального часу дозволяє швидко адаптувати стратегії, підвищуючи їхню ефективність і точність.

Наявні автоматизовані системи прогнозування показують високий потенціал у різноманітних сферах, таких як медицина, фінанси, логістика та енергетика. Наприклад, у медицині вони допомагають передбачати розвиток захворювань та оцінювати ризики, тоді як у фінансах їх використовують для аналізу ринкових трендів і виявлення фінансових шахрайств. У логістиці такі системи сприяють оптимізації маршрутів доставки, а в енергетиці — прогнозуванню попиту на електроенергію. Інтеграція інноваційних технологій, зокрема машинного навчання та нечіткої логіки, робить ці системи ще більш точними, адаптивними та зручними у використанні, відкриваючи нові перспективи для їхнього розвитку.

1.3 Сучасні підходи машинного навчання у медицині

Машинне навчання (МН) та штучний інтелект (ШІ) стали невід'ємною частиною сучасної медицини, суттєво змінюючи підходи до діагностики, лікування та управління охороною здоров'я. Наприклад, системи на базі нейронних мереж активно використовуються для аналізу медичних зображень, таких як рентгенівські та МРТ-знімки, виявлення патологій і прогнозування розвитку хвороб. Інші приклади включають IBM Watson, який допомагає у виборі терапії для пацієнтів із онкологічними захворюваннями, та Google Health, що розробляє моделі для прогнозування серцево-судинних ризиків. Завдяки цим технологіям можливим

стало аналізувати великі обсяги медичних даних, що забезпечує більш точні та своєчасні рішення у медичній практиці.

Ці алгоритми, зокрема глибокі нейронні мережі, такі як згорткові нейронні мережі (CNN), дозволяють автоматично виявляти патології, включаючи рак та інсульти, завдяки їхній здатності аналізувати зображення на рівні дрібних деталей. Це значно підвищує точність діагностики навіть у складних випадках, надаючи лікарям додаткову підтримку у виявленні аномалій. Системи можуть виявляти аномалії, які можуть бути пропущені лікарями, завдяки високій чутливості до деталей. Застосування ІІІ також охоплює персоналізоване лікування. Аналіз історії хвороби, генетичних даних і способу життя пацієнта дає змогу створювати індивідуальні плани лікування, прогнозувати ризики захворювань і вибирати оптимальні терапії, що значно підвищує ефективність медичної допомоги [18] [19].

Технології машинного навчання активно інтегруються у системи моніторингу стану пацієнтів, забезпечуючи реальний час для аналізу даних. Це стає особливо важливим під час пандемій або для пацієнтів із хронічними захворюваннями, яким необхідний постійний контроль. Крім того, МН відіграє ключову роль у ранньому виявленні захворювань. Наприклад, алгоритми допомагають у діагностиці спадкових патологій та різних форм раку. Системи на основі IBM Watson аналізують великі обсяги медичних даних для забезпечення високоточної діагностики та обґрунтованих рекомендацій [20] [21].

Ще одним важливим напрямком є розробка нових лікарських засобів. Алгоритми машинного навчання значно пришвидшують процес відкриття ліків, симулюючи клінічні випробування та аналізуючи ефективність різних підходів до лікування. Це дозволяє зменшити витрати на дослідження та прискорити виведення інноваційних препаратів на ринок. Наприклад, компанія Insilico Medicine використовує алгоритми машинного навчання для ідентифікації потенційних молекул-лідерів, що можуть стати основою нових ліків. Їхній підхід дозволив скоротити час на пошук ефективних сполук для лікування легеневої фіброзу та інших захворювань, що демонструє ефективність технології у фармацевтичних

дослідженнях. Усі ці підходи активно підтримуються технологічними інноваціями. Зокрема, глибоке навчання використовується для обробки неструктурованих даних, таких як зображення та текстові медичні звіти, що дозволяє досягати більшої точності у складних випадках [19] [22].

Аналіз великих даних (Big Data) став важливим інструментом у сучасній медицині. Завдяки обробці великих обсягів інформації лікарі можуть ухвалювати обґрунтовані рішення щодо лікування, прогнозування розвитку захворювань і оптимізації ресурсів. Наприклад, використання Big Data у клінічних дослідженнях дозволяє виявляти ефективність лікування пацієнтів із рідкісними захворюваннями, а також визначати найбільш підходящі терапії. Один із яскравих прикладів — проєкти, які аналізують дані мільйонів пацієнтів для виявлення патернів і оптимізації лікувальних стратегій у реальному часі. Ця технологія забезпечує більш точні прогнози та дозволяє виявляти закономірності, які не завжди очевидні навіть для досвідчених медиків [18] [22].

Сучасні підходи машинного навчання у медицині створюють нові можливості для вдосконалення медичних послуг. Інтеграція ШІ сприяє підвищенню точності діагностики, ефективності лікування та персоналізації медичної допомоги, що є ключовими чинниками розвитку сучасної системи охорони здоров'я. Завдяки швидкому розвитку технологій, ШІ і МН стають важливими інструментами у вирішенні складних медичних завдань [21] [22].

1.4 Вибір і обґрунтування напрямків дослідження

Дослідження автоматизованих систем прогнозування виявлення цукрового діабету (ЦД) є особливо важливим у зв'язку зі стрімким зростанням поширеності цього захворювання та значними медичними й соціально-економічними наслідками, які воно спричиняє. У 2023 році в Україні було зареєстровано понад 531 тисячу випадків ЦД, що свідчить про зростання порівняно з попереднім роком. Глобально кількість людей із ЦД зросла з 198 мільйонів у 1990 році до 828 мільйонів у 2022 році.

Таке поширення захворювання вимагає впровадження інноваційних технологій для покращення діагностики та прогнозування ЦД, що потребує глибокого аналізу та розробки ефективних рішень. Одним із ключових аспектів є дослідження сучасних методів моніторингу рівня глюкози. Ці системи, серед яких інвазивні та неінвазивні пристрої, забезпечують пацієнтам можливість контролювати рівень глюкози у крові в реальному часі. Проте в Україні спостерігається потреба у вдосконаленні таких систем через обмежену функціональність доступних рішень. Це ускладнює раннє виявлення ЦД та прогнозування розвитку захворювання, що зумовлює необхідність розробки більш точних і доступних технологій моніторингу [12].

Важливим напрямком дослідження є створення алгоритмів для прогнозування розвитку ЦД. Застосування машинного навчання, зокрема алгоритмів, таких як дерева рішень, випадкові ліси, логістична регресія та XGBoost, дозволяє досягати високої точності прогнозування. Наприклад, дерева рішень широко застосовуються для визначення ключових факторів ризику розвитку цукрового діабету, а XGBoost використовується для створення багатовимірних моделей, які враховують взаємодію між різними показниками. Випадкові ліси надають змогу створювати ансамблі моделей для зменшення похибок, тоді як логістична регресія є ефективним інструментом для прогнозування ймовірності захворювання на основі сукупності параметрів пацієнта. Використання відкритих баз даних про захворювання може забезпечити ефективне навчання таких алгоритмів, що сприятиме їх адаптації до умов реальної клінічної практики. Це відкриває можливості для розробки нових підходів до персоналізованої медицини [12].

Розробка концептуальної моделі системи збору та обробки даних також є одним із перспективних напрямків. Інтеграція локальних і центральних вузлів управління дозволить ефективно поєднувати різні джерела даних, включаючи мобільні додатки та IoT-пристрої. Такий підхід забезпечить цілодобовий моніторинг стану пацієнтів та підвищить точність прогнозування захворювання. Важливим є

також впровадження розподілених архітектур, які підвищують надійність і масштабованість систем [23].

Інноваційні технології, такі як мобільні додатки та інтернет речей (IoT), відкривають нові горизонти у сфері автоматизації збору медичних даних. Наприклад, система Dexcom G6 (рис. 2.1), яка використовує IoT, забезпечує безперервний моніторинг рівня глюкози в крові та передає ці дані на мобільні пристрої пацієнта та лікаря. Це дозволяє здійснювати своєчасну корекцію терапії та запобігати ускладненням. Зокрема, пристрої, що використовують IoT, можуть забезпечувати безперервний моніторинг рівня глюкози та передавати ці дані до централізованих систем для подальшого аналізу. Це не лише підвищує зручність для пацієнтів, але й дозволяє лікарям своєчасно отримувати інформацію про стан здоров'я пацієнтів, що є критично важливим у випадку гострих ускладнень [23].



Рис. 1.1. Загальний вигляд сенсора Dexcom G6

Соціально-економічний аспект впровадження автоматизованих систем прогнозування виявлення ЦД також заслуговує на увагу. Наприклад, дослідження показують, що раннє виявлення ЦД за допомогою таких систем може скоротити витрати на лікування ускладнень на 20–30%. У масштабах охорони здоров'я це дозволяє зекономити мільйони гривень щороку. Крім того, використання цих систем може зменшити кількість госпіталізацій на 15–25%, що суттєво розвантажує медичні установи і підвищує їхню ефективність. Зростання захворюваності на ЦД призводить до значних витрат на лікування та профілактику ускладнень. Розробка ефективних систем моніторингу може значно скоротити ці витрати завдяки ранньому виявленню захворювання, що сприяє зменшенню кількості госпіталізацій

і тривалості перебування пацієнтів у стаціонарах. Крім того, це забезпечить підвищення якості життя пацієнтів та зменшення навантаження на систему охорони здоров'я [24].

Вибір напрямків дослідження автоматизованих систем прогнозування виявлення цукрового діабету базується на необхідності покращення якості моніторингу, розробки точних алгоритмів прогнозування, створення інтегрованих систем збору даних та врахування соціально-економічних факторів. Такі дослідження сприятимуть розвитку медицини, підвищенню ефективності лікування та покращенню загального рівня охорони здоров'я.

2. МЕТОДОЛОГІЯ ТА МАТЕРІАЛИ

2.1 Опис використовуваних медичних даних (джерела, обсяги, формат)

Для навчання моделей прогнозування цукрового діабету було використано набір даних, що складається з 10 000 записів [25]. Кожен запис представляє індивідуальні медичні показники пацієнтів, які зібрані з різних медичних джерел та містять ключові параметри, що впливають на стан здоров'я та ризик розвитку цукрового діабету (рис. 2.1).

	A	B	C	D	E	F	G	H	I
1	gender	age	hypertension	heart_disease	smoking_histor	bmi	HbA1c_level	blood_glucose_level	diabetes
2	Female	80	0	1	never	25,19	6,6	140	0
3	Female	54	0	0	No Info	27,32	6,6	80	0
4	Male	28	0	0	never	27,32	5,7	158	0
5	Female	36	0	0	current	23,45	5,0	155	0
6	Male	76	1	1	current	20,14	4,8	155	0
7	Female	20	0	0	never	27,32	6,6	85	0
8	Female	44	0	0	never	19,31	6,5	200	1
9	Female	79	0	0	No Info	23,86	5,7	85	0
10	Male	42	0	0	never	33,64	4,8	145	0
11	Female	32	0	0	never	27,32	5,0	100	0
12	Female	53	0	0	never	27,32	6,1	85	0
13	Female	54	0	0	former	54,70	6,0	100	0
14	Female	78	0	0	former	36,05	5,0	130	0
15	Female	67	0	0	never	25,69	5,8	200	0
16	Female	76	0	0	No Info	27,32	5,0	160	0
17	Male	78	0	0	No Info	27,32	6,6	126	0
18	Male	15	0	0	never	30,36	6,1	200	0
19	Female	42	0	0	never	24,48	5,7	158	0
20	Female	42	0	0	No Info	27,32	5,7	80	0
21	Male	37	0	0	ever	25,72	3,5	159	0
22	Male	40	0	0	current	36,38	6,0	90	0
23	Male	5	0	0	No Info	18,80	6,2	85	0
24	Female	69	0	0	never	21,24	4,8	85	0
25	Female	72	0	1	former	27,94	6,5	130	0
26	Female	4	0	0	No Info	13,99	4,0	140	0

Рис. 2.1. Загальний вигляд набору даних

Основні характеристики даних:

1. **Стать пацієнтів ("gender").** Атрибут "gender" визначає стать пацієнтів і представлений категоріальними значеннями: "Female" або "Male". Це дозволяє враховувати гендерні відмінності у медичних показниках, які можуть впливати на ризик розвитку цукрового діабету.

2. **Вік пацієнтів ("age").** Атрибут "age" подано у числовому форматі з плаваючою точкою. Цей параметр забезпечує аналіз у широкому віковому діапазоні, що дозволяє враховувати вік як важливий фактор ризику.
3. **Наявність гіпертонії ("hypertension").** Атрибут "hypertension" є бінарним і відображає наявність (1) або відсутність (0) гіпертонії у пацієнтів. Гіпертонія є одним із ключових факторів ризику розвитку цукрового діабету та його ускладнень.
4. **Серцеві захворювання ("heart_disease").** Атрибут "heart_disease" також є бінарним і позначає наявність (1) або відсутність (0) серцевих захворювань. Цей параметр дозволяє враховувати кардіологічні ризики, які можуть супроводжувати цукровий діабет.
5. **Історія куріння ("smoking_history").** Атрибут "smoking_history" містить кілька категорій: "no info", "never", "former", "current", "not current". Категорія "no info" означає, що інформація про куріння відсутня. "Never" вказує на осіб, які ніколи не курили, тоді як "former" описує тих, хто курих у минулому, але зараз припинив. "Current" означає активних курців, а "not current" стосується осіб, які раніше курили, але вже не практикують цю звичку. Цей параметр дозволяє аналізувати вплив куріння на здоров'я та зв'язок із ризиком розвитку діабету. Цей параметр описує вплив звички куріння на здоров'я пацієнта та дозволяє аналізувати його зв'язок із ризиком захворювання.
6. **Індекс маси тіла ("bmi").** Атрибут "bmi" представлений числовим значенням з плаваючою точкою. Індекс маси тіла є ключовим показником фізіологічного стану пацієнта, що використовується для оцінки ризиків, пов'язаних із надмірною вагою або ожирінням. Порогові значення ВМІ, що вважаються критичними для ризику цукрового діабету, включають 25 і більше для надмірної ваги та 30 і більше для ожиріння. Ці значення дозволяють визначати пацієнтів із підвищеним ризиком розвитку діабету.
7. **Рівень глікованого гемоглобіну ("HbA1c_level").** Атрибут "HbA1c_level" подано у числовому форматі та використовується для оцінки довгострокового

контролю рівня глюкози в крові. Глікований гемоглобін є хімічним з'єднанням, що утворюється в результаті зв'язування глюкози з гемоглобіном у еритроцитах. Оскільки еритроцити мають середню тривалість життя близько 120 днів, рівень HbA1c відображає середню концентрацію глюкози в крові за останні 2-3 місяці. Значення HbA1c нижче 5.7% вважаються нормальними, показники в межах від 5.7% до 6.4% вказують на переддіабет, а рівень 6.5% і вище свідчить про наявність діабету. Показник HbA1c є особливо корисним для оцінки ефективності лікування, оскільки не залежить від щоденних коливань рівня глюкози в крові. Високий рівень HbA1c може свідчити про недостатній контроль захворювання, що збільшує ризик ускладнень, таких як ретинопатія, нефропатія та серцево-судинні захворювання. Цей показник допомагає лікарям не лише класифікувати пацієнтів, але й планувати довгострокові стратегії лікування.

8. **Рівень глюкози у крові ("blood_glucose_level").** Атрибут "blood_glucose_level" представлений числовим значенням, зазвичай у міліграмах на децилітр (mg/dL) або мілімолях на літр (mmol/L). Нормальний рівень глюкози у крові натще становить від 70 до 99 mg/dL (3.9–5.5 mmol/L), а значення від 100 до 125 mg/dL (5.6–6.9 mmol/L) вказують на переддіабет. Рівні глюкози 126 mg/dL (7.0 mmol/L) і вище свідчать про наявність діабету. Цей параметр є ключовим для діагностики та контролю цукрового діабету, дозволяючи оцінювати ефективність лікування та ризики ускладнень. В датасеті значення представлені в міліграмах на децилітр (mg/dL).
9. **Наявність діабету ("diabetes").** Атрибут "diabetes" є цільовим параметром для задач класифікації та визначає наявність (1) або відсутність (0) цукрового діабету у пацієнта. Цей показник використовується для побудови моделей прогнозування.

Дані представлені у форматі CSV, що забезпечує зручність у використанні та інтеграції з інструментами аналізу даних і машинного навчання. Наприклад, бібліотеки Python, такі як pandas і numpy, дозволяють ефективно обробляти цей

формат, тоді як інструменти машинного навчання, як-от scikit-learn, TensorFlow чи PyTorch, використовують ці дані для побудови моделей. Така структура даних дозволяє легко проводити попередню обробку, аналіз і створення моделей, орієнтованих на прогнозування та управління ризиками, пов'язаними з цукровим діабетом [25].

2.2 Підготовка даних

Підготовка даних є критично важливим етапом у розробці моделей машинного навчання. Вона забезпечує високу точність та ефективність моделей, зменшує вплив шумів у даних і сприяє коректному навчанню алгоритмів. Цей процес включає видалення пропусків, нормалізацію та обробку викидів, які виконуються з урахуванням специфіки набору даних і цільової задачі.

Відсутні значення в даних можуть суттєво вплинути на якість моделювання. Наприклад, у наборі даних для прогнозування серцево-судинних захворювань пропуски в параметрах, таких як рівень холестерину чи кров'яного тиску, значно знижували точність моделей. Заповнення цих пропусків середніми значеннями допомогло підвищити точність прогнозування на 15%, що підкреслює важливість якісної обробки відсутніх даних. Для обробки пропусків застосовуються кілька підходів. Один із найпоширеніших — заповнення пропущених значень середнім, медіанним чи модальним значенням, що дозволяє зберегти структуру набору даних. У складніших випадках можливе прогнозування пропусків за допомогою алгоритмів машинного навчання. У разі значної кількості пропусків (понад 80% від загальної кількості записів) доцільно видаляти відповідні рядки або стовпці, щоб уникнути спотворення моделі. Пропуски також можна ігнорувати, якщо їхня кількість є мінімальною і вони не впливають на загальний результат [26] [27].

Нормалізація даних — це процес приведення всіх числових ознак до одного масштабу. Це є важливим у таких задачах, як кластеризація, де використовуються методи на основі відстаней, або для алгоритмів машинного навчання, чутливих до масштабу ознак, наприклад, градієнтний спуск. Ненормалізовані дані можуть

призводити до некоректного навчання моделей, коли певні ознаки домінують через більший числовий діапазон. Це є важливим для алгоритмів, чутливих до масштабу ознак, таких як методи на основі відстаней (наприклад, k-найближчих сусідів). Мін-макс нормалізація зводить значення до діапазону $[0, 1]$, що дозволяє уникнути домінування одних ознак над іншими. Інший підхід — стандартизація, яка забезпечує середнє значення 0 і стандартне відхилення 1. Це особливо ефективно для наборів даних із нормальним розподілом [26] [28].

Мін-макс нормалізація: Перетворює всі значення в діапазон $[0, 1]$ за формулою (2.1):

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad 2.1$$

Стандартизація: Перетворює дані так, щоб вони мали середнє 0 і стандартне відхилення 1 (3.2):

$$z = \frac{x - \mu}{\sigma} \quad 2.2$$

де μ — середнє значення, а σ — стандартне відхилення.

Викиди або аномалії в наборі даних можуть спотворити результати навчання моделі. Для їх виявлення використовуються статистичні методи, такі як Z-оцінка, або спеціалізовані алгоритми, наприклад, Isolation Forest. Isolation Forest працює шляхом побудови дерева, яке ізолює аномальні точки даних шляхом розбиття їх за значеннями ознак. Аномальні значення, як правило, ізолюються швидше через їхню унікальність. Наприклад, цей алгоритм успішно використовується для виявлення шахрайства у фінансових транзакціях, де окремі транзакції можуть суттєво відрізнятися від типової поведінки користувачів. Викиди можуть бути видалені, якщо вони не несуть суттєвої інформації, або скориговані шляхом заміни середнім чи медіанним значенням. У складних випадках можливе використання алгоритмів машинного навчання для моделювання та корекції аномалій. Наприклад, аномалії у фінансових даних або даних медичного моніторингу можуть свідчити про значущі події, що потребують особливої уваги [29] [27].

Підготовка даних включає також кодування категоріальних змінних для використання у моделях машинного навчання. Наприклад, у задачі класифікації клієнтів банку для прогнозування кредитоспроможності, використання One-Hot Encoding для змінної "Education" зі значеннями "High School", "Bachelor's", "Master's" підвищило точність моделі на 10%. Це дозволило ефективніше використовувати інформацію про рівень освіти у моделі, яка базувалася на алгоритмах логістичної регресії. Найчастіше використовуються методи One-Hot Encoding або Label Encoding. Наприклад, для змінної "gender" зі значеннями "Male" і "Female" можна створити дві окремі бінарні змінні. Цей підхід дозволяє ефективно використовувати категоріальні дані у числових моделях [28].

Процес підготовки даних є основою для успішного моделювання. Наприклад, моделі прогнозування, треновані на даних без видалення пропусків та нормалізації, показували точність 75%, тоді як після ретельної підготовки точність зросла до 90%. Це підкреслює значущість етапу підготовки для досягнення оптимальних результатів у машинному навчанні. Ретельно підготовлений набір даних дозволяє досягти високої продуктивності моделей, зменшити ризик помилок у прогнозах і забезпечити коректне функціонування алгоритмів. Успішна підготовка також сприяє більш швидкому навчання моделей і зменшує потребу в додаткових обчислювальних ресурсах.

2.3 Обґрунтування вибору моделей машинного навчання

Вибір моделей машинного навчання є критично важливим етапом у розробці ефективних рішень для різноманітних завдань. Наприклад, у задачі прогнозування ризику розвитку діабету вибір логістичної регресії дозволив швидко отримати початкові результати, тоді як використання градієнтного бустингу значно підвищило точність за рахунок виявлення складних взаємозв'язків між ознаками. Це обґрунтування базується на кількох ключових аспектах, які допомагають визначити, яка модель найкраще підходить для конкретної задачі. Важливість цього етапу полягає в тому, що неправильно обрана модель може призвести до низької

продуктивності та ненадійних результатів, навіть якщо дані є якісними та добре підготовленими.

Тип даних відіграє ключову роль у визначенні підходящої моделі. Наприклад, структуровані дані включають таблиці з числовими та категоріальними змінними, такі як медичні записи або фінансові транзакції, де часто застосовуються логістична регресія або методи ансамблю. Неструктуровані дані охоплюють текст, зображення чи аудіо, наприклад, медичні знімки або чат-бот діалоги, для яких оптимальними є нейронні мережі, такі як згорткові мережі для зображень або рекурентні мережі для тексту. Для структурованих даних, таких як таблиці з числовими значеннями, часто використовуються алгоритми, як-от логістична регресія, яка є швидким і простим рішенням для задач двокласової класифікації. У випадку складніших задач, де структура даних містить більше ознак, доцільно використовувати ансамблеві методи, такі як рандомний ліс (Random Forest) або градієнтний бустинг (XGBoost, CatBoost). Якщо ж дані є великими за обсягом і включають багато ознак, штучні нейронні мережі (ANN) стають оптимальним вибором завдяки їхній здатності ефективно обробляти складні багатовимірні дані.

Ціль задачі також визначає, яка модель буде найбільш доречною. У задачах класифікації, наприклад, для розподілу електронної пошти на спам і не спам, ефективними є логістична регресія, методи ансамблю (випадкові ліси, градієнтний бустинг) або нейронні мережі. У задачах з більш складними взаємозв'язками між ознаками методи опорних векторів (SVM) забезпечують високу точність, особливо для невеликих наборів даних із чіткими розділеннями між класами. Для задач регресії, таких як прогнозування числових значень, оптимальними будуть методи градієнтного бустингу або багатошарові перцептрони (MLP).

Обсяг та якість даних значно впливають на вибір моделі. Наприклад, у логістичній регресії низька якість даних, зокрема наявність шуму або пропусків, може призвести до значного зниження точності класифікації. Для нейронних мереж такі проблеми ускладнюють навчання, оскільки моделі можуть переобучуватися на шумі або неадекватно реагувати на пропуски, що робить попередню очистку та

нормалізацію даних критично важливими для їхньої продуктивності. Особливу увагу варто звернути на ансамблеві методи, такі як рандомний ліс і градієнтний бустинг, які мають низку переваг у задачах машинного навчання. Наприклад, у задачі прогнозування відтоку клієнтів у банківській сфері використання градієнтного бустингу дозволило досягти точності понад 90%, що значно перевищило результати, отримані за допомогою базових моделей, таких як логістична регресія. Цей підхід ефективно враховує складні взаємозв'язки між ознаками, забезпечуючи високу продуктивність навіть на нерівномірних наборах даних. Вони поєднують результати декількох базових моделей для підвищення точності та зменшення ймовірності помилок. Наприклад, рандомний ліс створює ансамбль дерев рішень і є стійким до шуму в даних, а градієнтний бустинг поступово зменшує похибки, додаючи моделі, що фокусуються на невирішених помилках. Це робить ці методи ідеальними для задач, де дані мають високу варіативність або складні взаємозв'язки між ознаками. Для великих наборів даних штучні нейронні мережі стають незамінним інструментом завдяки їхній масштабованості та здатності знаходити приховані залежності.

Процес вибору моделі включає кілька етапів (Додаток Б). Наприклад, оптимізація гіперпараметрів є одним із ключових кроків, що суттєво впливає на кінцеву продуктивність моделі. Для рандомного лісу це може включати налаштування кількості дерев або максимального розміру вузлів, що дозволяє досягти балансу між узагальнювальною здатністю та обчислювальною ефективністю. У задачах класифікації після оптимізації точність моделі може зрости на 5-10%, що забезпечує значно кращу адаптацію до тестових даних. Спочатку здійснюється збір та підготовка даних, які очищуються від пропусків, викидів та інших аномалій. Наступним кроком є визначення метрики успіху, наприклад, точність, F1-міра, середньоквадратична помилка (RMSE) або коефіцієнт детермінації (R^2), що допомагає об'єктивно оцінити якість моделі. Далі виконується тестування кількох моделей з використанням крос-валідації для забезпечення стабільності результатів. Після вибору найкращої моделі проводиться оптимізація її гіперпараметрів, наприклад, кількості дерев у рандомному лісі або швидкості

навчання у градієнтному бустингу. Завершальний етап включає оцінку моделі на тестових даних для перевірки її узагальнювальної здатності.

Обґрунтування вибору моделей машинного навчання базується на специфіці задачі, характеристиках даних та меті проекту. Вибір правильної моделі дозволяє досягти високої продуктивності та точності рішень, забезпечуючи оптимальне використання ресурсів і високу надійність системи. Машинне навчання продовжує розширювати свої можливості, демонструючи потужний інструмент для вирішення складних завдань у різних галузях, таких як медицина, фінанси та наука [30] [31] [32] [33].

2.4 Архітектура розробленої системи

Функціональна схема автоматизованої системи прогнозування виявлення цукрового діабету представлена у Додатку А.

Архітектура розробленої системи складається з бекенду, реалізованого на Flask, та фронтенду, побудованого на Vue.js. Ця комбінація забезпечує ефективне та гнучке середовище для створення інтерактивних веб-додатків, об'єднуючи потужність Python з легкістю JavaScript.

Бекенд системи побудований на Flask, мікрофреймворку для Python, який ідеально підходить для створення легковажних веб-додатків та API. Flask забезпечує легкість у розробці завдяки своїй мінімалістичній архітектурі. Це дозволяє розробникам швидко створювати веб-сервіси, зосереджуючись на бізнес-логіці. Однією з головних переваг Flask є його гнучкість, що дозволяє легко інтегрувати сторонні бібліотеки та інструменти. Flask також підтримує розширення, які значно спрощують реалізацію RESTful API. Наприклад, використання розширень Flask-RESTful та Flask-SQLAlchemy забезпечує легку інтеграцію з базами даних та створення ендпоінтів. Потужний шаблонний движок Jinja2 дозволяє генерувати динамічні сторінки HTML, що може бути корисним для створення серверно-генерованих інтерфейсів. Крім того, використання Flask як API дозволяє інтегрувати

сервіс не тільки з фронтендом, але й із іншими сервісами, програмами та мобільними додатками, забезпечуючи високу гнучкість та розширюваність. Завдяки цим перевагам Flask є ідеальним вибором для мікросервісних архітектур та швидкого прототипування [34] [36].

Фронтенд системи реалізований за допомогою Vue.js, сучасного JavaScript-фреймворку для створення інтерактивних інтерфейсів. Vue.js вирізняється своєю простотою та легкістю вивчення, що дозволяє навіть новачкам швидко освоїти основи фреймворку. Однією з головних переваг Vue.js є його реактивна система, яка автоматично оновлює інтерфейс при зміні даних. Це значно підвищує продуктивність і спрощує розробку інтерактивних компонентів. Vue.js також має потужну екосистему, яка включає такі інструменти, як Vue Router для маршрутизації та Vuex для управління станом програми. Оптимізований віртуальний DOM, використовуваний у Vue.js, забезпечує високу швидкість рендерингу компонентів. Крім того, Vue.js легко інтегрується в існуючі проекти, що робить його гнучким вибором для різних масштабів розробки.

Об'єднання Flask та Vue.js у розробці цієї системи забезпечує надійність та продуктивність. Завдяки розподіленню завдань між бекендом і фронтендом серверна частина може ефективно обробляти запити, забезпечуючи швидкий доступ до даних через API, тоді як клієнтська частина відповідає за оновлення інтерфейсу в реальному часі. Такий підхід дозволяє зменшити навантаження на сервер, покращуючи загальну продуктивність системи, особливо у випадках високого рівня користувацької активності. Flask виконує роль сервера, який обробляє запити та надає дані через API, тоді як Vue.js відповідає за динамічний та інтуїтивний інтерфейс користувача. Ця архітектура дозволяє розподілити навантаження між серверною та клієнтською частинами, забезпечуючи плавну роботу додатку та зручність для кінцевого користувача. Використання цієї технологічної зв'язки також сприяє гнучкості розробки, дозволяючи легко масштабувати проект та додавати нові функціональності [35] [36].

Висновок щодо вибору Flask як бекенду та Vue.js як фронтенду обґрунтований їхньою простотою, гнучкістю та широкими можливостями інтеграції. У порівнянні з Django, Flask пропонує більшу гнучкість і менше навантаження, що дозволяє створювати легковажні додатки швидше. Зі свого боку, Vue.js забезпечує легшу інтеграцію та кращу продуктивність для менш складних проєктів порівняно з React. Такий вибір дозволяє оптимально поєднувати простоту розробки з високою продуктивністю та ефективністю. Ця комбінація є ідеальним рішенням для створення сучасних веб-додатків, що відповідають вимогам продуктивності та зручності використання.

Citations:

2.5 Використовувані технології та програмні засоби

Розробка системи вимагала застосування сучасних технологій та інструментів, які забезпечують ефективність, зручність у роботі та можливості для масштабування. Вибір таких інструментів, як Git, GitHub, Visual Studio Code та HeroKi, базується на їх функціональності, популярності та перевагах у сучасному розробницькому середовищі.

Git є однією з найбільш популярних систем контролю версій, що дозволяє ефективно управляти змінами у коді. Його швидкість і ефективність забезпечують миттєве виконання операцій, таких як коміти, злиття та відкат змін. Розподілена архітектура Git дозволяє кожному розробнику мати повну копію репозиторію, що сприяє безпеці даних та можливості роботи офлайн. Гнучкість Git у співпраці дозволяє кільком розробникам працювати над одним проєктом одночасно, а потужні інструменти управління версіями спрощують повернення до попередніх станів проєкту. Попри складність для новачків, Git залишається незамінним інструментом для професійної розробки програмного забезпечення [37] [38] [39] [40].

GitHub є платформою для спільної роботи над програмним забезпеченням, яка базується на Git. GitHub забезпечує зручний інтерфейс для співпраці, інтеграції з

різними інструментами та управління проектами. У даній системі використовувалися GitHub для автоматизації розгортання, що дозволило значно скоротити час розробки та забезпечити стабільність випуску нових версій. Можливість інтеграції з CI/CD системами, такими як Jenkins , дозволяє автоматизувати процеси розгортання та тестування. Потужні функції контролю версій сприяють організованості проекту, а велика спільнота розробників надає доступ до корисних ресурсів, таких як документація та приклади коду. Інструмент GitHub Copilot, що базується на штучному інтелекті, підвищує продуктивність, допомагаючи автоматично генерувати код та рекомендації [37] [38] [39] [40].

Visual Studio Code (VS Code) є популярним редактором коду, який використовується завдяки своїй мультиплатформенності та підтримці великої кількості мов програмування. Наприклад, розширення Prettier для автоматичного форматування коду значно спростило підтримку єдиного стилю коду в проекті, що покращило зручність співпраці в команді. Його простий інтерфейс дозволяє як новачкам, так і досвідченим розробникам ефективно працювати над проектами. Вбудовані функції, такі як IntelliSense, підтримка налагодження та інтеграція з Git, підвищують продуктивність роботи. Розширюваність редактора за допомогою плагінів дозволяє адаптувати його під конкретні потреби проекту. Популярність VS Code серед розробників підкріплена опитуваннями Stack Overflow, які демонструють, що цей інструмент є найпоширенішим серед програмістів [38] [39] [41] [42].

Heroku є платформою як послуга (PaaS), яка забезпечує простоту розгортання, масштабованість і автоматизацію управління інфраструктурою. Наприклад, у даному проекті, використання Heroku дозволило розгорнути API для інтеграції з клієнтськими додатками менш ніж за дві години. Завдяки автоматизації розгортання та підтримці Python, налаштування середовища виконання стало простим і швидким, а можливість масштабування дозволила ефективно обробляти зростаючу кількість запитів без збоїв у роботі. Її інтуїтивно зрозумілий інтерфейс дозволяє розробникам швидко налаштовувати середовище для розробки без потреби в глибоких знаннях

системного адміністрування. Підтримка різноманітних мов програмування, таких як Python, Node.js, Ruby та Java, робить Heroku універсальним рішенням для різних типів проектів. Інтеграція з DevOps-практиками, такими як CI/CD, дозволяє автоматизувати процеси розгортання та підвищити ефективність командної роботи. Гнучкість у масштабуванні додатків забезпечує швидке реагування на зміни в обсягах трафіку, що є важливим для стартапів і зростаючих проектів [37] [38] [39].

Вибір цих технологій дозволив досягти високої продуктивності, зручності в роботі та надійності системи. Зокрема, використання Git спростило управління версіями та колаборацію, GitHub забезпечив автоматизацію CI/CD процесів, а Visual Studio Code покращив швидкість написання коду завдяки розширенням для форматування та автозаповнення. Heroku, у свою чергу, забезпечила легке розгортання додатку та його масштабування, що дозволило обробляти зростаючу кількість запитів без значного впливу на час відгуку. Git і GitHub забезпечили ефективне управління кодом і співпрацю між розробниками. Visual Studio Code сприяв продуктивності завдяки своєму функціоналу та розширенням. Heroku надала можливість швидкого розгортання та масштабування системи, що зробило її готовою до викликів реального використання.

3. РОЗРОБКА МОДЕЛЕЙ

3.1 Створення моделі машинного навчання на основі логістичної регресії

Логістична регресія є одним з найпопулярніших алгоритмів для вирішення задач класифікації, зокрема бінарної класифікації. Основна ідея алгоритму полягає у використанні логістичної функції (сигмоїди), яка дозволяє перетворити лінійну комбінацію вхідних ознак у ймовірності належності до певного класу. Вихід моделі представлений у вигляді значення від 0 до 1, що дозволяє інтерпретувати результат як імовірність.

Логістична функція є ключовим елементом моделі логістичної регресії, яка використовується для бінарної класифікації даних. Вона дозволяє перетворювати лінійні комбінації незалежних змінних у ймовірності, обмежені діапазоном від 0 до 1. Це досягається через сигмоїдальну функцію, яка має форму:

$$p = \frac{1}{1 + e^{-z}} \quad (3.1)$$

Де z є лінійною комбінацією вхідних змінних, вираженою як:

$$z = W^T X \quad (3.2)$$

тут W — вектор вагових коефіцієнтів, а X — вектор значень ознак об'єкта [43] [44].

Навчання моделі здійснюється шляхом оптимізації логістичної функції втрат (log-loss), яка мінімізує різницю між передбаченнями моделі та реальними значеннями. Для цієї оптимізації зазвичай використовується метод градієнтного спуску.

Підготовка даних

Завантаження даних і їх попередня обробка є ключовими етапами для забезпечення високої якості моделі. Вибір методу обробки пропущених даних залежить від їх кількості та важливості ознак. Наприклад, якщо відсутніх значень небагато, їх можна заповнити середнім або медіанним значенням, а якщо значна

частина даних відсутня, доцільно видалити ці ознаки або рядки. Масштабування числових даних виконується для забезпечення однакового внеску всіх ознак у модель, що є критично важливим для алгоритмів, чутливих до масштабу, таких як логістична регресія. Для навчання моделі використовується набір даних "diabetes_prediction_dataset.csv".

```
# Завантаження даних
import pandas as pd
import os
from sklearn.preprocessing import LabelEncoder, StandardScaler

data = pd.read_csv(os.path.join(Config.TEACH_DIR, 'diabetes_prediction_dataset.csv'))

# Перевірка на пропущені значення
print("Пропущені значення в кожній колонці:")
print(data.isnull().sum())

# Кодифікація категоріальних даних
label_encoders = {
    "gender": {"Female": 0, "Male": 1},
    "smoking_history": {"No Info": 2, "never": 0, "former": 1, "current": 1, "not
current": 1}
}
for col in ['gender', 'smoking_history']:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col])

# Масштабування числових даних
scaler = StandardScaler()
numerical_columns = ['age', 'bmi', 'HbA1c_level', 'blood_glucose_level']
data[numerical_columns] = scaler.fit_transform(data[numerical_columns])
```

Навчання моделі

Для навчання моделі логістичної регресії дані розділяються на тренувальну та тестову вибірки. Розподіл 80/20 було обрано, оскільки він забезпечує достатньо даних для навчання моделі, залишаючи при цьому значну частину для оцінки її продуктивності. Такий підхід дозволяє збалансувати точність і узагальнюючу здатність моделі. Потім модель тренується за допомогою тренувальної вибірки.

```
from sklearn.model_selection import train_test_split
```

```

from sklearn.linear_model import LogisticRegression

# Розділення на ознаки та цільову змінну
X = data.drop('diabetes', axis=1)
y = data['diabetes']

# Розділення на тренувальні та тестові дані
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Навчання моделі
model = LogisticRegression(random_state=42)
model.fit(X_train, y_train)

```

Оцінка моделі

Оцінка моделі здійснюється за допомогою метрик, таких як класифікаційний звіт, матриця плутанини та ROC-AUC. Значення ROC-AUC відображає здатність моделі розрізняти класи: якщо AUC дорівнює 1, це означає ідеальну класифікацію, тоді як значення 0.5 вказує на випадкове передбачення. У контексті цієї задачі високе значення ROC-AUC свідчить про те, що модель успішно визначає пацієнтів із діабетом та без нього, навіть за умови неоднорідності даних. Візуалізація ROC-кривої дозволяє краще зрозуміти продуктивність моделі (рис. 3.1).

```

from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score,
roc_curve
import matplotlib.pyplot as plt

# Прогнозування
y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]

# Метрики
print("Класифікаційний звіт:")
print(classification_report(y_test, y_pred))

print("Матриця плутанини:")
print(confusion_matrix(y_test, y_pred))

roc_auc = roc_auc_score(y_test, y_proba)
print(f"ROC-AUC: {roc_auc}")

# Візуалізація ROC-кривої
fpr, tpr, _ = roc_curve(y_test, y_proba)
plt.figure(figsize=(10, 6))

```

```
plt.plot(fpr, tpr, color='blue', label=f"ROC Curve (AUC = {roc_auc:.2f})")
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.grid()
plt.show()
```

Класифікаційний звіт:

	precision	recall	f1-score	support
0	0.96	0.99	0.98	18292
1	0.86	0.61	0.72	1708
accuracy			0.96	20000
macro avg	0.91	0.80	0.85	20000
weighted avg	0.96	0.96	0.96	20000

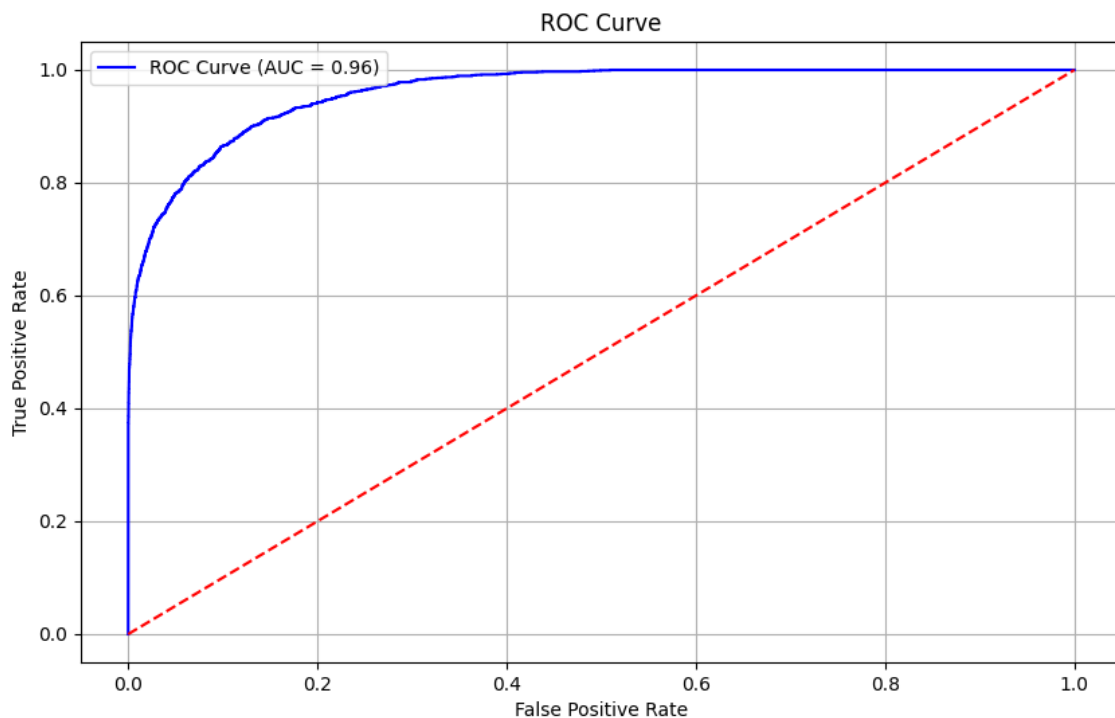


Рис. 3.1 ROC крива навчання логістичної регресії

ROC (Receiver Operating Characteristic) крива демонструє співвідношення між чутливістю (True Positive Rate, TPR) та специфічністю (1 - False Positive Rate, FPR) моделі. Синя крива на графіку представляє ROC-криву нашої моделі логістичної

регресії. Вона ілюструє здатність моделі розрізняти два класи (наприклад, пацієнти з діабетом і без нього) при зміні порогових значень для передбачених ймовірностей. Червона пунктирна лінія відповідає випадковим передбаченням, де модель не має предиктивної сили ($AUC = 0.5$). Чим далі ROC-крива розташована від цієї лінії, тим кращою є продуктивність моделі (рис. 3.1).

AUC (Area Under Curve) у вашому випадку становить 0.96, що є дуже високим показником. Це означає, що модель має високу здатність розрізняти позитивні та негативні класи. Наприклад, у 96% випадків модель правильно визначає, чи належить об'єкт до класу 1 чи 0. Ліва частина кривої, де FPR є низьким, демонструє, що модель має високу чутливість із мінімальною кількістю помилкових позитивних передбачень. У правій частині кривої, де FPR вищий, модель збільшує чутливість за рахунок зниження специфічності. Це може бути прийнятним у задачах, де критично важливе виявлення позитивних випадків, наприклад, у медичній діагностиці. Крутий підйом кривої свідчить про те, що модель досягає високої точності навіть при низьких FPR .

Значення $AUC = 0.96$ та вигляд кривої свідчать про те, що модель логістичної регресії дуже добре виконує класифікацію і може бути ефективно використана для завдання передбачення діабету.

Логістична регресія є простою у реалізації та ефективною для задач бінарної класифікації, таких як прогнозування захворювань, аналіз кредитного ризику чи визначення ймовірності відтоку клієнтів. Вона добре працює, коли залежність між змінними є лінійною, а також забезпечує інтерпретацію результатів через ваги моделі, що дає можливість оцінювати вплив кожної ознаки на результат. Її перевагами є інтерпретованість результатів, низькі вимоги до обчислювальних ресурсів та можливість оцінки важливості ознак. Однак модель може бути менш ефективною при роботі з великими та складними наборами даних, де існує нелінійна залежність між ознаками та цільовою змінною. У таких випадках доцільно використовувати більш складні алгоритми, наприклад, градієнтний бустинг чи нейронні мережі.

3.2 Створення моделі машинного навчання на основі методу опорних векторів (SVM)

Метод опорних векторів (SVM) є потужним алгоритмом машинного навчання, який використовується для класифікації та регресії. Його основна ідея полягає у побудові гіперплощини, яка максимально відокремлює різні класи даних у багатовимірному просторі. Головна мета алгоритму — знайти таку гіперплощину, яка має найбільшу відстань до найближчих точок з кожного класу, забезпечуючи кращу узагальненість моделі [45] [46]. Наприклад, для двовимірного простору це буде лінія, яка розділяє точки двох класів із максимальним зазором. Опорні вектори — це точки з обох класів, що знаходяться найближче до гіперплощини. Ці точки впливають на положення гіперплощини, тому будь-яка зміна в них змінює конфігурацію моделі. Для складних задач, коли класи не є лінійно роздільними, використовується "ядровий трюк", що дозволяє проєктувати дані у простір вищої розмірності для побудови більш ефективної гіперплощини. В задачах класифікації SVM ефективно виконує як лінійну, так і нелінійну класифікацію завдяки використанню ядрового трюку.

Гіперплощина у SVM визначається рівнянням:

$$w \cdot x + b = 0 \quad (3.3)$$

де w — вектор ваг, x — вектор ознак, а b — зсув [3]. Опорні вектори є точками, найближчими до гіперплощини, які визначають її положення. Завдяки своїй структурі, SVM ідеально підходить для задач, де дані добре розділяються [45].

Метод опорних векторів базується на кількох важливих компонентах. Одним із ключових аспектів є вибір ядра, яке визначає, як модель працюватиме з даними. Наприклад, лінійне ядро підходить для лінійно роздільних даних, тоді як радіально-базисна функція (RBF) дозволяє враховувати складні нелінійні взаємозв'язки.

Навчання SVM включає такі етапи:

- Підготовка даних, включаючи масштабування та обробку пропусків, є критично важливим етапом. Масштабування, зокрема, допомагає вирівняти

вплив різних ознак, які можуть мати різні одиниці вимірювання або величини. Наприклад, якщо одна ознака вимірюється в тисячах, а інша — в одиницях, це може вплинути на визначення оптимальної гіперплощини. Масштабування числових ознак до однакового діапазону дозволяє уникнути цього перекосу, забезпечуючи стабільну роботу алгоритму. Пропуски в даних також необхідно обробляти, оскільки вони можуть викликати помилки або спотворювати результати моделі.

- Вибір типу ядра та його параметрів, залежить від характеру даних та складності завдання. Лінійне ядро підходить для задач, де дані є лінійно роздільними, дозволяючи моделі створювати просту гіперплощину для класифікації. Радіально-базисне ядро (RBF) використовується для роботи з нелінійними даними, оскільки воно ефективно проєктує точки у вищий вимір для кращого розділення. Поліноміальне ядро застосовується, коли зв'язки між класами є поліноміальними. Вибір параметрів ядра, таких як коефіцієнт C або γ , впливає на узагальнюючу здатність моделі, і його здійснюють шляхом крос-валідації.
- Оптимізація функції втрат для визначення параметрів гіперплощини використовується для мінімізації помилок класифікації та забезпечення максимального зазору між класами. Це досягається за допомогою квадратичного програмування, яке дозволяє ефективно обчислювати оптимальні параметри моделі, такі як вектор ваг та зсув. Зокрема, для задач із нелінійними кордонами квадратичне програмування допомагає враховувати вплив ядрових перетворень, що покращує здатність моделі до узагальнення.
- Перевірка моделі на тестових даних є важливою для оцінки здатності моделі узагальнювати нові дані. Для цього обчислюються ключові метрики продуктивності, такі як точність, F1-міра, ROC-AUC та інші, які дають змогу оцінити, наскільки ефективно модель відокремлює класи або прогнозує результат. Зокрема, ROC-AUC дозволяє оцінити здатність моделі балансувати

між чутливістю та специфічністю, що є критично важливим у медичних задачах, де помилкові прогнози можуть мати серйозні наслідки.

Підготовка даних

Перед початком навчання необхідно виконати обробку даних. Це включає в себе видалення пропусків, масштабування числових змінних та кодування категоріальних ознак. Наприклад, у наборі даних "diabetes_prediction_dataset.csv" виконувались наступні кроки:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Завантаження даних
data = pd.read_csv('diabetes_prediction_dataset.csv')

# Перевірка пропущених значень
print("Пропущені значення в кожній колонці:")
print(data.isnull().sum())

# Кодування категоріальних ознак
label_encoders = {}
categorical_columns = ['gender', 'smoking_history']
for col in categorical_columns:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col])
    label_encoders[col] = le

# Масштабування числових даних
scaler = StandardScaler()
numerical_columns = ['age', 'bmi', 'HbA1c_level', 'blood_glucose_level']
data[numerical_columns] = scaler.fit_transform(data[numerical_columns])
```

Навчання моделі

Після обробки даних SVM було навчено за допомогою ядра RBF. Цей тип ядра ефективний для роботи з нелінійно роздільними даними.

```
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

# Розділення даних на тренувальні та тестові
X = data.drop('diabetes', axis=1)
y = data['diabetes']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

```
# Навчання SVM
model = SVC(probability=True, kernel='rbf', C=1.0, random_state=42)
model.fit(X_train, y_train)
```

Оцінка моделі

Для оцінки продуктивності моделі використовувалися метрики, такі як класифікаційний звіт, матриця плутанини та ROC-AUC.

```
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score,
roc_curve
import matplotlib.pyplot as plt

# Оцінка моделі
y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]

print("Класифікаційний звіт:")
print(classification_report(y_test, y_pred))

print("Матриця плутанини:")
print(confusion_matrix(y_test, y_pred))

roc_auc = roc_auc_score(y_test, y_proba)
print(f"ROC-AUC: {roc_auc}")

# Побудова ROC-кривої
fpr, tpr, _ = roc_curve(y_test, y_proba)
plt.figure()
plt.plot(fpr, tpr, label=f"ROC Curve (AUC = {roc_auc:.2f})")
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()
```

Класифікаційний звіт:

	precision	recall	f1-score	support
0	0.96	1.00	0.98	18292
1	0.99	0.56	0.72	1708
accuracy			0.96	20000
macro avg	0.98	0.78	0.85	20000
weighted avg	0.96	0.96	0.96	20000

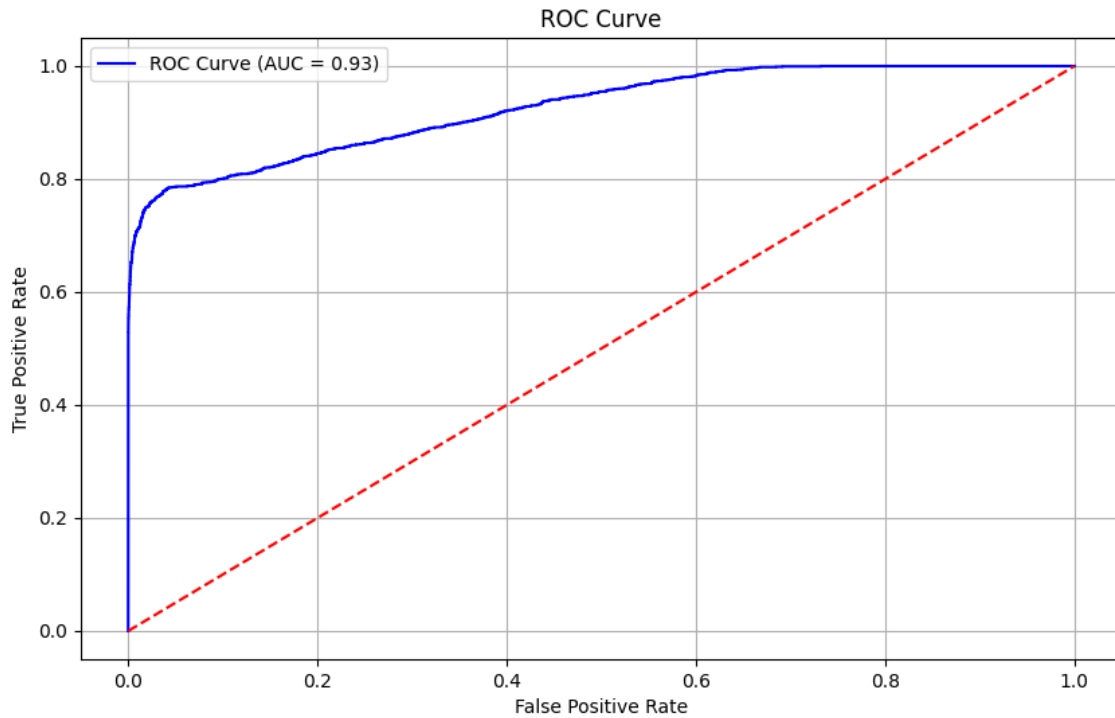


Рис. 3.2 ROC крива навчання моделі по методу опорних векторів (SVM)

Площа під кривою (AUC - Area Under Curve) (рис. 3.2) дорівнює 0.93, що є показником високої якості класифікації. AUC, близьке до 1, вказує на те, що модель здатна добре відрізняти позитивні зразки від негативних. Велика площа під кривою підтверджує, що модель ефективно балансує між чутливістю та специфічністю, але поступається моделі навченої за методом логістичної регресії (рис. 3.1).

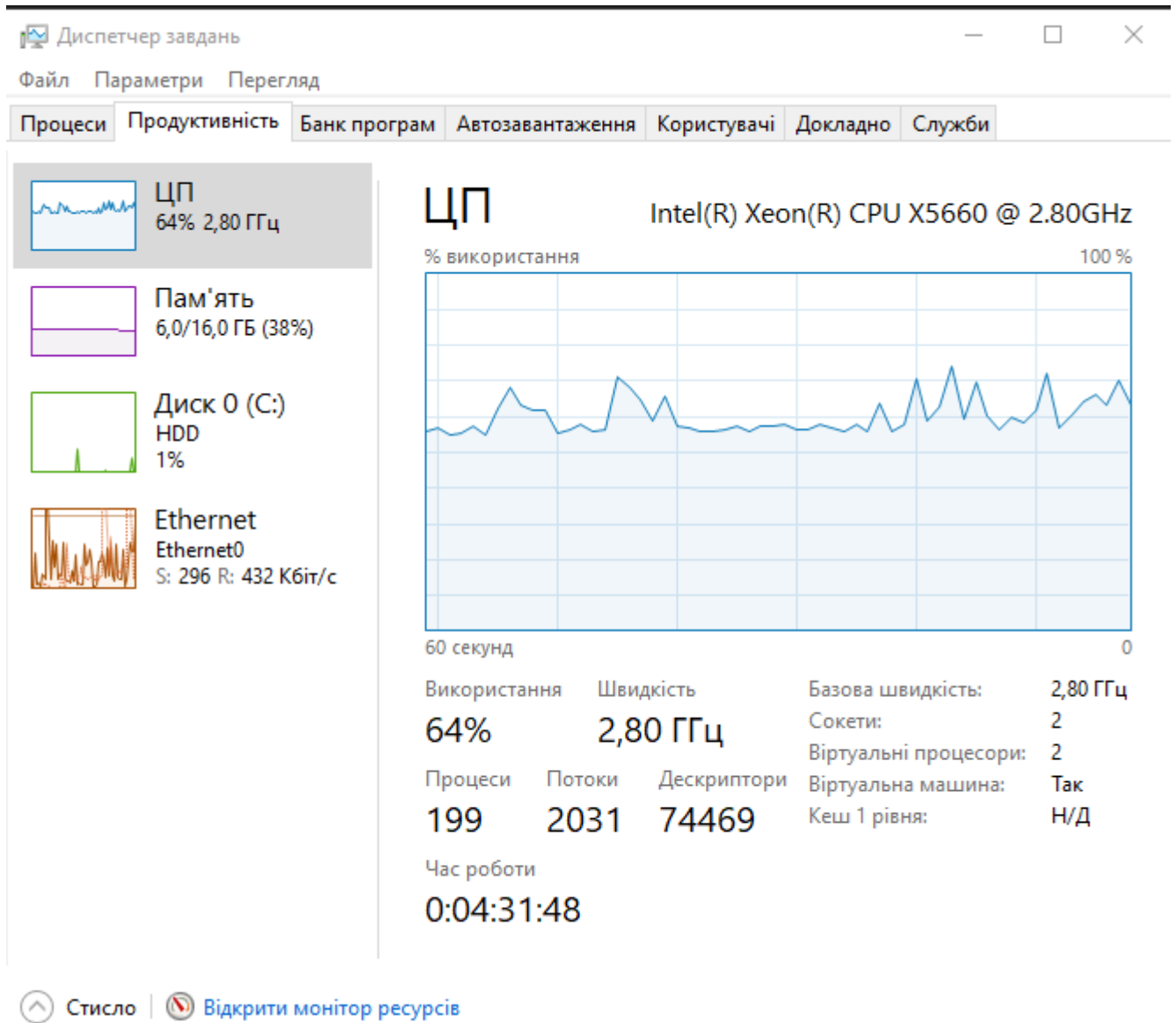


Рис. 3.3 Завантаження центрального процесора під час навчання моделі по методу опорних векторів (SVM)

Метод опорних векторів забезпечує високу точність і здатність узагальнювати дані навіть для складних задач класифікації. Наприклад, у задачі розпізнавання рукописних цифр із набору даних MNIST SVM демонструє високу точність завдяки здатності ефективно працювати з нелінійними кордонами. У цій задачі радіально-базисне ядро дозволяє моделі навчитися складним патернам, недоступним для простіших алгоритмів, таких як лінійна регресія. Його основними перевагами є можливість роботи з лінійно та нелінійно розділними даними, висока стабільність моделі та точність. Однак недоліками можуть бути висока обчислювальна складність при роботі з великими наборами даних (процес навчання на даному об'ємі

даних тривав близько 15хв) та складність вибору оптимальних параметрів ядра (рис. 3.3).

3.3 Створення моделі машинного навчання на основі методу градієнтного бустингу (XGBoost)

XGBoost (eXtreme Gradient Boosting) є однією з найпотужніших і найпопулярніших бібліотек для машинного навчання, яка реалізує метод градієнтного бустингу. Цей підхід базується на ансамблі дерев рішень, які поетапно зменшують похибку моделі. Основною ідеєю XGBoost є поступове покращення точності прогнозів через врахування помилок попередніх ітерацій. XGBoost забезпечує високу продуктивність, точність і швидкість навчання, що робить його вибором номер один для багатьох прикладних задач у сфері машинного навчання. Завдяки гнучкості у налаштуванні параметрів та підтримці великих наборів даних XGBoost широко використовується в класифікації, регресії та задачах ранжування.

Метод градієнтного бустингу полягає у створенні ансамблю моделей, де кожна нова модель коригує помилки попередніх. Початковий етап складається зі створення базової моделі, яка дає первинні прогнози. Далі кожне наступне дерево рішень навчається на залишках (помилках) попередньої моделі. Цей процес продовжується доти, доки модель не досягне бажаного рівня точності або не виконає максимальну кількість ітерацій. На фінальному етапі всі прогнози об'єднуються для створення остаточного результату. XGBoost включає в себе спеціальні оптимізації, такі як паралельне навчання дерев, регуляризація та кешування, що значно підвищують ефективність алгоритму [47] [48].

XGBoost має декілька ключових характеристик, які виділяють його серед інших методів машинного навчання. Паралелізація процесу навчання дерев дозволяє значно скоротити час розрахунків, що особливо важливо для великих наборів даних [47]. Регуляризація, що включає L1 (LASSO) та L2 (Ridge), допомагає знизити ризик перенавчання, забезпечуючи високу здатність моделі до узагальнення [47] [48]. Оптимізація, в тому числі використання спеціалізованих алгоритмів обробки даних,

дозволяє XGBoost працювати ефективно навіть з великими та складними наборами даних. Завдяки своїй архітектурі бібліотека підтримує роботу з пропусками в даних, що є важливим аспектом для реальних задач.

У порівнянні з іншими алгоритмами градієнтного бустингу, такими як CatBoost та LightGBM, XGBoost має свої переваги. Наприклад, XGBoost демонструє високу точність у задачах бінарної класифікації завдяки оптимізації процесу навчання [47]. У свою чергу, CatBoost краще підходить для роботи з категоріальними даними завдяки вбудованим методам їхньої обробки [47] [48]. LightGBM забезпечує надзвичайно високу швидкість навчання, що робить його зручним для задач із великими наборами даних. Проте XGBoost залишається універсальним вибором завдяки своїй стабільності та гнучкості.

XGBoost широко використовується у різних галузях, включаючи фінансову аналітику, медичну діагностику, маркетинг та прогнозування. Наприклад, у медичних дослідженнях XGBoost застосовується для прогнозування захворювань на основі широкого спектра факторів, таких як аналіз історії хвороб, біомаркери та способу життя пацієнтів. Його висока точність дозволяє ефективно вирішувати завдання, де точність моделі є критичною.

Підготовка даних

Перед навчанням моделі XGBoost необхідно виконати попередню обробку даних. Це включає видалення пропущених значень, кодування категоріальних змінних та масштабування числових змінних. Наприклад, у наборі даних для прогнозування цукрового діабету виконуються такі кроки:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from xgboost import XGBClassifier

# Завантаження даних
data = pd.read_csv('diabetes_prediction_dataset.csv')

# Кодування категоріальних змінних
```

```

label_encoders = {}
categorical_columns = ['gender', 'smoking_history']
for col in categorical_columns:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col])
    label_encoders[col] = le

# Розділення даних на тренувальний та тестовий набори
X = data.drop('diabetes', axis=1)
y = data['diabetes']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

```

Навчання моделі

Навчання моделі XGBoost виконується із застосуванням оптимальних параметрів, які можна налаштувати за допомогою крос-валідації. В цьому прикладі використовується бінарна класифікація для прогнозування цукрового діабету:

```

# Навчання моделі XGBoost
model = XGBClassifier(n_estimators=100, learning_rate=0.1, max_depth=3,
random_state=42)
model.fit(X_train, y_train)

```

Оцінка моделі

Оцінка продуктивності моделі виконується за допомогою метрик точності, класифікаційного звіту та ROC-AUC:

```

from sklearn.metrics import classification_report, roc_auc_score, roc_curve
import matplotlib.pyplot as plt

# Прогнози
y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]

# Класифікаційний звіт
print(classification_report(y_test, y_pred))

# ROC-AUC
roc_auc = roc_auc_score(y_test, y_proba)
print(f"ROC-AUC: {roc_auc}")

# Побудова ROC-кривої
fpr, tpr, _ = roc_curve(y_test, y_proba)

```

```
plt.figure()
plt.plot(fpr, tpr, label=f"ROC Curve (AUC = {roc_auc:.2f})")
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.show()
```

Класифікаційний звіт:

	precision	recall	f1-score	support
0	0.97	1.00	0.98	18292
1	0.95	0.69	0.80	1708
accuracy			0.97	20000
macro avg	0.96	0.85	0.89	20000
weighted avg	0.97	0.97	0.97	20000

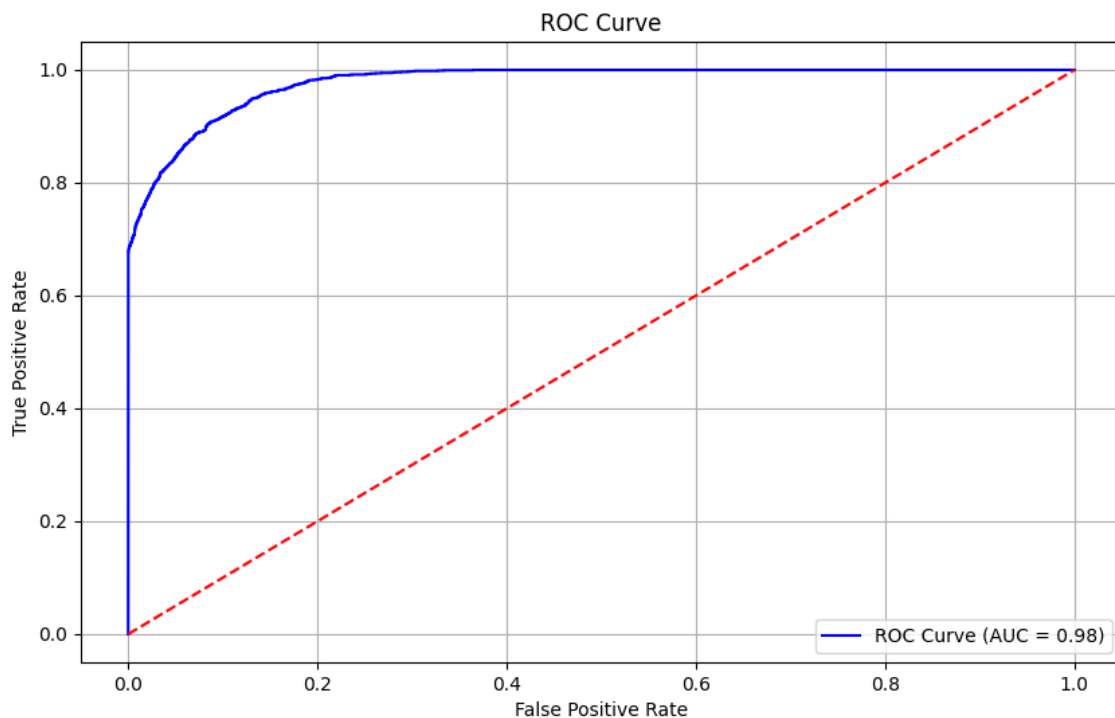


Рис. 3.4 ROC крива навчання моделі по методу градієнтного бустингу (XGBoost)

На зображенні представлена ROC-крива (Receiver Operating Characteristic curve) для моделі, створеної методом градієнтного бустингу (XGBoost). ROC-крива демонструє здатність моделі розрізняти класи при різних порогах класифікації, де

показник True Positive Rate (чутливість) відкладається по осі Y, а False Positive Rate (співвідношення помилково позитивних результатів) — по осі X (рис. 3.4).

Висока вигнутість кривої до верхнього лівого кута вказує на високий рівень точності моделі. Значення площі під кривою ($AUC = 0.98$) свідчить про виняткову здатність моделі класифікувати дані. AUC значення, близьке до 1, означає, що модель може ефективно розрізняти позитивні та негативні класи.

Червона пунктирна лінія на графіку представляє випадкове передбачення ($AUC = 0.5$). Модель XGBoost значно перевищує цю лінію, що демонструє її перевагу над випадковими класифікаторами. Високе значення AUC також свідчить про те, що модель добре збалансована між чутливістю та специфічністю, забезпечуючи високу якість класифікації навіть у випадку нерівномірного розподілу класів у даних.

XGBoost є універсальним і потужним алгоритмом машинного навчання, який забезпечує високу точність прогнозів навіть для складних задач. Основними перевагами є паралелізація, регуляризація та здатність працювати з великими наборами даних. Недоліком є потенційна складність налаштування параметрів та час навчання для великих моделей. Проте XGBoost залишається популярним вибором завдяки своїй ефективності та точності у різноманітних задачах.

3.4 Створення моделі машинного навчання на основі методу k найближчих сусідів (K-NN)

Метод k-найближчих сусідів (K-NN) є одним із найпростіших і водночас потужних алгоритмів машинного навчання, що застосовується для задач класифікації та регресії. Основна ідея цього алгоритму базується на припущенні, що об'єкти з подібними ознаками мають схожі класи або значення. K-NN належить до категорії алгоритмів контрольованого навчання і є непараметричним методом, тобто не вимагає попередніх припущень про розподіл даних.

Метод K-NN працює шляхом пошуку найближчих сусідів для кожного нового зразка даних у навчальному просторі. Першим етапом є вибір значення параметра k , який визначає кількість найближчих сусідів, що враховуються під час класифікації. Після цього обчислюються відстані між новим зразком і кожною точкою навчального набору. Найпоширенішою метрикою для обчислення відстаней є евклідова відстань (3.4), хоча також можуть застосовуватися манхеттенська, косинусна подібність чи відстань Махаланобіса (3.5) [1][5].

Евклідова відстань обчислюється за формулою:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (3.4)$$

де x та y — це точки в n -вимірному просторі, а x_i та y_i — координати відповідних точок у вимірі i .

Відстань Махаланобіса враховує кореляцію між змінними та обчислюється за формулою:

$$d_M(x, y) = \sqrt{(x - y)^T S^{-1} (x - y)} \quad (3.5)$$

де S — коваріаційна матриця, x та y — вектори точок, а T позначає транспонування. Ця метрика корисна для даних з корельованими ознаками.

Коли найближчі сусіди визначено, алгоритм класифікує нову точку на основі більшості голосів серед цих сусідів. Наприклад, якщо більшість сусідів належить до класу A , то і новий зразок буде віднесено до цього класу. У випадку задачі регресії результатом є середнє значення цільової змінної серед найближчих сусідів [51] [52].

Однією з головних переваг K-NN є його простота реалізації та інтуїтивно зрозуміла логіка роботи. Цей метод також є гнучким і може бути використаний для різних типів задач, таких як класифікація чи регресія. Його непараметричний характер дозволяє працювати з даними без припущень щодо їхнього розподілу, що є значною перевагою в багатьох реальних сценаріях [51].

Однак K-NN має і свої недоліки. Перш за все, це висока обчислювальна складність, оскільки для кожної нової точки необхідно обчислювати відстань до всіх зразків у навчальному наборі. Крім того, метод є чутливим до шуму в даних, що може призводити до неправильних класифікацій. Також вибір оптимального значення k може бути складним завданням, оскільки занадто мале значення k робить модель чутливою до шуму, а занадто велике може призвести до переузагальнення [53] [52].

Ефективність методу K-NN залежить від якісної підготовки даних. Перш за все, необхідно забезпечити відсутність пропусків у даних, що може бути досягнуто шляхом заповнення пропущених значень середнім, медіанним або модою. Нормалізація або стандартизація даних є критично важливими, оскільки K-NN використовує метрики відстані, які можуть бути викривлені через різні масштаби змінних [52]. Наприклад, якщо одна змінна має значення в діапазоні від 0 до 1000, а інша — від 0 до 1, остання матиме значно менший вплив на обчислення відстаней. Нормалізація дозволяє привести всі змінні до єдиного масштабу, наприклад, у діапазон $[0, 1]$, що забезпечує рівномірний внесок кожної ознаки. У експериментах було показано, що застосування нормалізації значно підвищує точність класифікації, особливо у наборах даних із великою варіативністю величин ознак.

Процес підготовки даних починається з завантаження набору даних. У цьому проекті було використано датасет для прогнозування діабету. Спершу перевіряються пропуски в даних, оскільки їхня наявність може негативно вплинути на якість моделі. Пропуски усуваються шляхом заповнення середніми або модальними значеннями. Наявність пропусків визначаємо наступним чином:

```
print("Пропущені значення в кожній колонці:")  
print(data.isnull().sum())
```

Категоріальні змінні, такі як "gender" та "smoking_history", кодуються за допомогою LabelEncoder для перетворення текстових значень у числові. Числові ознаки, включаючи "age", "bmi", "HbA1c_level", "blood_glucose_level", масштабуються за допомогою StandardScaler. Масштабування забезпечує однаковий внесок ознак у процес обчислення відстаней між точками.

```
scaler = StandardScaler()
numerical_columns = ['age', 'bmi', 'HbA1c_level', 'blood_glucose_level']
data[numerical_columns] = scaler.fit_transform(data[numerical_columns])
```

Дані розділяються на ознаки (X) та цільову змінну (y), а потім розділяються на тренувальний та тестовий набори у співвідношенні 80:20 для забезпечення збалансованої оцінки моделі.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

Крім того, попередня обробка даних може включати видалення викидів, які можуть суттєво вплинути на точність моделі. Для цього можуть бути використані методи, такі як Z-оцінка або міжквартильний діапазон [5][6].

Навчання моделі

Навчання моделі K-NN полягає у збереженні всіх даних навчального набору, оскільки сам алгоритм не будує явної моделі, а використовує всі точки для класифікації нових даних. Для оцінки моделі часто використовуються методи перехресної валідації, що дозволяє знайти оптимальне значення k . Перехресна валідація допомагає розподілити дані на навчальні та тестові набори декількома способами, що мінімізує вплив вибору конкретного тестового набору. Наприклад, вона дає можливість оцінити, яке значення k забезпечує найкращу точність моделі. Це особливо важливо у випадках, коли дані містять неоднорідні або потенційно шумні зразки, оскільки вибір неправильного k може суттєво вплинути на якість прогнозів. Для навчання використовується клас KNeighborsClassifier з параметром k , який визначає кількість сусідів для голосування. У цьому прикладі значення k встановлено рівним 5.

```
model = KNeighborsClassifier(n_neighbors=5)
model.fit(X_train, y_train)
```

Оцінка моделі

Оцінка моделі включає прогнозування класів для тестових даних та обчислення метрик, таких як точність, F1-міра, а також побудову матриці плутанини. Для оцінки якості класифікації також використовується ROC-AUC. [3][4].

```
y_pred = model.predict(X_test)
y_proba = model.predict_proba(X_test)[:, 1]
print(classification_report(y_test, y_pred))
print(confusion_matrix(y_test, y_pred))
roc_auc = roc_auc_score(y_test, y_proba)
print(f"ROC-AUC: {roc_auc}")
```

Побудова графіку ROC-кривої ілюструє співвідношення між рівнем помилкових спрацьовувань і справжніми позитивними спрацьовуваннями при різних порогах класифікації.

```
fpr, tpr, _ = roc_curve(y_test, y_proba)
plt.plot(fpr, tpr, color='blue', label=f"ROC Curve (AUC = {roc_auc:.2f})")
plt.plot([0, 1], [0, 1], color='red', linestyle='--')
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("ROC Curve")
plt.legend()
plt.grid()
plt.show()
```

Навчена модель та об'єкт масштабування зберігаються у файли для подальшого використання без необхідності повторного навчання.

```
dump(model, 'diabetes_knn_model.joblib')
dump(scaler, 'diabetes_knn_scaler.joblib')
```

Класифікаційний звіт:

	precision	recall	f1-score	support
0	0.96	0.99	0.98	18292
1	0.90	0.61	0.73	1708
accuracy			0.96	20000
macro avg	0.93	0.80	0.85	20000
weighted avg	0.96	0.96	0.96	20000

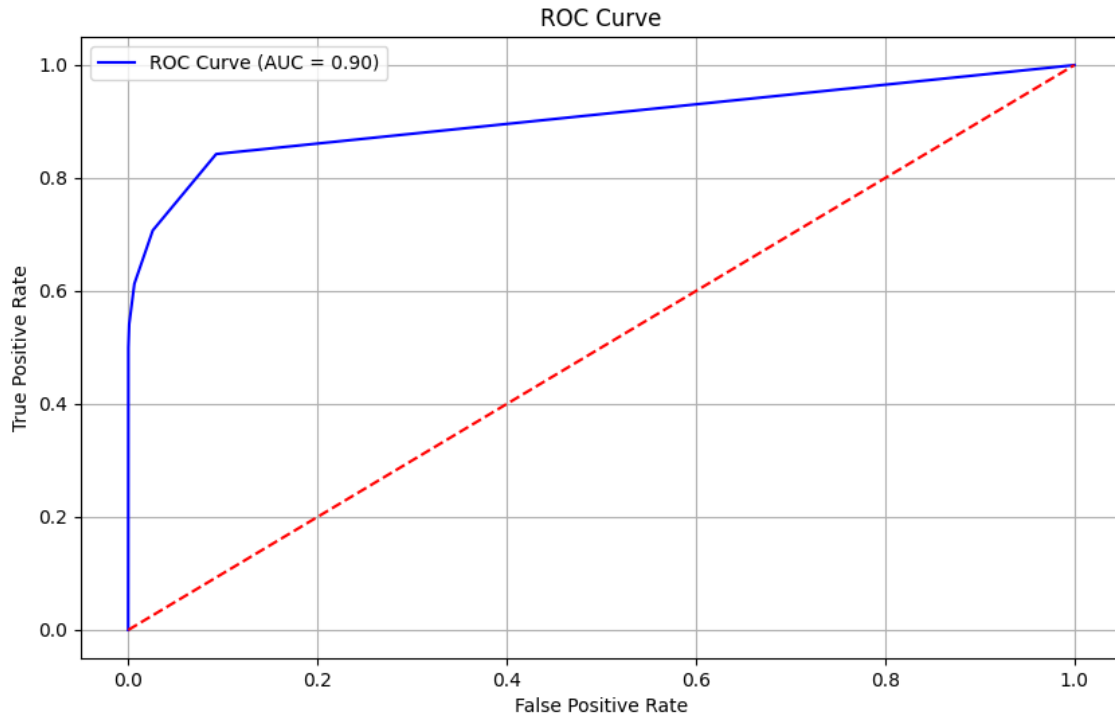


Рис. 3.5 ROC крива навчання моделі по методу k найближчих сусідів (K-NN)

На зображенні 3.5 представлено ROC-криву, побудовану для моделі K-найближчих сусідів (K-NN). ROC-крива (Receiver Operating Characteristic) ілюструє співвідношення між показником True Positive Rate (справжня позитивна частота) та False Positive Rate (хибна позитивна частота) при різних порогових значеннях.

AUC (Area Under the Curve) — площа під ROC-кривою — є числовим показником якості моделі. У даному випадку значення AUC дорівнює 0.90, що є дуже хорошим результатом. Це свідчить про те, що модель K-NN має високу здатність відрізнати позитивні приклади від негативних.

На початку кривої можна помітити стрімке зростання True Positive Rate при низьких значеннях False Positive Rate. Це означає, що модель здатна ефективно класифікувати більшість позитивних випадків без значного збільшення помилкових спрацьовувань. Однак у міру наближення до правої частини графіка зростає й False Positive Rate, що вказує на компроміс між чутливістю та специфічністю.

Таким чином, ROC-крива та $AUC = 0.90$ свідчать про те, що метод K-NN продемонстрував гарну продуктивність на даних, забезпечуючи точну класифікацію

більшості зразків. Хоча модель демонструє хорошу продуктивність, її ROC крива не така крута, як у логістичної регресії чи XGBoost. Це вказує на те, що K-NN менш ефективно розрізняє позитивні і негативні класи, особливо при низьких значеннях False Positive Rate. K-NN є більш чутливим до вибору параметра k та до структури даних, що може пояснити його нижчу продуктивність у порівнянні з іншими моделями.

Таким чином, XGBoost показує найвищу ефективність серед представлених методів з найвищим значенням AUC, що робить його найбільш підходящим для задачі. Логістична регресія також демонструє чудові результати, трохи поступаючись XGBoost. Натомість K-NN хоч і є конкурентоспроможною моделлю, але поступається іншим двом через нижчу точність і чутливість до особливостей даних.

Метод k -найближчих сусідів є потужним і універсальним інструментом у машинному навчанні, який відзначається своєю простотою та інтуїтивністю. Однак його застосування вимагає ретельної підготовки даних і вибору параметра k . Правильна обробка даних і налаштування алгоритму можуть значно підвищити ефективність цього методу. Незважаючи на свої обмеження, K-NN залишається популярним вибором для багатьох задач завдяки своїй гнучкості та здатності адаптуватися до різних типів даних.

Порівняльний аналіз моделей

Для прогнозування наявності цукрового діабету було розглянуто чотири моделі машинного навчання: логістичну регресію, метод опорних векторів (SVM), градієнтний бустинг (XGBoost) та метод k найближчих сусідів (K-NN). Кожна з цих моделей має свої особливості, переваги та недоліки, що впливають на їхню продуктивність у конкретній задачі.

Логістична регресія є базовим і часто використовуваним алгоритмом для задач бінарної класифікації. Основна перевага логістичної регресії полягає у її

інтерпретованості та відносно низьких обчислювальних витратах. Модель є ефективною, коли між ознаками та цільовою змінною існує лінійний зв'язок. У даній роботі логістична регресія продемонструвала високу продуктивність з показником $AUC = 0.96$, що свідчить про її здатність успішно розрізняти позитивні та негативні випадки.

Метод опорних векторів (SVM) є потужним інструментом для задач класифікації, особливо у випадках, коли дані є лінійно або нелінійно роздільними. Використання ядра RBF дозволяє проєктувати дані у простір більшої розмірності для досягнення кращого розділення класів. Проте SVM є обчислювально витратним, особливо для великих наборів даних. При використанні різних типів ядер, зокрема RBF та поліноміального ядра, час обчислення може значно зростати, що робить цей алгоритм менш ефективним для масштабних задач. У цій задачі модель SVM показала хороший результат із $AUC = 0.93$, але поступилася логістичній регресії та XGBoost через нижчу ефективність при обробці великих обсягів даних.

Гرادієнтний бустинг (XGBoost) є однією з найефективніших моделей машинного навчання завдяки своїй здатності працювати з нелінійними зв'язками та великими обсягами даних. XGBoost використовує ансамбль дерев рішень, що дозволяє йому зменшувати помилки на кожній ітерації навчання. У даній роботі XGBoost досяг найвищого значення $AUC = 0.98$, що підтверджує його високу точність та здатність до узагальнення на нових даних. Паралелізація та регуляризація роблять XGBoost оптимальним вибором для задач із великими наборами даних.

Метод k найближчих сусідів (K-NN) є простим та інтуїтивно зрозумілим алгоритмом. Його головна ідея полягає у порівнянні нових точок із найближчими сусідами у навчальному наборі. Однак K-NN є обчислювально інтенсивним, оскільки потребує збереження всіх навчальних даних і обчислення відстаней для кожної нової точки. Це обмеження впливає на продуктивність алгоритму, особливо для великих наборів даних, де час обробки зростає експоненційно зі збільшенням кількості точок. Крім того, ресурсомісткість обчислення відстаней може створювати

труднощі при використанні K-NN у задачах із реального часу, що обмежує його застосування в практичних сценаріях. Крім того, K-NN є чутливим до вибору параметра k і наявності шуму у даних. У цій задачі модель K-NN показала $AUC = 0.90$, що є хорошим результатом, але нижчим у порівнянні з XGBoost та логістичною регресією.

На основі отриманих результатів для задачі прогнозування цукрового діабету було обрано дві найефективніші моделі: **градієнтний бустинг (XGBoost)** та **логістичну регресію**.

XGBoost продемонстрував найвищий показник $AUC = 0.98$, що вказує на його відмінну здатність до класифікації навіть у випадках складних нелінійних взаємозв'язків. Висока ресурсомісткість XGBoost обумовлена обчислювальними витратами на побудову ансамблю дерев рішень і регуляризацію. Це може обмежувати його застосування у задачах з обмеженими апаратними ресурсами, але водночас робить його надзвичайно ефективним для роботи з великими наборами даних, де він може оптимально використовувати доступні ресурси. Ця модель є особливо ефективною завдяки своїй гнучкості, регуляризації та можливості роботи з великими наборами даних. У задачах медичної діагностики, де критично важливе високе значення чутливості та специфічності, XGBoost є ідеальним вибором.

Логістична регресія також показала відмінний результат із $AUC = 0.96$. Її простота реалізації, інтерпретованість і невеликі обчислювальні витрати роблять цю модель особливо зручною для медичних досліджень, де важлива прозорість ухвалення рішень. Логістична регресія є надійним методом для завдань, у яких дані є добре підготовленими і демонструють лінійний зв'язок між ознаками.

Таким чином, поєднання XGBoost та логістичної регресії забезпечує оптимальний баланс між продуктивністю, точністю та інтерпретованістю. Вибір між цими моделями залежить також від ресурсоємності обчислень і обсягів даних. XGBoost є більш ефективним для великих наборів даних завдяки своїй здатності до паралелізації та роботи зі складними нелінійними зв'язками, тоді як логістична регресія є менш вимогливою до обчислювальних ресурсів і чудово підходить для

менших та добре підготовлених наборів даних. Це забезпечує оптимальні результати для ефективного прогнозування цукрового діабету.

4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Архітектура API на Flask для взаємодії з моделями машинного навчання

Flask є одним із найпопулярніших мікрофреймворків на Python, що активно використовується для розробки RESTful API. Наприклад, компанії Netflix і Lyft використовували Flask для створення частини своїх сервісів, демонструючи його можливості у розробці високонавантажених додатків. Інші проекти, такі як OpenAI, також активно застосовують Flask для інтеграції своїх ML моделей через API. Ці API служать для взаємодії з моделями машинного навчання (ML), забезпечуючи передачу даних між користувачами та моделями, обчислювальними серверами та іншими компонентами системи. Завдяки своїй простоті, гнучкості та масштабованості, Flask залишається одним із найкращих виборів для створення таких архітектур.

Flask побудований за принципом "мінімальної основи" ("micro-framework"), що надає лише необхідний набір функціоналу для створення веб-додатків, залишаючи розробнику свободу вибору додаткових бібліотек чи структур. У контексті роботи з моделями машинного навчання, цей підхід дозволяє налаштовувати архітектуру відповідно до специфічних потреб проекту.

Фреймворк Flask підтримує ключові концепції RESTful архітектури, що є фундаментальними для побудови API. У контексті проектів машинного навчання це означає можливість чітко структурованої передачі даних, наприклад, від користувача до серверної частини, де дані обробляються моделлю, і назад до користувача у вигляді результатів. Як приклад, через HTTP-запити можна передати параметри для моделі прогнозування, яка повертає результат аналізу у форматі JSON. Такий підхід дозволяє інтегрувати ML моделі в існуючі системи чи мобільні додатки з мінімальними зусиллями. Це включає в себе обробку HTTP-запитів (GET, POST, PUT, DELETE) і повернення відповідей у форматі JSON. Останнє є особливо важливим для роботи з ML моделями, оскільки саме через JSON передаються дані для аналізу та отримані результати.

Переваги використання Flask

Flask має ряд переваг, які роблять його привабливим для розробників ML API: у порівнянні з Django, Flask забезпечує більшу гнучкість та меншу складність, що ідеально підходить для невеликих або спеціалізованих ML-проектів. Django, натомість, надає багато вбудованих функцій, таких як ORM і система адміністрування, що може бути перевагою для масштабних проектів. FastAPI, як сучасний конкурент Flask, пропонує нативну підтримку асинхронності та автоматичну генерацію документації API, що робить його кращим вибором для високонавантажених систем. Також цей фреймворк має такі переваги:

- Завдяки мінімалістичному дизайну Flask, розробники можуть швидко створювати прототипи або мінімально життєздатні продукти (MVP). Це значно знижує вхідний бар'єр для початківців та пришвидшує процес розробки [54].
- Flask дозволяє налаштовувати структуру проекту залежно від потреб, не нав'язуючи строгих стандартів. Це особливо важливо для ML проектів, які можуть мати різноманітну архітектуру [54].
- Існує багато додаткових бібліотек і розширень, які інтегруються з Flask, наприклад, Flask-SQLAlchemy для роботи з базами даних або Flask-JWT для аутентифікації.
- Jinja2 спрощує рендеринг шаблонів і генерацію динамічних HTML-сторінок, дозволяючи ефективно візуалізувати результати роботи ML моделей [54].
- Flask ідеально підходить для побудови RESTful API, що дозволяє ефективно взаємодіяти з ML моделями через HTTP-запити [55].

Виклики та недоліки

Незважаючи на численні переваги, Flask має й певні недоліки, які потрібно враховувати.

Одним із найбільших викликів є відсутність вбудованої асинхронності. Flask не підтримує асинхронні запити "з коробки", що може створювати проблеми при роботі з довготривалими операціями. Для вирішення цієї проблеми використовуються сторонні інструменти, такі як Celery, який дозволяє виконувати задачі у фоновому режимі. Celery інтегрується з Flask через спеціальні розширення, такі як Flask-Celery, що полегшує передачу довготривалих обчислень до черг завдань. Наприклад, запити до моделей машинного навчання можуть бути поставлені в чергу, а клієнт отримує відповідь про те, що запит обробляється, з можливістю отримати результат пізніше [55].

Ще одним обмеженням є продуктивність для масштабованих проєктів. Flask добре підходить для малих та середніх проєктів, але його продуктивність може бути недостатньою для великих систем із високими вимогами до навантаження. У таких випадках доцільно використовувати альтернативні рішення або оптимізувати інфраструктуру.

Також слід враховувати, що Flask пропонує менше вбудованих функцій у порівнянні з комплексними фреймворками, такими як Django. Відсутність ORM або вбудованих систем адміністрування може вимагати більше часу на налаштування та інтеграцію сторонніх бібліотек [54].

Окремо слід зазначити питання безпеки. Розробники повинні самостійно додавати механізми захисту, такі як захист від CSRF-атак або впровадження політик CORS, оскільки Flask не має вбудованих засобів для забезпечення безпеки [54].

Отже Flask є ефективним інструментом для створення API, які взаємодіють із моделями машинного навчання. Він спрощує передачу даних до моделей і отримання результатів у стандартизованому форматі JSON, що прискорює інтеграцію з іншими системами. Наприклад, Flask дозволяє легко налаштувати маршрути для подачі вхідних даних до моделей через POST-запити або отримання передбачень через GET-запити, оптимізуючи таким чином процеси обчислення і доступу до результатів аналізу. Його простота, гнучкість і розширюваність роблять його ідеальним вибором для швидкої розробки прототипів та реалізації невеликих

проектів. Водночас для великих проектів чи систем із високими вимогами до продуктивності варто враховувати його обмеження і заздалегідь планувати відповідні оптимізації чи інтеграцію додаткових інструментів.

Створення API для прогнозування цукрового діабету на Flask

У рамках кваліфікаційної роботи була розроблена серверна частина застосунку для прогнозування наявності цукрового діабету у користувачів. Основним завданням цієї частини було забезпечення доступу до машинних моделей через HTTP-запити, що реалізовано за допомогою веб-фреймворку Flask.

Блок-схема алгоритму роботи додатку та вибору найточнішого передбачення представлена у Додатку В.

Для реалізації проекту були обрані інструменти, що забезпечують простоту, масштабованість та зручність інтеграції з машинним навчанням. Python 3.11 був обраний через його популярність у наукових обчисленнях, розвинену екосистему бібліотек та сумісність з інструментами машинного навчання. Flask забезпечує легкий і гнучкий спосіб створення API, а Flask-CORS дозволяє легко налаштувати доступ між фронтендом і бекендом. Бібліотеки `scikit-learn` та `joblib` використовуються для роботи з моделями машинного навчання, включаючи їх серіалізацію та завантаження. `Pandas` і `NumPy` забезпечують ефективну обробку даних для підготовки до передбачень.

Розробка почалася зі створення базового структурування проекту та підготовки середовища. Для цього використовувалися наступні інструменти:

1. Python версії 3.11.
2. Flask для створення веб-сервісу.
3. Бібліотека Flask-CORS для забезпечення взаємодії з фронтенд-застосунком.
4. Бібліотеки для роботи з машинними моделями: `scikit-learn`, `joblib`.
5. Додаткові залежності: `pandas`, `numpy` для обробки даних.

```

from flask import Flask, request, jsonify
from joblib import load
import os
import pandas as pd
import numpy as np
from sklearn.preprocessing import StandardScaler
from flask_cors import CORS

# Ініціалізація Flask-додатку
app = Flask(__name__)
CORS(app) # Дозвіл CORS для всіх маршрутів

```

Як уже було зазначено у попередньому розділі для передбачення були використані дві моделі машинного навчання: Logistic Regression та XGBoost, які були попередньо навчені та збережені у форматі `.joblib`. Logistic Regression обрана за її простоту, швидкість роботи та інтерпретованість результатів, тоді як XGBoost відома своєю високою точністю, особливо на складних наборах даних. Для кожної моделі також завантажувався відповідний StandardScaler для нормалізації вхідних даних.

```

# Завантаження моделей
modelLR = load(os.path.join('models', 'diabetes_logisticRegression_model.joblib'))
modelXGBoost = load(os.path.join('models', 'diabetes_xgboost_model.joblib'))

# Завантаження скейлерів
scalerLR = load(os.path.join('models', 'diabetes_logisticRegression_scaler.joblib'))
scalerXGBoost = load(os.path.join('models', 'diabetes_xgboost_scaler.joblib'))

```

Основний функціонал сервісу реалізовано через маршрут `/api/predict`, який обробляє POST-запити. Наприклад, JSON-запит може виглядати так:

```

{
  "gender": "Male",
  "age": 45,
  "hypertension": 1,
  "heart_disease": 0,
  "smoking_history": "former",
  "bmi": 28.5,
  "HbA1c_level": 6.1,
  "blood_glucose_level": 140
}

```

Очікується, що вхідні дані будуть надіслані у форматі JSON та міститимуть наступні ключі:

- ``gender``: стать пацієнта (``Male`` або ``Female``).
- ``age``: вік пацієнта.
- ``hypertension``: наявність гіпертонії (0 або 1).
- ``heart_disease``: наявність серцевих захворювань (0 або 1).
- ``smoking_history``: історія паління (наприклад, ``never``, ``former``, ``current``).
- ``bmi``: індекс маси тіла.
- ``HbA1c_level``: рівень HbA1c.
- ``blood_glucose_level``: рівень глюкози у крові.

Код функції, яку фласк викликає при запиті до відповідного маршруту виглядає наступним чином:

```
@app.route('/api/predict', methods=['POST'])
def predict():
    try:
        # Отримання даних із запиту
        data = request.json

        # Ініціалізація результату
        result = {
            "diabetes_prediction_model": "",
            "diabetes_prediction": 0,
            "diabetes_probability": 0.5,
            "msg": "Наші моделі видали суперечливу інформацію, проконсультуйтеся з лікарем!"
        }

        # Передбачення за допомогою кожної моделі
        xgBoostResult = predictByModel(data, modelXGBoost, scalerXGBoost, {"diabetes_prediction_model": "XGBoost"})
        lrResult = predictByModel(data, modelLR, scalerLR, {"diabetes_prediction_model": "Logistic Regression"})

        # Вибір результату моделі з вищою впевненістю
        if abs(0.5 - xgBoostResult["diabetes_probability"]) >= abs(0.5 - lrResult["diabetes_probability"]):
            result = xgBoostResult
        else:
            result = lrResult

        return jsonify(result)
    except Exception as e:
        return jsonify({"error": str(e)}), 500
```

Допоміжна функція «predictByModel» виконує основну обробку вхідних даних, нормалізацію та отримання передбачення від моделі. Для кодування категорійних змінних використовується метод заміни значень через словники, що дозволяє легко адаптувати обробку до нових категорій, якщо вони з'являться.

```
def predictByModel(data, model, scaler, addData=dict()):
    label_encoders = {
        "gender": {"Female": 0, "Male": 1},
        "smoking_history": {"no info": 2, "never": 0, "former": 1, "current": 1, "not
current": 1}
    }

    # Перетворення вхідних даних
    new_patient_encoded = [
        label_encoders["gender"][data["gender"]],
        float(data["age"]),
        int(data["hypertension"]),
        int(data["heart_disease"]),
        label_encoders["smoking_history"][data["smoking_history"].lower()],
        float(data["bmi"]),
        float(data["HbA1c_level"]),
        float(data["blood_glucose_level"])
    ]

    # Масштабування даних
    new_patient_encoded = np.array(new_patient_encoded).reshape(1, -1)
    new_patient_encoded = scaler.transform(new_patient_encoded)

    # Прогнозування
    diabetes_prediction = model.predict(new_patient_encoded)
    diabetes_probability = model.predict_proba(new_patient_encoded)[: , 1]

    # Формування відповіді
    result = {
        "diabetes_prediction_model": addData.get("diabetes_prediction_model", ""),
        "diabetes_prediction": int(diabetes_prediction[0]),
        "diabetes_probability": round(float(diabetes_probability[0]), 2),
        "msg": ""
    }
    return result
```

Додаток запускається в середовищі розробки для локального тестування. У продуктивному середовищі запуск можна здійснювати через WSGI-сервери на зразок Gunicorn.

```
if __name__ == '__main__':  
    app.run(debug=True)
```

Розроблений API забезпечує зручний інтерфейс для доступу до машинних моделей, реалізуючи ключовий функціонал системи прогнозування. Він підтримує обробку вхідних даних, взаємодію з моделями та повертає структуровані результати. Надійність роботи забезпечується обробкою винятків та перевіркою вхідних даних. Для оцінки продуктивності API проводилися тести, зокрема на швидкість обробки запитів та точність передбачень, результати яких підтвердили відповідність вимогам.

4.2 Розробка користувацького інтерфейсу на Vue.js

Vue.js є сучасним JavaScript-фреймворком, що активно використовується у розробці користувацьких інтерфейсів для веб-застосунків. Наприклад, він застосовується у таких відомих проєктах, як Xiaomi, Alibaba та Laravel Nova, що підкреслює його адаптивність та ефективність. Його популярність серед розробників зумовлена рядом унікальних переваг, які дозволяють ефективно створювати інтерактивні, динамічні та масштабовані додатки. У цьому розділі розглянуто основні причини вибору Vue.js для розробки користувацького інтерфейсу у межах цього проєкту.

Однією з ключових особливостей Vue.js є його прогресивна природа, яка дозволяє інтегрувати фреймворк у вже існуючі проєкти без необхідності значного переписування коду. Це забезпечується його модульною структурою, що дозволяє використовувати лише необхідні компоненти, та легким синтаксисом, який спрощує адаптацію фреймворку. Це сприяє поступовій міграції до нових технологій і збереженню існуючої інфраструктури. Vue.js також забезпечує підтримку різних масштабів проєктів, від невеликих інтерактивних компонентів до великих додатків, що робить його універсальним інструментом для веб-розробки [56].

Інтуїтивно зрозумілий синтаксис Vue.js дозволяє розробникам швидко опанувати основи роботи з фреймворком, навіть якщо вони лише початково знайомі з HTML, CSS та JavaScript. Офіційна документація Vue.js, доступна кількома мовами, включно з українською, а також численні навчальні матеріали й інтерактивні курси, сприяють ефективному засвоєнню фреймворку. Простота налаштування, забезпечена інструментами на кшталт Vue CLI, дозволяє зменшити час, необхідний для розгортання нового проекту або прототипу. Реактивний підхід до управління станом даних у поєднанні з можливістю двостороннього зв'язку даних (data binding) значно спрощує підтримку та розвиток проекту [57].

Незважаючи на багатофункціональність, Vue.js відзначається невеликим розміром ядра, що сприяє зменшенню часу завантаження сторінок. Реактивність даних у Vue.js дозволяє автоматично оновлювати елементи інтерфейсу у відповідь на зміни даних, що значно підвищує швидкість роботи додатку і зменшує ризик помилок [58].

Vue.js має активну та зростаючу спільноту розробників, що забезпечує стабільну підтримку, регулярні оновлення та багатий набір розширень і плагінів. Серед популярних інструментів можна виділити Vuetify для створення адаптивного дизайну та Nuxt.js, що значно спрощує розробку серверних додатків. Детальна документація фреймворку дозволяє розробникам швидко вирішувати складні завдання і зменшує час, необхідний для навчання нових членів команди. Крім того, екосистема Vue.js включає Vuex для управління станом та Vue Router для навігації, що забезпечує повний набір інструментів для створення сучасних веб-додатків [58].

Таким чином, Vue.js є оптимальним вибором для реалізації користувацького інтерфейсу у рамках цього проекту завдяки його гнучкості, простоті, високій продуктивності та активній підтримці спільноти. Ці переваги дозволяють ефективно реалізувати ключові вимоги нашого проекту, такі як інтерактивність, швидкодія та легка масштабованість, що робить його ідеальним інструментом для створення сучасного інтерфейсу. Використання цього фреймворку дозволяє створювати

сучасні, інтерактивні та масштабовані веб-додатки, які відповідають найвищим вимогам користувачів та розробників.

Основна структура проекту

Проект побудовано у відповідності до типової структури Vue.js додатка, що дозволяє підтримувати чистоту, масштабованість і зручність у розробці.

Директорія **assets** містить статичні ресурси, які використовуються у додатку. Тут зберігаються графічні файли, такі як **ADD_logo.png** і **logo.png**, а також **styles.css**, що відповідає за глобальні стилі.

Директорія **components** призначена для багаторазових компонентів інтерфейсу користувача. Наприклад, тут можуть бути створені такі компоненти, як **ButtonComponent.vue** для кнопок з однаковим стилем або **FormInput.vue** для полів вводу, що можуть використовуватися у різних формах додатка. У даному проекті ця директорія поки що не заповнена, але в майбутньому може містити компоненти для UI-елементів, які можна використовувати у різних частинах додатка.

У директорії **plugins** розміщуються файли для підключення сторонніх плагінів і модулів. Файл **axios.js** містить налаштування для обробки HTTP-запитів за допомогою бібліотеки Axios, тоді як **vuetify.js** інтегрує Vuetify, що базується на Material Design для розробки UI.

Директорія **router** містить файл **index.js**, де налаштовано маршрутизацію додатка. Це дозволяє реалізовувати переходи між сторінками за допомогою Vue Router. У цьому файлі задаються маршрути для сторінок та компонентів додатка.

Директорія **views** призначена для зберігання Vue-компонентів, що використовуються як окремі сторінки. Вона містить такі компоненти, як **DisclaimerNotice.vue** (для відображення юридичних або інформаційних застережень), **UserForm.vue** (форма для введення даних користувачем), **PredictionResult.vue** (сторінка з результатами прогнозування) і **AboutProject.vue** (сторінка про проект).

Файл **App.vue** є головним компонентом Vue-додатка, з якого починається його відображення. У ньому зазвичай визначаються основні структури сторінок, загальні елементи інтерфейсу, такі як хедер, футер, бокова панель, а також логіка для інтеграції глобальних компонентів і плагінів. У ньому зазвичай визначаються основні структури і компоненти інтерфейсу. Файл **main.js** відповідає за ініціалізацію Vue-додатка. Тут підключаються основні плагіни, маршрутизація та головний компонент **App.vue**.

У проєкті також наявні конфігураційні та службові файли. Файли **.env** і **.env.production** використовуються для збереження змінних середовища. **.gitignore** дозволяє виключити певні файли та директорії з системи контролю версій. **babel.config.js** налаштовує Babel для сумісності коду з різними браузерами, а **jsconfig.json** надає підтримку JavaScript у редакторі коду.

Файли **package.json** і **package-lock.json** відповідають за керування залежностями проєкту. **package.json** містить список залежностей та основну інформацію про проєкт, а **package-lock.json** фіксує конкретні версії пакетів, що забезпечує стабільність і однаковість встановлення залежностей на різних середовищах. У **README.md** зазвичай розміщуються інструкції для розробників і користувачів проєкту. Файл **server.js** вказує на наявність серверної логіки або проксі для додатка, а **vue.config.js** дозволяє налаштувати кастомні параметри для Vue CLI.

Загальна структура проєкту є логічно організованою та відповідає стандартам Vue.js. Файли та директорії розподілені так, щоб полегшити підтримку, розробку та масштабування додатка. Завдяки модульності структури та розділенню логіки між компонентами, додаток легко адаптується до нових вимог, забезпечуючи ефективне управління кодовою базою і можливість розширення функціоналу.

Ініціалізації Vue-додатка у файлі **main.js**

Файл **main.js** є головним скриптом, відповідальним за ініціалізацію Vue-додатка. Він виконує ключові функції, такі як підключення основних модулів, плагінів і стилів, а також монтування додатка на DOM.

Процес починається з імпорту необхідних залежностей. Спочатку імпортується функція *createApp* з пакету **Vue**, яка створює додаток. Імпортується основний компонент додатка **App.vue**, а також маршрутизація з файлу *./router*. Для глобальних стилів підключається файл *./assets/styles.css*, який містить CSS-код, що застосовується до всього додатка.

```
import { createApp } from 'vue';
import App from './App.vue';
import router from './router';
import './assets/styles.css'; // Import the global styles
```

Плагіни інтегруються шляхом імпорту відповідних конфігурацій. Плагін **Vuetify**, що базується на Material Design, імпортується з файлу *./plugins/vuetify*. Також підключається конфігурація для **axios**, яка використовується для виконання HTTP-запитів.

```
import vuetify from './plugins/vuetify'; // Import the Vuetify plugin
import axios from './plugins/axios';
```

Далі створюється додаток за допомогою функції *createApp*. Цей додаток використовує модулі і плагіни. Спочатку підключається маршрутизація через *.use(router)*, потім додаються **axios** та **Vuetify**, що налаштовують додаток для роботи з HTTP-запитами і надають компоненти інтерфейсу на основі Material Design. Заключним кроком є монтування додатка до DOM-елемента з ідентифікатором *#app*.

```
createApp(App)
  .use(router)
  .use(axios)
  .use(vuetify) // Use Vuetify
  .mount('#app');
```

Ця структура забезпечує гнучкість та масштабованість. Глобальні стилі, маршрути, плагіни і основний компонент об'єднуються в єдиному місці, що робить процес ініціалізації додатка простим та зрозумілим.

Створення основного компонента App.vue

Файл **App.vue** є основним компонентом Vue-додатка, який відповідає за базову структуру інтерфейсу і служить кореневим вузлом для всіх інших компонентів. Цей компонент розроблено у форматі однокомпонентного файлу Vue (Single File Component, SFC), який містить три основні секції: шаблон, скрипт і стилі.

У секції `<template>` визначається HTML-структура, яка буде рендеритися на сторінці. У цьому файлі створюється кореневий елемент `<div>` з атрибутом `id="app"`. Усі дочірні компоненти рендеряться всередині цього кореневого елемента через `<router-view>`, який відповідає за динамічне відображення компонентів на основі маршрутизації.

```
<template>
  <v-app>
    <!-- Верхня панель додатку -->
    <v-app-bar app color="primary" dark>
      <v-container>
        <v-row align="center">
          <!-- Колонка з логотипом та заголовком -->
          <v-col cols="12" md="3" class="d-flex align-center">
            <!-- Логотип -->
            <v-img
              src="@/assets/ADD_logo.png"
              alt="Логотип"
              max-height="40"
              contain
              class="mr-2"
            ></v-img>
            <!-- Заголовок -->
            <v-toolbar-title class="white--text">
              Прогноз діабету
            </v-toolbar-title>
          </v-col>
          <!-- Колонка з навігаційними кнопками -->
          <v-col cols="12" md="9" class="text-right">
            <v-btn text to="/" class="white--text">Головна</v-btn>
            <v-btn text to="/form" class="white--text">Форма</v-btn>
            <v-btn text to="/about" class="white--text">Про нас</v-btn>
          </v-col>
        </v-row>
      </v-container>
    </v-app-bar>

    <!-- Основний контент -->
    <v-main>
```

```

    <v-container>
      <!-- Роутер-в'ю для відображення компонентів -->
      <router-view></router-view>
    </v-container>
  </v-main>

  <!-- Нижній колонтитул -->
  <v-footer app color="primary" dark>
    <v-container>
      <v-row justify="center">
        <v-col cols="12" md="6" class="text-center">
          @ {{ new Date().getFullYear() }} Прогноз діабету
        </v-col>
      </v-row>
    </v-container>
  </v-footer>
</v-app>
</template>

```

Секція `<script>` містить основну логіку компонента. Вона визначає ім'я компонента, а також містить будь-які необхідні дані чи методи. У цьому файлі використовується базова структура, що експортує компонент із заданою назвою.

```

<script>
export default {
  name: 'App',
};
</script>

```

Секція `<style>` відповідає за стилізацію компонента. У цьому прикладі стилі застосовуються глобально і надають базовий вигляд кореневого елемента з налаштуваннями шрифтів, кольору тексту і вирівнювання.

```

<style>
/* Стили для заголовка в верхній панелі */
.v-toolbar-title {
  font-size: 24px;
  font-weight: 500;
}

/* Стили для навігаційних кнопок в верхній панелі */
.v-app-bar .v-btn {
  margin-left: 16px;
}

/* Стили для основного контейнера */

```

```

.v-main {
  background-color: #f5f5f5;
}

/* Стили для контейнера контента */
.v-container {
  padding-top: 16px;
  padding-bottom: 16px;
}

/* Стили для нижнього колонтитулу */
.v-footer {
  padding-top: 8px;
  padding-bottom: 8px;
}
</style>

```

Компонент **App.vue** виступає центральним вузлом додатка, який об'єднує структуру HTML, логіку JavaScript і стилі CSS. Його призначення полягає у забезпеченні місця для рендерингу дочірніх компонентів через маршрутизацію, визначення глобальних стилів і базових налаштувань для інтерфейсу.

Опис структури та компонентів файлу DisclaimerNotice.vue

Секція `<template>` містить розмітку компонента, яка визначає його візуальне представлення. У даному файлі вона включає контейнер з основним текстом застереження, що відображається користувачу. Компонування елементів підтримує інтуїтивний стиль і враховує потребу у мінімалізмі. Наприклад, код з секції:

```

<template>
  <v-container class="fill-height" fluid>
    <!-- Center the card horizontally and vertically -->
    <v-row align="center" justify="center">
      <!-- Card column with responsive width -->
      <v-col cols="12" md="8" lg="6">
        <v-card class="pa-5">
          <!-- Grid container inside the card -->
          <v-container>
            <!-- Title Row -->
            <v-row>
              <v-col cols="12">

```

```

        <v-card-title class="text-h5">
            Увага
        </v-card-title>
    </v-col>
</v-row>

<!-- Text Row -->
<v-row>
    <v-col cols="12">
        <v-card-text>
            Цей сервіс надається "як є". Використання на власний ризик.
            Ми не несемо відповідальності за будь-які наслідки.
        </v-card-text>
    </v-col>
</v-row>

<!-- Button Row -->
<v-row justify="center">
    <v-col cols="12" md="4" class="text-center">
        <v-btn color="primary" @click="goToForm">
            Спробувати безкоштовно
        </v-btn>
    </v-col>
</v-row>
</v-container>
</v-card>
</v-col>
</v-row>
</v-container>
</template>

```

Секція `<script>` містить логіку компонента та визначає його поведінку. У цьому файлі вона використовує синтаксис ES6 та надає модульний підхід для управління даними та логіку для переходу до компонента UserForm.

```

<script>
export default {
  name: 'DisclaimerNotice',
  methods: {
    goToForm() {
      // Переходимо до форми вводу даних
      this.$router.push({ name: 'UserForm' });
    },
  },
};
</script>

```

Основні елементи:

- **export default** визначає компонент для експорту.
- **name** задає ім'я компонента, яке використовується для його ідентифікації у проектах.
- **data** повертає об'єкт, який містить динамічні дані компонента. У даному випадку, це текст застереження, який буде відображений у шаблоні.

Секція `<style>` забезпечує стилізацію компонента, визначаючи його візуальну презентацію. Вона може включати як локальні, так і глобальні стилі.

```
<style scoped>
/* Styling for the card */
.v-card {
  padding: 32px;
}

/* Center and style the title */
.v-card-title {
  font-size: 28px;
  margin-bottom: 16px;
  text-align: center;
}

/* Center and style the text */
.v-card-text {
  font-size: 16px;
  margin-bottom: 24px;
  text-align: center;
}

/* Responsive button width */
.v-btn {
  width: 100%;
}

/* Adjust button width on medium and larger screens */
@media (min-width: 960px) {
  .v-btn {
    width: auto;
  }
}
</style>
```

Використовується атрибут **scoped**, що обмежує вплив стилів лише до цього компонента.

Опис структури та компонентів файлу UserForm.vue

Компонент “UserForm.vue” у секції `template` описує основну структуру користувацького інтерфейсу форми введення даних. Форма реалізована з використанням Vuetify, що забезпечує адаптивність та зручність інтерфейсу завдяки таким компонентам, як для створення макета, для керування валідацією та для взаємодії користувача.

У секції знаходиться структура форми, яка реалізована з використанням компонентів бібліотеки Vuetify. Основний контейнер — це , у якому розташований , що відповідає за валідацію введених даних через прив'язку до змінної `isValid`. Форма містить кілька груп компонентів для вводу даних:

Стать Користувач обирає свою стать за допомогою компонента , де доступні два варіанти: "Чоловіча" і "Жіноча". Вибір обов'язковий.

```
<v-radio-group
  v-model="formData.gender"
  :rules="[rules.required]"
  label="Стать"
  color="primary"
>
  <v-radio label="Чоловіча" value="Male"></v-radio>
  <v-radio label="Жіноча" value="Female"></v-radio>
</v-radio-group>
```

Гіпертонія Користувач вказує, чи є у нього гіпертонія, обираючи "Так" або "Ні". Для цього використовується . Поле обов'язкове.

```
<v-radio-group
  v-model="formData.hypertension"
  :rules="[rules.required]"
  label="Гіпертонія"
  color="primary"
>
  <v-radio label="Так" value="1"></v-radio>
  <v-radio label="Ні" value="0"></v-radio>
</v-radio-group>
```

Серцеві захворювання Користувач вказує, чи є у нього серцеві захворювання, обираючи "Так" або "Ні". Поле обов'язкове.

```
<v-radio-group
  v-model="formData.heart_disease"
  :rules="[rules.required]"
  label="Серцеві захворювання"
  color="primary"
>
  <v-radio label="Так" value="1"></v-radio>
  <v-radio label="Ні" value="0"></v-radio>
</v-radio-group>
```

Вік Поле для введення віку користувача. Використовується з обмеженням діапазону від 1 до 120 років. Поле обов'язкове.

```
<v-text-field
  v-model="formData.age"
  :rules="[rules.required, rules.age]"
  label="Вік"
  type="number"
  min="1"
  max="120"
></v-text-field>
```

Історія паління Користувач обирає історію паління зі списку доступних варіантів ("Немає інформації", "Ніколи", "Раніше", "Зараз") через компонент . Поле обов'язкове.

```
<v-select
  v-model="formData.smoking_history"
  :items="localizedSmokingOptions"
  :rules="[rules.required]"
  label="Історія паління"
></v-select>
```

Зріст (см) Користувач вводить свій зріст у сантиметрах через . Діапазон допустимих значень — від 80 до 250 см. Поле обов'язкове.

```
<v-text-field
  v-model="bodyHeight"
  :rules="[rules.required, rules.bodyHeight]"
  label="Зріст (см)"
  type="number"
></v-text-field>
```

Вага (кг) Користувач вводить свою вагу в кілограмах через . Діапазон допустимих значень — від 30 до 300 кг. Поле обов'язкове.

```
<v-text-field
  v-model="bodyWeight"
  :rules="[rules.required, rules.bodyWeight]"
  label="Вага (кг)"
  type="number"
></v-text-field>
```

Рівень HbA1c (%) Користувач вводить рівень глікованого гемоглобіну у відсотках через . Діапазон допустимих значень — від 2% до 20%. Поле обов'язкове.

```
<v-text-field
  v-model="formData.HbA1c_level"
  :rules="[rules.required, rules.hba1c]"
  label="Рівень HbA1c (%)"
  type="number"
></v-text-field>
```

Рівень глюкози в крові Користувач вводить рівень глюкози у вибраних одиницях ("mg/dL" або "mmol/L"). Поле складається з двох частин: вибір одиниць через і введення числового значення через . Поле обов'язкове.

```
<v-row>
  <v-col cols="5">
    <v-select
      v-model="bloodGlucoseUnit"
      :items="bloodGlucoseUnits"
      label="Одиниці"
    ></v-select>
  </v-col>
  <v-col cols="7">
    <v-text-field
      v-model="bloodGlucoseValue"
      :rules="[rules.required, rules.bloodGlucose]"
      :label="`Рівень глюкози (${bloodGlucoseUnit})`"
      type="number"
    ></v-text-field>
  </v-col>
</v-row>
```

Таким чином, користувач заповнює дані щодо своєї статі, віку, медичного стану, звичок і параметрів тіла, що є необхідними для подальшої обробки.

Нижче наведено опис коду, що міститься у секції script компонента UserForm. Код розділено на логічні частини. Кожну частину описано окремо абзацами без використання нумерованих або маркованих списків. Наведено приклади відповідних фрагментів коду, а після них пояснення їх призначення та логіки роботи.

```
import axios from '../plugins/axios';
```

Цей рядок імпортує бібліотеку `axios`, яка використовується для надсилання HTTP-запитів. Її застосовують для відправлення даних форми на сервер і отримання результатів.

```
export default {
  name: 'UserForm',
  ...
}
```

Експорт за замовчуванням з об'єкта налаштувань компонента `Vue`. Тут визначається назва компонента як `'UserForm'`, а також оголошуються всі дані, методи та обчислювані властивості, необхідні для його роботи.

```
data() {
  return {
    isValid: false,
    formData: {
      gender: '',
      age: null,
      hypertension: '',
      heart_disease: '',
      smoking_history: '',
      bmi: null,
      HbA1c_level: null,
      blood_glucose_level: null,
    },
    bodyHeight: null,
    bodyWeight: null,
    bloodGlucoseUnit: 'mmol/L',
    bloodGlucoseUnits: ['mg/dL', 'mmol/L'],
    bloodGlucoseValue: null,
    smokingHistoryMapping: {
      'No Info': 'Немає інформації',
      'Never': 'Ніколи',
      'Former': 'Раніше',
      'Current': 'Зараз',
    },
    errorMessage: '',
    showError: false,
    rules: {
      required: (value) => !!value || 'Обов'язкове поле',
      age: (value) => (value >= 1 && value <= 120) || 'Вік має бути між 1 та 120',
      bmi: (value) => (value >= 10 && value <= 80) || 'ІМТ має бути між 10 та 80',
      bodyHeight: (value) => (value >= 80 && value <= 250) || 'Зріст має бути між 80см та 250см',
    },
  }
}
```

```

    bodyWeight: (value) => (value >= 30 && value <= 300) || 'Вага має бути між 30кг
та 300кг',
    hba1c: (value) => (value >= 2 && value <= 20) || 'Рівень HbA1c має бути між 2% та
20%',
    bloodGlucose: (value) => {
      if (!value) return 'Обов'язкове поле';
      let mgValue = this.bloodGlucoseUnit === 'mmol/L' ? value * 18 : value;
      return (mgValue >= 40 && mgValue <= 400) || 'Рівень глюкози має бути між 40 та
400 mg/dL';
    },
  },
};
},

```

Блок `data()` визначає всі змінні та стан компонента. Змінна `isValid` відстежує валідність форми. Об'єкт `formData` містить основні дані користувача, такі як стать, вік, наявність гіпертонії, серцевих захворювань, паління та інші параметри для відправки на сервер. Змінні `bodyHeight`, `bodyWeight`, `bloodGlucoseUnit` та `bloodGlucoseValue` необхідні для розрахунку ІМТ та перетворення рівня глюкози до потрібних одиниць. Об'єкт `smokingHistoryMapping` забезпечує зіставлення внутрішніх англomовних значень з локалізованими українськими. Поля `errorMessage` та `showError` керують відображенням можливих помилок. Об'єкт `rules` визначає валідаційні правила для полів форми, перевіряючи чи задані всі необхідні значення, чи відповідають вони допустимим межам, і чи коректно перетворено одиниці вимірювання.

```

computed: {
  localizedSmokingOptions() {
    return Object.values(this.smokingHistoryMapping);
  },
},

```

Обчислювана властивість `localizedSmokingOptions` повертає масив локалізованих значень історії паління. Це дозволяє зручніше відобразити ці значення у випадальному списку безпосередньо у шаблоні, відокремивши логіку трансформації даних від їх відображення.

```

methods: {
  handleSubmit() {
    if (!this.isValid) {
      return;
    }
  }
}

```

```

    }

    this.formData.smoking_history = Object.keys(this.smokingHistoryMapping).find(
      (key) => this.smokingHistoryMapping[key] === this.formData.smoking_history
    );

    this.formData.bmi = this.bodyWeight / Math.pow(this.bodyHeight/100, 2);

    let bloodGlucoseMg =
      this.bloodGlucoseUnit === 'mmol/L'
        ? this.bloodGlucoseValue * 18
        : this.bloodGlucoseValue;
    this.formData.blood_glucose_level = bloodGlucoseMg;

    axios({
      method: 'POST',
      url: '/api/predict',
      data: this.formData,
      headers: {
        'Content-type': 'application/json',
      }
    })
    .then((response) => {
      const probability = (response.data.diabetes_probability * 100).toFixed(1);
      if (response.data.diabetes_prediction === 1) {
        this.$router.push({
          name: 'Result',
          query: {
            message: `Користувач, ймовірно, має діабет. Ймовірність:
${probability}%`,
            model: response.data.diabetes_prediction_model,
          },
        });
      } else {
        this.$router.push({
          name: 'Result',
          query: {
            message: `У користувача, ймовірно, немає діабету. Ймовірність:
${probability}%`,
            model: response.data.diabetes_prediction_model,
          },
        });
      }
    })
    .catch((error) => {
      this.errorMessage = error.response?.data?.message || 'Сталася помилка';
      this.showError = true;
    });
  },
},

```

Метод `handleSubmit` виконує логіку, що відбувається після відправлення форми. Спочатку він перевіряє, чи форма валідна. Якщо ні, процес зупиняється. Якщо так, значення поля `smoking_history` локалізованої опції повертається до початкового англійського значення для відправки на сервер. Обчислюється ІМТ (`bmi`) на основі ваги та зросту користувача. Рівень глюкози перераховується до `mg/dL`, якщо користувач вказав значення у `mmol/L`. Потім за допомогою `axios` надсилається POST-запит на `/api/predict` з усіма необхідними даними. Якщо відповідь сервера містить передбачення діабету та ймовірність, компонент перенаправляє користувача до сторінки результатів. Якщо під час запиту виникла помилка, відображається повідомлення про помилку для користувача.

Таким чином, код у секції `script` визначає логіку компонента: структуру даних, правила валідації, перетворення локалізованих даних у формат для сервера, розрахунки медичних показників, взаємодію з сервером та обробку результатів прогнозу.

Секція `Style`: CSS-стилі оформлення розміщені в секції `style` з атрибутом `scoped`, що гарантує їх застосування лише до цього компонента. Варто розглянути можливість додавання медіазапитів для оптимізації відображення на різних пристроях, особливо для поліпшення адаптивності.

```
<style scoped>
.v-card {
  padding: 24px;
}

.v-card-title {
  margin-bottom: 16px;
}

.v-card-text {
  margin-bottom: 16px;
}

.v-text-field,
.v-select,
.v-radio-group {
  margin-bottom: 24px;
}
```

```

.v-radio-group .v-radio {
  margin-bottom: 8px;
}

.v-btn {
  margin-top: 16px;
}

.v-alert {
  margin-top: 16px;
}

.v-tooltip {
  font-size: 12px;
}
</style>

```

Дані стилі забезпечують послідовність та зручність відображення компонентів інтерфейсу, органічно інтегруючись із темами Vuetify для створення візуально приємного та узгодженого інтерфейсу.

Опис структури та компонентів файлу PredictionResult.vue

Нижче описано окремі частини файлу PredictionResult.vue у вигляді абзаців з прикладами коду. Нумеровані списки та інші маркери не застосовуються, лише текстові абзаци.

У секції `template` представлено розмітку інтерфейсу користувача. Використано компоненти Vuetify для створення адаптивної сітки, картки та кнопки повернення. Основним контейнером є тег `template`, всередині якого знаходиться `v-container` з класом `fill-height`, що розтягує його на всю висоту. Для позиціонування картки застосовано `v-row` з властивостями `align="center"` та `justify="center"`. Всередині картки використано `v-container` та кілька `v-row` та `v-col` для розташування заголовків, текстових повідомлень та кнопки. Загалом ця частина коду визначає візуальну структуру сторінки, що демонструє результати прогнозу.

Приклад коду з секції `template`:

```
<template>
```

```

<v-container class="fill-height" fluid>
  <v-row align="center" justify="center">
    <v-col cols="12" md="8" lg="6">
      <v-card class="pa-5">
        <v-container>
          <v-row>
            <v-col cols="12">
              <v-card-title class="text-h5">
                Результат прогнозу
              </v-card-title>
            </v-col>
          </v-row>

          <v-row>
            <v-col cols="12">
              <v-card-text>
                {{ message }}
              </v-card-text>
              <v-card-text>
                {{ predictionModel }}
              </v-card-text>
            </v-col>
          </v-row>

          <v-row justify="center">
            <v-col cols="12" md="4" class="text-center">
              <v-btn color="primary" @click="goBack">
                Повернутися
              </v-btn>
            </v-col>
          </v-row>
        </v-container>
      </v-card>
    </v-col>
  </v-row>
</v-container>
</template>

```

У секції script реалізовано логіку компонента. Використано `export default`, де вказано ім'я компонента, а також обчислювані властивості `message` та `predictionModel`. Ці властивості отримують дані з `query`-параметрів маршруту. У випадку, якщо параметри не визначені або не відповідають відомим моделям, виводяться значення за замовчуванням. Також тут визначено метод `goBack`, який повертає користувача до форми за допомогою переходу до маршруту з ім'ям `UserForm`. Приклад коду з секції script:

```

<script>
export default {
  name: 'PredictionResult',
  computed: {
    message() {
      return this.$route.query.message || 'Немає результату';
    },
    predictionModel() {
      if(this.$route.query.model && (this.$route.query.model).toLowerCase() === 'lr') {
        return 'Модель передбачення: Логістична регресія (LR)'
      }
      else if(this.$route.query.model && (this.$route.query.model).toLowerCase() ===
'xgboost'){
        return 'Модель передбачення: Градієнтний бустинг (XGBoost)'
      }

      return 'Немає моделі';
    },
  },
  methods: {
    goBack() {
      this.$router.push({ name: 'UserForm' });
    },
  },
};
</script>

```

У секції `style` описано стилі, що стосуються компонента. Застосовано властивість `scoped` для обмеження стилів лише до цього компонента. Встановлено падінг для картки, визначено розмір та вирівнювання заголовків і тексту, а також задано ширину кнопки. Використано медіа-запит для коригування ширини кнопки на великих екранах. Приклад коду з секції `style`:

```

<style scoped>
.v-card {
  padding: 32px;
}

.v-card-title {
  font-size: 28px;
  margin-bottom: 16px;
  text-align: center;
}

.v-card-text {
  font-size: 18px;
  margin-bottom: 24px;
  text-align: center;
}

```

```

}

.v-btn {
  width: 100%;
}

@media (min-width: 960px) {
  .v-btn {
    width: auto;
  }
}
</style>

```

Компонентна структура Vue.js сприяє модульності, багаторазовому використанню коду та спрощенню підтримки.

Опис структури та компонентів файлу AboutProject.vue

Секція Template відповідає за візуальну частину сторінки, яка буде відображатися користувачу. У даному файлі використовується структура `<template>`, що визначає основний каркас сторінки. Фрагмент коду:

```

<template>
  <v-container>
    <v-card>
      <!-- Початок сітки всередині картки -->
      <v-container>
        <!-- Рядок з логотипом та заголовком -->
        <v-row align="center" justify="space-between">
          <v-col cols="12" md="8">
            <v-card-title class="text-h5">
              Автоматизована система прогнозування виявлення цукрового діабету
            </v-card-title>
            <v-text>
              <p>Проект було виконано в рамках виконання дипломної роботи студента
                групи АТ-28м <strong>Підопригори С.В.</strong></p>
              <p>Науковий керівник: кандидат технічних наук <strong>Ткачук
                А.Г.</strong></p>
              <p>Тема дипломної роботи: <strong><em>"Автоматизована система
                прогнозування виявлення цукрового діабету на основі медичних даних
                пацієнтів"</em></strong></p>
            </v-text>
          </v-col>
          <v-col cols="12" md="4" class="text-right">
            <!-- Логотип -->
            <v-img

```

```

        src="@/assets/ADD_logo.png"
        alt="Логотип"
        max-height="140"
        contain
    ></v-img>
</v-col>
</v-row>

<!-- Рядок з контентом -->
<v-row>
  <v-col cols="12">
    <!-- Використовуємо компоненти типографії для оформлення тексту -->
    <v-card-text>
      <v-divider></v-divider>
      <v-card-subtitle class="mt-4">
        Про проект
      </v-card-subtitle>
      . . .
      . . .
      . . .
    </v-row>
  </v-container>
  <!-- Кінець сітки всередині картки -->
</v-card>
</v-container>
</template>

```

Основне призначення цієї секції — забезпечити чітку та зрозумілу структуру візуальних компонентів сторінки.

Секція Script використовується для додавання логіки до компонента. У файлі AboutProject.vue застосовано синтаксис ES6 та стандартні методи Vue.js. Наприклад, основна структура компонента визначена наступним чином:

```

<script>
export default {
  name: 'AboutProject',
};
</script>

```

Це дозволяє відокремлювати логіку від візуальної частини, забезпечуючи гнучкість у відображенні даних.

Секція Style забезпечує оформлення елементів, створених у шаблоні. У файлі застосовано `scoped`-стили, які обмежують вплив CSS тільки на поточний компонент. Ось приклад оформлення:

```
<style scoped>
/* Стили для картки */
.v-card {
  padding: 24px;
}

/* Стили для заголовка */
.v-card-title {
  margin-bottom: 16px;
  font-size: 24px;
  font-weight: bold;
}

/* Стили для підзаголовків */
.v-card-subtitle {
  font-size: 20px;
  font-weight: bold;
  margin-bottom: 8px;
}

/* Стили для тексту */
.v-text {
  font-size: 16px;
  line-height: 1.6;
  margin-bottom: 16px;
  display: block;
}

/* Стили для списків */
.v-list-item-content {
  font-size: 16px;
}

/* Стили для логотипу */
.v-img {
  max-height: 60px;
}

/* Відступи для списків */
.v-list {
  margin-bottom: 16px;
}
</style>
```

Ці стилі створюють привабливий та професійний вигляд сторінки.

Використання даних компонентної структури Vue.js дозволяє забезпечити модульність, повторне використання коду та легкість підтримки. Кожна секція має своє чітке призначення: шаблон — для візуалізації, скрипт — для логіки, стилі — для оформлення.

5 Інтеграція апаратних модулів через REST API для віддаленого медичного аналізу

5.1 Опис протоколу HTTP та RESTful API

Протокол HTTP (Hypertext Transfer Protocol) — це основа сучасної веб-архітектури, яка використовується для передачі даних між клієнтом і сервером. Наприклад, коли користувач вводить URL у веб-браузері, браузер надсилає HTTP-запит на сервер, який повертає веб-сторінку для відображення. HTTP є протоколом прикладного рівня, що базується на моделі запит-відповідь. Клієнт, наприклад веб-браузер або REST-клієнт, надсилає запит до сервера, а сервер обробляє його і повертає відповідь. HTTP є текстово орієнтованим протоколом, який використовує статичну структуру повідомлень для комунікації [59].

Запит HTTP складається з кількох ключових елементів. Спершу зазначається метод запиту, такий як GET, POST, PUT, DELETE, PATCH тощо. Ці методи мають конкретне призначення: GET використовується для отримання даних без внесення змін, POST дозволяє створювати нові ресурси, PUT оновлює весь ресурс, PATCH змінює його частину, а DELETE видаляє ресурс. Така структура дозволяє реалізовувати операції CRUD (Create, Read, Update, Delete), забезпечуючи чіткий і стандартизований підхід до роботи з даними. Ці методи визначають дію, яку клієнт бажає виконати. Наприклад, метод GET використовується для отримання ресурсу, тоді як POST слугує для надсилання даних на сервер. Після методу вказується URL (Uniform Resource Locator), що визначає шлях до ресурсу на сервері. Далі йде версія протоколу HTTP (наприклад, HTTP/1.1 або HTTP/2). Запит також може містити заголовки, які передають додаткову інформацію, наприклад, тип вмісту, формат прийнятного відповіді або дані авторизації. У разі методів POST чи PUT запит включає також тіло, яке містить передані дані, зазвичай у форматі JSON або XML.

Відповідь HTTP має подібну структуру. Вона включає статусний код, наприклад, 200 для успіху, 404 для відсутності ресурсу або 500 для помилки сервера. Заголовки відповіді передають метадані, таку як тип вмісту чи інформація про кешування. Тіло відповіді зазвичай містить дані, які запитував клієнт.

RESTful API використовує HTTP як основу для створення архітектурно незалежних веб-сервісів. Архітектура REST (Representational State Transfer) передбачає використання стандартних методів HTTP для виконання операцій CRUD (Create, Read, Update, Delete). Це забезпечує інтуїтивно зрозумілий інтерфейс для взаємодії з ресурсами [59].

Відправка запитів по RESTful API починається з формування правильного запиту. Наприклад, щоб отримати дані про пацієнта, використовується метод GET із зазначенням відповідного ресурсу, такого як “/api/patients/{id}”. Для створення нового запису пацієнта метод POST надсилає JSON-об'єкт із даними пацієнта на ресурс “/api/patients”. Зміна даних виконується за допомогою методу PUT або PATCH, а видалення — через DELETE.

RESTful API забезпечує розширену функціональність через параметри запиту та фільтрацію. Наприклад, передача параметрів у URL дозволяє сортувати, фільтрувати або здійснювати пошук. Крім того, для авторизації та аутентифікації зазвичай використовуються токени доступу, що передаються через заголовки Authorization.

Для інтеграції з апаратними модулями віддаленого медичного аналізу, сервер, який прогнозує наявність цукрового діабету, приймає запити від апаратних модулів, обробляє дані через ML алгоритми, повертаючи результат прогнозу. Наприклад, POST-запит до “/api/predict” повинен містити JSON-об'єкт із наступними параметрами:

```
{
  "gender": "Female",
  "age": 45.0,
  "hypertension": 1,
  "heart_disease": 0,
  "smoking_history": "never",
  "bmi": 28.5,
  "HbA1c_level": 7.5,
  "blood_glucose_level": 110
}
```

Сервер аналізує дані, запускає алгоритм прогнозування і повертає JSON-відповідь із результатом передбачення. Приклад відповіді:

```
{
  "diabetes_prediction": 0,
  "diabetes_prediction_model": "LR",
  "diabetes_probability": 0.26,
```

```
"msg": ""  
}
```

Таким чином, протокол HTTP і RESTful API надають потужний інструмент для побудови інтерактивних систем, особливо у медичній сфері. Вони дозволяють ефективно інтегрувати апаратні модулі віддаленого медичного аналізу, забезпечуючи швидкий і зручний доступ до даних пацієнтів. Це спрощує обробку медичної інформації та прийняття обґрунтованих рішень, що є критично важливим для своєчасного лікування та діагностики. Ці технології спрощують передачу даних між клієнтом і сервером, забезпечуючи ефективність, масштабованість і зручність у використанні.

5.2 Опис модуля ESP8266 (Arduino), як основи для систем віддаленого медичного аналізу

Модуль ESP8266 на платформі Arduino є популярним вибором для інтеграції з системами прогнозування завдяки своїм потужним можливостям роботи з мережею Wi-Fi та підтримкою HTTP-запитів. У цьому розділі розглянуто приклад підключення модуля ESP8266 до REST API для передавання даних пацієнта та отримання результатів прогнозування наявності цукрового діабету.

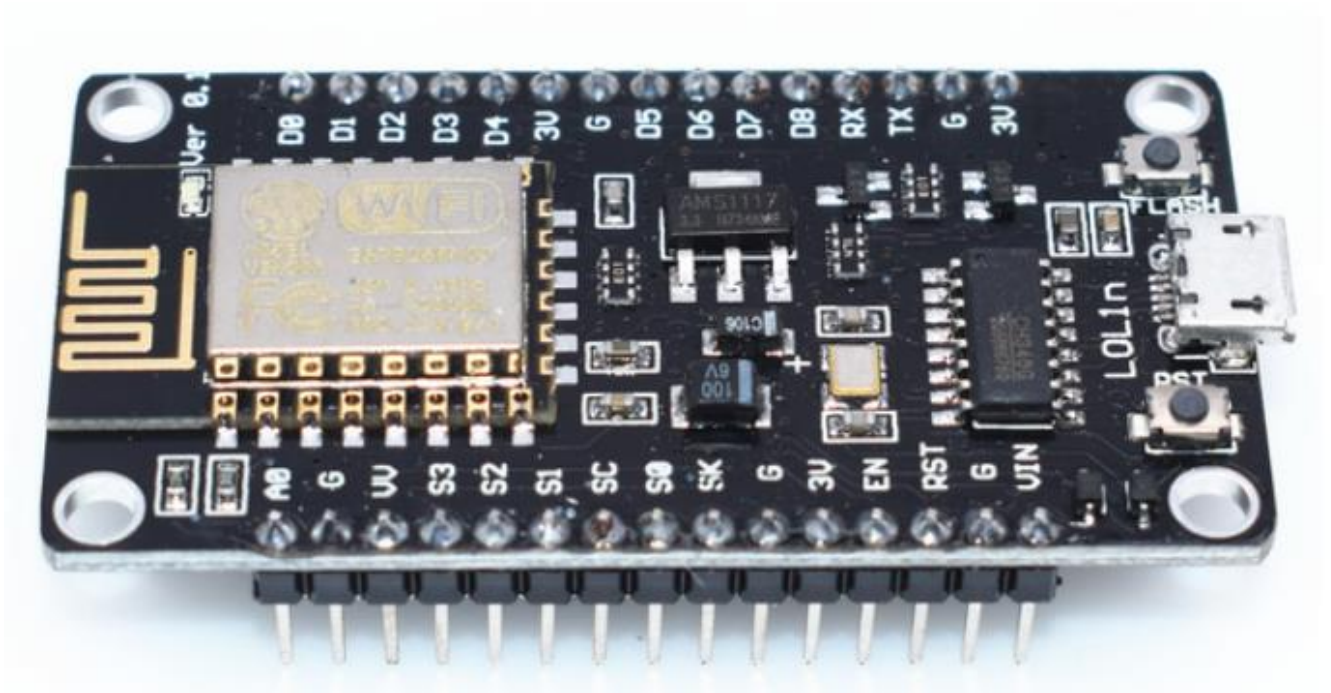


Рис. 5.1. WI-FI модуль NodeMCU ESP8266 CH340

ESP8266 є високоефективним мікроконтролером, розробленим компанією Espressif Systems. Завдяки його багатофункціональності, низькій вартості та компактності, цей модуль став одним із найбільш популярних у розробці пристроїв для Інтернету речей (IoT). Основною перевагою ESP8266 є його вбудована підтримка Wi-Fi, що дозволяє легко інтегрувати його в мережеві застосунки. Модуль ідеально підходить для реалізації сценаріїв, таких як розумні будинки, моніторинг навколишнього середовища, бездротове управління пристроями та передача даних у реальному часі [61].

ESP8266 оснащений 32-бітним процесором Tensilica Xtensa L106, який працює на частоті 80 МГц, з можливістю розгону до 160 МГц. Модуль включає в себе 80 КБ оперативної пам'яті (RAM) та підтримує до 16 МБ зовнішньої флеш-пам'яті через інтерфейс SPI. Завдяки цьому він здатен виконувати складні обчислювальні задачі, одночасно залишаючись компактним (рис. 5.1) [61].

Wi-Fi підтримує стандарти IEEE 802.11 b/g/n, з шифруванням WEP та WPA/WPA2, що забезпечує високу безпеку підключення. Живлення модуля здійснюється від джерела 2,2–3,6 В, а його споживання струму змінюється залежно від режиму роботи:

- **Режим передачі:** до 215 мА.
- **Режим прийому:** до 100 мА.
- **Режим очікування:** близько 70 мА.
- **Режим енергозбереження:** від 15 мкА до 15 мА.

ESP8266 має до 14 GPIO портів, 11 з яких доступні для використання. До цих портів можна підключати різноманітні пристрої, такі як датчики температури, вологості, освітленості, кнопки, світлодіоди, а також реле для управління іншими електронними компонентами. Підтримуються такі інтерфейси:

- SPI (послідовний периферійний інтерфейс)
- I²C (програмно реалізований)
- UART (універсальний асинхронний приймач-передавач)
- I²S (для обробки звуку)

Також модуль має 10-бітний АЦП для аналого-цифрового перетворення, що дозволяє вимірювати аналогові сигнали.

Модуль підтримує різні режими роботи, що робить його зручним для енергоефективних проєктів:

1. У режимі Modem sleep споживання енергії мінімізується, коли Wi-Fi залишається увімкненим, але не здійснюється активний обмін даними. Це дозволяє знизити витрати енергії без необхідності відключення мережевого модуля.
2. У режимі Light sleep споживання енергії додатково знижується шляхом відключення центрального процесора, що дозволяє значно зменшити витрати енергії під час коротких періодів бездіяльності.
3. Deep sleep -- споживання зменшується до мікроампер, що ідеально для пристроїв із періодичним виходом з режиму очікування.

ESP8266 представлений у кількох модифікаціях. Найбільш популярні: ESP-01, ESP-12E та ESP-12F. ESP-01 має 8 виводів, з яких доступні два GPIO; підходить для простих завдань, таких як керування світлодіодами, моніторинг температури через датчики DS18B20 або створення базових бездротових передавачів для передачі даних. ESP-12E та ESP-12F мають 22 виводи, з яких доступні 11 GPIO; ідеально підходять для складних проєктів [61].

Для програмування ESP8266 широко використовується середовище Arduino IDE, яке забезпечує зручність у розробці навіть для початківців. Також підтримуються інші платформи, такі як NodeMCU, MicroPython, та Espressif SDK.

ESP8266 широко використовується для розробки IoT-пристроїв, включаючи розумні будинки, автоматизовані системи моніторингу, датчики навколишнього середовища, контролери освітлення тощо. Його можливості дозволяють створювати інтерактивні рішення з підтримкою мережевого підключення.

ESP8266 є універсальним і потужним інструментом для розробки IoT-проєктів. Його багатофункціональність, енергоефективність та простота інтеграції роблять його незамінним у сучасній електроніці. Завдяки підтримці широкого спектра програмних інструментів, цей модуль може використовуватися як у простих, так і у високотехнологічних рішеннях [61].

5.3 Приклад інтеграції модуля ESP8266 (Arduino) з системою прогнозування виявлення діабету

ESP8266 дозволяє легко встановлювати з'єднання з Wi-Fi мережею та виконувати запити до серверів через HTTP протокол. Цей модуль містить вбудований TCP/IP стек, який забезпечує просте налаштування з'єднань і обмін даними між пристроями в мережі. Його можна використовувати для надсилання запитів, отримання відповідей та обробки даних у реальному часі. Для коректної роботи необхідно використовувати бібліотеки **ESP8266WiFi.h** та

ESP8266HTTPClient.h, які забезпечують відповідні API для підключення до мережі та виконання HTTP-запитів.

Перш за все, необхідно налаштувати параметри мережі. Рекомендується використовувати надійні паролі для Wi-Fi та уникати загальнодоступних мереж без шифрування. Для підвищення рівня безпеки можна також обмежити доступ до API за допомогою фільтрації IP-адрес або токенів аутентифікації. Для цього у скетчі визначаються змінні SSID і пароль для Wi-Fi мережі. Крім того, вказується URL-адреса сервера, який відповідає за обробку запитів. Сервер може бути розташований у локальній мережі або у хмарі [62].

Нижче наведено приклад програмного коду, який ілюструє підключення модуля ESP8266 до Wi-Fi та передачу JSON-даних через HTTP POST-запит. Код використовує параметри для відправлення медичних даних пацієнта до серверу для аналізу:

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>

// Введіть ваші дані мережі:
const char* ssid = "YOUR_WIFI_SSID";
const char* password = "YOUR_WIFI_PASSWORD";

// Адреса вашого сервера. "127.0.0.1" зсередини ESP8266 буде некоректна,
// оскільки це localhost для самого модуля. Використайте IP-адресу сервера або
// відповідний домен мережі.
const char* serverName = "http://192.168.1.100:5000/predict";

void setup() {
  Serial.begin(115200);
  delay(1000);

  // Підключаємось до Wi-Fi
  Serial.print("Підключення до ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Підключено! IP-адреса: ");
```

```

Serial.println(WiFi.localIP());
}

void loop() {
  if (WiFi.status() == WL_CONNECTED) {
    // Формуємо JSON-тіло запиту
    String jsonData = R"({
      "gender": "Female",
      "age": 45.0,
      "hypertension": 0,
      "heart_disease": 0,
      "smoking_history": "never",
      "bmi": 28.5,
      "HbA1c_level": 5.9,
      "blood_glucose_level": 120
    })";

    // Створюємо HTTP клієнт
    HTTPClient http;
    http.begin(serverName); // Ініціалізація підключення до URL
    http.addHeader("Content-Type", "application/json"); // Додаємо заголовок

    // Виконуємо POST-запит
    int httpCode = http.POST(jsonData);

    // Перевіряємо відповідь сервера
    if (httpCode > 0) {
      // httpCode буде статус-кодом HTTP
      Serial.print("HTTP-код відповіді: ");
      Serial.println(httpCode);

      // Якщо відповідь містить тіло, виводимо його
      String payload = http.getString();
      Serial.println("Відповідь сервера:");
      Serial.println(payload);
    } else {
      // Якщо помилка під час з'єднання або відправки запиту
      Serial.print("Помилка підключення: ");
      Serial.println(http.errorToString(httpCode).c_str());
    }

    http.end(); // Закриваємо з'єднання
  } else {
    Serial.println("WiFi не підключено!");
  }

  delay(10000); // Чекаємо 10 секунд перед наступною спробою
}

```

При інтеграції модуля ESP8266 із системою прогнозування важливо забезпечити відповідність формату даних вимогам API. Також необхідно врахувати, що у випадку відсутності з'єднання з сервером, модуль повинен коректно обробляти помилки та повторювати спроби з'єднання.

Інтеграція модуля ESP8266 з системою прогнозування є потужним інструментом для створення IoT-рішень у медичній сфері. Цей приклад демонструє, як використання сучасних технологій дозволяє автоматизувати обмін даними між пристроями і сервером, що значно підвищує ефективність роботи таких систем.

6 ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ

6.1 Можливі схеми монетизації технології машинного навчання для діагностики цукрового діабету

Технології штучного інтелекту (AI) та машинного навчання (ML), застосовані для діагностики та прогнозування цукрового діабету, мають великий потенціал для монетизації. Існує кілька можливих схем, які можуть бути використані для генерації прибутку від таких рішень [59]:

1. Один із найбільш поширених підходів — це створення спеціалізованого програмного забезпечення для медичних установ, яке використовує AI/ML для автоматизації діагностики та прогнозування захворювань. Медичні заклади можуть купувати ліцензії для використання цього ПЗ або підписуватися на SaaS-моделі (програмне забезпечення як послуга), що забезпечує постійний дохід для розробників.
2. Іншою популярною моделлю монетизації є передплата на використання хмарних платформ, що надають послуги з аналізу медичних даних на основі AI. Такі рішення можуть бути доступні як для великих лікарень, так і для приватних клінік, що прагнуть отримати доступ до новітніх технологій діагностики без необхідності розгортати власну інфраструктуру.
3. AI-системи можуть бути інтегровані в платформи EHR для автоматизованого аналізу даних пацієнтів і прогнозування діабету. Розробники таких рішень можуть заробляти через угоди з постачальниками EHR-систем або стягувати плату за кожен оброблений запит або обчислення прогнозу для пацієнта.
4. AI-інструменти, здатні прогнозувати розвиток діабету, можуть бути цінними для фармацевтичних компаній та страхових компаній. Фармацевтичні компанії можуть використовувати такі технології для вибору цільових груп пацієнтів для клінічних випробувань або для просування своїх продуктів. Страхові компанії можуть впроваджувати ці системи для оцінки ризиків

пацієнтів і розробки індивідуальних страхових планів, ґрунтуючись на прогнозах AI.

5. У поєднанні з телемедициною, AI/ML-алгоритми можуть використовуватися для онлайн-діагностики діабету або його переддіабетичних станів. Платформа може стягувати плату за кожну консультацію або використовувати модель передплати для надання послуг пацієнтам на постійній основі. Це дозволяє пацієнтам отримувати своєчасну допомогу та рекомендації віддалено.
6. Розробники можуть укладати партнерства з медичними та фармацевтичними компаніями для реклами своїх продуктів або послуг на платформах, що використовують AI для діагностики діабету. Це може бути модель спільного використання доходів або продаж рекламних місць на платформі.

6.2 Розрахунок собівартості створення та просування сервісу для прогнозування виявлення цукрового діабету

Розрахунок собівартості створення та просування сервісу для прогнозування виявлення цукрового діабету включає декілька важливих етапів, які охоплюють як технічні, так і маркетингові аспекти. Для коректного обчислення собівартості необхідно врахувати такі основні категорії витрат:

Розробка програмного забезпечення

Зарплата програмістів, фахівців з машинного навчання (ML), дата-інженерів та DevOps-фахівців. Витрати на кожного залежатимуть від регіону та рівня експертизи команди.

Збір, очищення та підготовка даних для навчання моделей потребує часу та ресурсів. Якщо використовується платна база даних або сторонні набори даних, це також слід враховувати.

Вартість розробки = кількість розробників × \$3000/міс. × 3 місяці

Тестування програмного забезпечення, включаючи автоматизоване тестування, ручне тестування та А/В-тестування.

Вартість тестування = кількість тестувальників $\times$ \$1500/міс. $\times$ 3 місяці

Ось основні фахівці, які можуть бути потрібні для успішного виконання проекту:

1. Data Scientist (Фахівець з даних) 2–3 спеціаліста;
2. Backend Developer (Розробник бекенду) 1–2 спеціаліста;
3. Frontend Developer (Розробник фронтенду) 1 спеціаліст;
4. DevOps Engineer (Фахівець з інфраструктури та розгортання) 1 спеціаліст;
5. Machine Learning Engineer (Інженер з машинного навчання) 1–2 спеціаліста;
6. Database Administrator (Адміністратор баз даних) 1 спеціаліст;
7. UI/UX Designer (Дизайнер інтерфейсів) 1 спеціаліст;
8. QA Engineer (Тестувальник) 2 спеціаліста;
9. Project Manager (Менеджер проекту) 1 спеціаліст;

Загалом, для повної реалізації проекту знадобиться команда із 11-14 розробників та спеціалістів з різних галузей. З урахуванням того, що на розробку знадобиться 3-4 місяці, вартість робіт буде складати приблизно \$150000,00.

Хостинг та інфраструктура.

Вартість серверів, хмарних рішень (наприклад, AWS, Google Cloud) для розгортання моделей машинного навчання та обробки великих обсягів даних близько будуть складатись з витрат на:

- Хмарні сервери для роботи сервісу, зберігання даних, оновлення та підтримка безпеки близько - \$1500.00/міс;
- Зарплати технічних фахівців, які займаються підтримкою продукту (гарячі лінії, служби підтримки клієнтів) - \$1000.00/міс .

Тобто вартість витрати на інфраструктуру буде близько - \$30000.00/рік.

Маркетинг та просування

Витрати на просування продукту будуть складатись з витрат на:

- Визначення цільового ринку (лікарі, медичні установи, страхові компанії, пацієнти) та вартість реклами для кожного сегменту.
- Витрати на рекламу у соцмережах (Facebook, Google Ads), SEO-оптимізація сайту, просування через контент-маркетинг (статті, вебінари, блоги).
- Вартість співпраці з медичними організаціями, страховими компаніями та клініками, які можуть використовувати або пропонувати сервіс.
- Витрати на участь у медичних конференціях, наукових симпозіумах та виставках, щоб залучити потенційних клієнтів і партнерів.

Вартість витрат на просування продукту складатиме близько \$20000.00/рік.

Юридичні та регуляторні витрати

Сертифікація програмного забезпечення як медичного пристрою у відповідних регулюючих органах (наприклад, FDA, CE для Європи), що може вимагати часу та значних коштів. Також витрати на юридичні консультації щодо захисту даних (GDPR, HIPAA) та захисту інтелектуальної власності будуть складати \$10000.00/рік.

Інші можливі витрати

Інші можливі витрати можуть складати:

- Вартість навчання медичних працівників чи технічних команд у лікарнях та клініках для інтеграції рішення в їхню практику.

- Витрати на інтеграцію сервісу з іншими медичними системами (електронні медичні записи, телемедицина тощо).
- Витрати на захист інтелектуальної власності (патенти на методи діагностики, алгоритми).
- Залежно від обсягу фінансових операцій і географічного охоплення сервісу, річні витрати на банківське обслуговування можуть бути близько 3-5% від суми транзакцій.

Ці витрати можуть складати близько \$20000.00/рік.

Отже розрахунок собівартості створення та просування сервісу для прогнозування виявлення цукрового діабету та перший рік експлуатації цієї системи буде складати приблизно \$230000,00 - \$250000,00.

6.3 Мінімальної кількості клієнтів, необхідної для виходу на точку беззбитковості

Точка беззбитковості визначається як обсяг продажів, при якому дохід бізнесу повністю покриває витрати, але не приносить прибутку. Для її розрахунку необхідно врахувати постійні витрати та маржинальний дохід з кожного клієнта. У випадку, коли фіксовані витрати бізнесу становлять \$250 000 на рік, а ціна послуги або підписки встановлена на рівні \$1000, необхідно залучити 250 клієнтів, щоб дохід дорівнював витратам. Це обчислюється шляхом ділення загальної суми витрат на вартість послуги.

Сума \$1000 на рік є конкурентоспроможною для ринку, якщо продукт або послуга пропонують реальну цінність для споживачів. Наприклад, якщо послуга допомагає компаніям або приватним особам заощаджувати час, гроші або розвивати бізнес, така ціна може бути виправданою. У цьому випадку 250 клієнтів виглядають досяжним орієнтиром, навіть на локальному ринку України.

Український ринок має потенціал для залучення кількох сотень клієнтів, особливо якщо бізнес чітко визначить свою цільову аудиторію та ефективно реалізує маркетингову стратегію. Таким чином, розрахунок точки беззбитковості демонструє, що за умови правильної оцінки витрат і встановлення ціни бізнес може швидко досягти стабільності й почати генерувати прибуток [59].

6.4 Висновок щодо перспектив розвитку автоматизованої система прогнозування виявлення цукрового діабету

Економічне обґрунтування розвитку автоматизованої системи прогнозування виявлення цукрового діабету демонструє її значний потенціал, враховуючи сучасні тенденції в медицині, такі як впровадження телемедицини, використання електронних медичних записів (EHR) та розвиток хмарних платформ для зберігання і аналізу медичних даних, а також постійно зростаючий попит на інноваційні рішення. Застосування технологій машинного навчання для прогнозування та діагностики захворювань може суттєво підвищити ефективність медичних установ, оптимізуючи процес діагностики та дозволяючи медикам зосередитися на лікуванні. Монетизація таких технологій можлива через кілька різноманітних моделей, що включають продаж ліцензій, SaaS-рішення, інтеграцію з електронними медичними записами та співпрацю з фармацевтичними компаніями. Найперспективнішою є SaaS-модель, оскільки вона забезпечує стабільний потік доходів завдяки підпискам, а також знижує бар'єри для впровадження, оскільки користувачам не потрібно інвестувати в розгортання власної інфраструктури. Ці підходи дозволяють значно розширити ринок збуту, включаючи як державні, так і приватні медичні установи.

Оцінка витрат показує, що початкові інвестиції для створення сервісу є значними, складаючи близько \$230,000–250,000 на перший рік, з урахуванням розробки, інфраструктури, маркетингу та юридичних витрат. Проте ринок послуг прогнозування діабету має потенціал для залучення значної кількості клієнтів, особливо якщо ціна послуги є конкурентоспроможною. На цьому ринку вже активно працюють міжнародні компанії, такі як IBM Watson Health і Google Health, які

пропонують хмарні сервіси на основі штучного інтелекту. Конкуренція з такими гравцями вимагає чіткого позиціонування, пропонування унікальних переваг і локалізації продукту для потреб регіональних ринків. Наприклад, встановлення ціни на рівні \$1000 на рік дозволяє бізнесу вийти на точку беззбитковості при залученні 250 клієнтів. Враховуючи глобальну тенденцію зростання попиту на хмарні медичні рішення, досягнення цього орієнтира виглядає цілком реалістичним, особливо при правильному маркетинговому підході [59].

Важливо зазначити, що додатковими перевагами системи є її універсальність та можливість інтеграції з іншими медичними платформами, такими як EHR-системи, що відкриває нові перспективи для партнерств. Особливо перспективними виглядають моделі, орієнтовані на співпрацю зі страховими та фармацевтичними компаніями, які можуть використовувати дані для оптимізації своїх процесів. Інвестиції в маркетинг, сертифікацію та просування дозволяють значно збільшити охоплення і забезпечити вихід на міжнародний ринок. Найефективнішими маркетинговими стратегіями можуть бути таргетована реклама для медичних установ, участь у спеціалізованих конференціях і виставках, співпраця з ключовими гравцями галузі, такими як страхові компанії та великі лікарні, а також використання SEO та контент-маркетингу для залучення потенційних клієнтів через онлайн-канали.

Таким чином, розвиток автоматизованої системи прогнозування цукрового діабету є економічно доцільним за умови ретельного планування витрат, визначення цільової аудиторії та ефективної монетизації продукту. Високий попит на такі рішення, їх здатність оптимізувати медичні процеси та різноманітні схеми монетизації створюють передумови для довгострокової економічної вигоди [59].

ВИСНОВОК

Узагальнення отриманих результатів

У результаті виконання роботи створено автоматизовану систему прогнозування виявлення цукрового діабету на ранніх стадіях, що відповідає сучасним вимогам медичної практики. Використання технологій машинного навчання дозволило забезпечити високу точність та ефективність аналізу великих обсягів медичних даних пацієнтів, зокрема алгоритмів градієнтного бустингу та регресійних моделей. Проведено обґрунтування вибору алгоритмів аналізу даних, оптимізованих для поставленого завдання.

Інтеграція розробленої системи у веборієнтоване середовище зробила її доступною для широкого кола користувачів, включаючи медичних працівників та пацієнтів. Наприклад, лікар може використовувати систему для швидкої оцінки ризику діабету у пацієнта під час прийому, а пацієнт має змогу самостійно ввести дані про свій стан через інтуїтивний інтерфейс та отримати рекомендації щодо подальших дій. Зручний користувацький інтерфейс сприяє легкості використання, а модульна архітектура забезпечує масштабованість і можливість інтеграції з існуючими медичними інформаційними системами.

Результати роботи мають практичну цінність для медичних закладів, оскільки сприяють підвищенню точності діагностики та ефективності лікування; для страхових компаній — через можливість оцінки ризиків клієнтів і розробку адаптованих страхових планів; для пацієнтів — завдяки зручному моніторингу стану здоров'я та своєчасним рекомендаціям; для урядових і фармацевтичних організацій — через доступ до агрегованих даних, що полегшує планування клінічних досліджень і оптимізацію витрат. Система сприяє підвищенню точності діагностики, персоналізації лікування та профілактики захворювання, що є важливим кроком у боротьбі з цукровим діабетом.

Отримані результати можуть бути використані як основа для подальших досліджень і розробок у сфері медичних інформаційних систем та штучного інтелекту, орієнтованих на поліпшення здоров'я населення. Зокрема,

перспективними напрямками є створення систем індивідуального прогнозування ризиків розвитку ускладнень, інтеграція з носимими пристроями для безперервного моніторингу стану здоров'я та розробка алгоритмів для персоналізованого підбору терапії.

Внесок у вирішення проблеми автоматизованого виявлення діабету

Розробка автоматизованої системи прогнозування для раннього виявлення цукрового діабету робить важливий внесок у вдосконалення сучасної медицини. Вона не лише відповідає на виклики, пов'язані зі зростанням захворюваності, але й допомагає ефективно знижувати економічні витрати, пов'язані з лікуванням ускладнень, підвищувати продуктивність населення та зменшувати навантаження на медичну інфраструктуру.

Ключовим аспектом дослідження є інтеграція методів машинного навчання для аналізу великих обсягів медичних даних, що дозволяє з високою точністю передбачати розвиток захворювання. Використання таких алгоритмів, як логістична регресія, градієнтний бустинг або нейронні мережі, дозволяє створювати моделі, які враховують складні залежності між множинними факторами ризику. Це значно підвищує швидкість і точність діагностики порівняно з традиційними методами.

Проект також сприяє підвищенню доступності медичних технологій. Інтеграція розробленої системи у веборієнтоване середовище робить її доступною для медичних працівників і пацієнтів, незалежно від їхнього місцезнаходження. Це особливо важливо для регіонів із обмеженими ресурсами або недостатньо розвиненою медичною інфраструктурою.

Практичне значення роботи

Практичне значення розробки полягає у зниженні витрат на діагностику та лікування діабету. Наприклад, впровадження цієї системи може зменшити витрати

завдяки скороченню потреби в дорогих аналізах і своєчасному виявленню захворювання, що дозволяє уникнути витрат на лікування ускладнень. Система дозволяє виявляти захворювання на ранніх стадіях, використовуючи сучасні технології аналізу даних та сенсори для моніторингу рівня глюкози, що є вирішальним для запобігання ускладнень, таких як серцево-судинні захворювання або ураження нирок. Окрім того, раннє втручання підвищує якість життя пацієнтів і зменшує навантаження на медичну систему.

Наукова новизна роботи полягає у створенні концепції прогнозування, яка об'єднує сучасні алгоритми машинного навчання з гнучкою архітектурою інтерактивного інтерфейсу. Такий підхід дозволяє адаптувати систему до різних задач, що стосуються прогнозування та управління іншими хронічними захворюваннями, такими як гіпертонія, серцева недостатність або хронічні захворювання легень, відкриваючи перспективи її використання в різних галузях медицини.

Ця робота є важливим кроком у напрямку автоматизації процесів діагностики, сприяючи інтеграції інноваційних технологій у практичну медицину, і демонструє, як наука та технології можуть об'єднати зусилля для покращення охорони здоров'я у глобальному масштабі.

РЕКОМЕНДАЦІЇ

Подальші напрямки розвитку системи

Подальший розвиток системи прогнозування виявлення цукрового діабету, побудованої на засадах машинного навчання та інтегрованої у веборієнтоване середовище, може спрямовуватися на вдосконалення як точності моделей, так і розширення функціональних можливостей для різних груп користувачів. Одним із ключових векторів удосконалення є поглиблення персоналізації, що полягає у врахуванні дедалі ширшого спектра індивідуальних медичних показників, інформації про генетичну схильність, спосіб життя, харчові звички та наявність супутніх захворювань. Така персоналізація сприятиме максимально точному визначенню ризиків та ефективніших рекомендацій, зокрема індивідуалізованих планів профілактики та терапії.

Іншим важливим напрямком є вдосконалення алгоритмічних методів та розробка нових моделей, здатних не лише прогнозувати ризики, але й аналізувати довготривалу динаміку стану здоров'я пацієнта. Це може передбачати розробку модулів для прогнозування ймовірності виникнення ускладнень, моделювання впливу різних стратегій лікування та навіть врахування соціально-економічних факторів, що впливають на стан здоров'я пацієнтів. Інтеграція з носимими пристроями та сенсорами відкриває можливості для безперервного контролю рівня глюкози, фізичної активності та інших параметрів, які можуть оновлювати прогностичні моделі в реальному часі, підвищуючи їхню чутливість та оперативність.

Важливим аспектом розвитку є також удосконалення інтерфейсу користувача, що може включати створення мобільних додатків для моніторингу багатьох параметрів стану здоров'я пацієнтів, а також стилю їх життя і харчування. Це допоможе забезпечити наочну візуалізацію даних та моделювання різних сценаріїв лікування. У той самий час широкомасштабне впровадження у медичні заклади вимагає уваги до питань безпеки даних, кіберзахисту та забезпечення анонімності. Подальша робота над оптимізацією алгоритмів шифрування, технічними заходами

для захисту конфіденційності та відповідністю міжнародним стандартам безпеки стане одним із ключових викликів.

Крім того, слід урахувати перспективи розширення застосування цієї технології для інших напрямів медицини. Методологічна основа, закладена у систему прогнозування діабету, може бути адаптована до виявлення ризиків розвитку серцево-судинних захворювань, онкологічних патологій або неврологічних розладів. Такий міждисциплінарний підхід сприятиме більшій універсальності системи, а також створенню єдиної інтегрованої платформи, здатної охопити комплексні аспекти стану здоров'я населення. Розширення тематичної сфери аналізу даних та включення додаткових типів медичної інформації, таких як геномні дані, стане підґрунтям для створення ще більш потужних інструментів підтримки клінічних рішень.

Зрештою, майбутні кроки повинні зосередитися на формуванні цілісної екосистеми, в якій системи прогнозування стануть центральним елементом персоналізованої медицини. Взаємодія з урядовими структурами, страховими компаніями, фармацевтичними концернами та медичними закладами уможливить розбудову гармонійного середовища, де технології штучного інтелекту, відповідальна обробка даних та передові методи аналітики об'єднуються для підвищення якості, доступності та ефективності медичної допомоги. Такий комплексний підхід не лише забезпечить подальше технологічне зростання, а й сприятиме реалізації глобальної місії — покращенню здоров'я та добробуту суспільства в цілому.

Можливості інтеграції з існуючими медичними інформаційними системами

Інтеграція розробленої системи автоматизованого прогнозування виявлення цукрового діабету з існуючими медичними інформаційними системами відкриває широкий спектр можливостей для покращення якості надання медичних послуг та підвищення ефективності управління даними про здоров'я пацієнтів. Поєднання

інноваційних методів машинного навчання з надійною інфраструктурою уже діючих електронних медичних карток, лабораторних інформаційних систем чи систем управління клінічними рішеннями сприяє не лише оперативному доступу до пацієнтських даних, але й глибокому аналізу показників, швидкому виявленню ризиків та прийняттю відповідних клінічних рішень.

Забезпечення двостороннього обміну даними між системами, що вже використовуються у лікарнях, поліклініках та приватних медичних закладах, дає змогу лікарям одразу отримувати попередження про підвищений ризик розвитку діабету без необхідності додаткових запитів чи ручної обробки інформації. Такий підхід значно пришвидшує процес діагностики, адже пацієнтські показники, зокрема рівні глюкози, наявність супутніх хронічних захворювань чи спадкові чинники, аналізуються в режимі реального часу, відображаючи картину загального стану здоров'я. Завдяки цьому інтеграція дозволяє не тільки виявляти хворобу на ранніх стадіях, але й оптимізувати стратегії лікування, підбираючи ефективні медикаменти та рекомендуючи зміни у способі життя залежно від індивідуальних потреб.

Особливо важливою стає масштабованість такої інтеграції, оскільки вона дозволяє системі працювати з різними джерелами даних та програмними інструментами, зберігаючи при цьому високу точність і надійність. Впровадження у хмарну інфраструктуру чи адаптація до стандартів електронного обміну даними, прийнятих у міжнародній медичній практиці, значно розширює географію використання системи, роблячи її доступною навіть у віддалених регіонах з обмеженими технологічними ресурсами. Це не лише позитивно впливає на забезпечення рівного доступу до медичних технологій, а й створює умови для глобального обміну досвідом та статистикою, що є основою для покращення загальної ситуації зі здоров'ям населення.

Крім того, інтеграція з існуючими інформаційними системами спрощує управління даними, робить можливим оперативне оновлення профілів пацієнтів та забезпечує безперервність їх історії хвороби. Інтегрованість відкриває шлях до

створення єдиних національних чи навіть міжнародних електронних реєстрів, що спрощує планування ресурсів, забезпечує контроль за поширенням захворювань та посилює ефективність науково-дослідних проєктів. Взаємодія з системами, які відповідають за страхові виплати, державні медичні програми чи управління електронними рецептами, дає змогу оптимізувати весь ланцюг надання медичної допомоги — від первинної діагностики до довгострокового ведення хронічних пацієнтів.

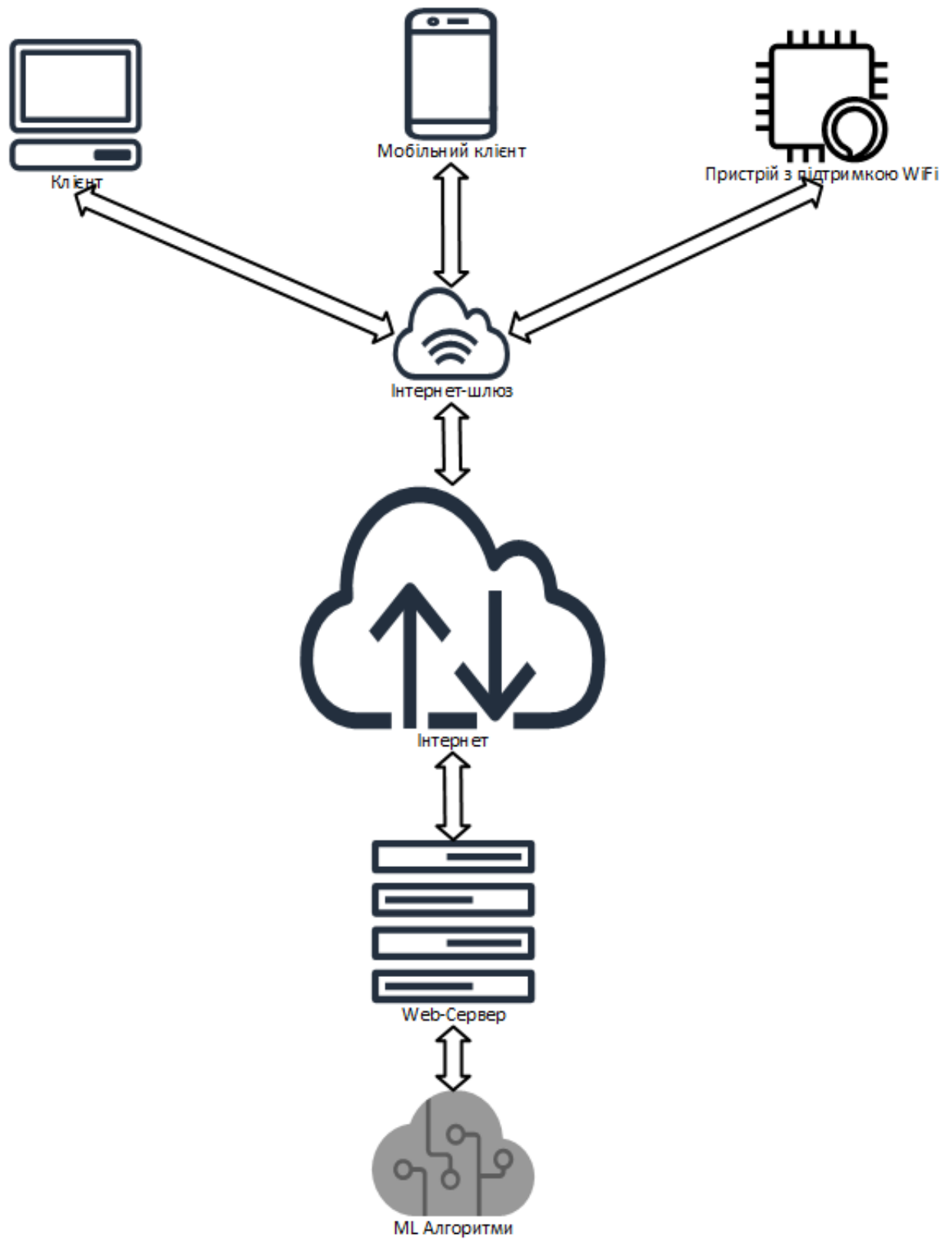
Таким чином, інтеграція автоматизованої системи прогнозування виявлення цукрового діабету з існуючими медичними інформаційними системами стає ключовим фактором ефективної трансформації галузі охорони здоров'я, дозволяючи максимально використовувати потенціал нових технологій для підвищення якості та доступності медичної допомоги.

СПИСОК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

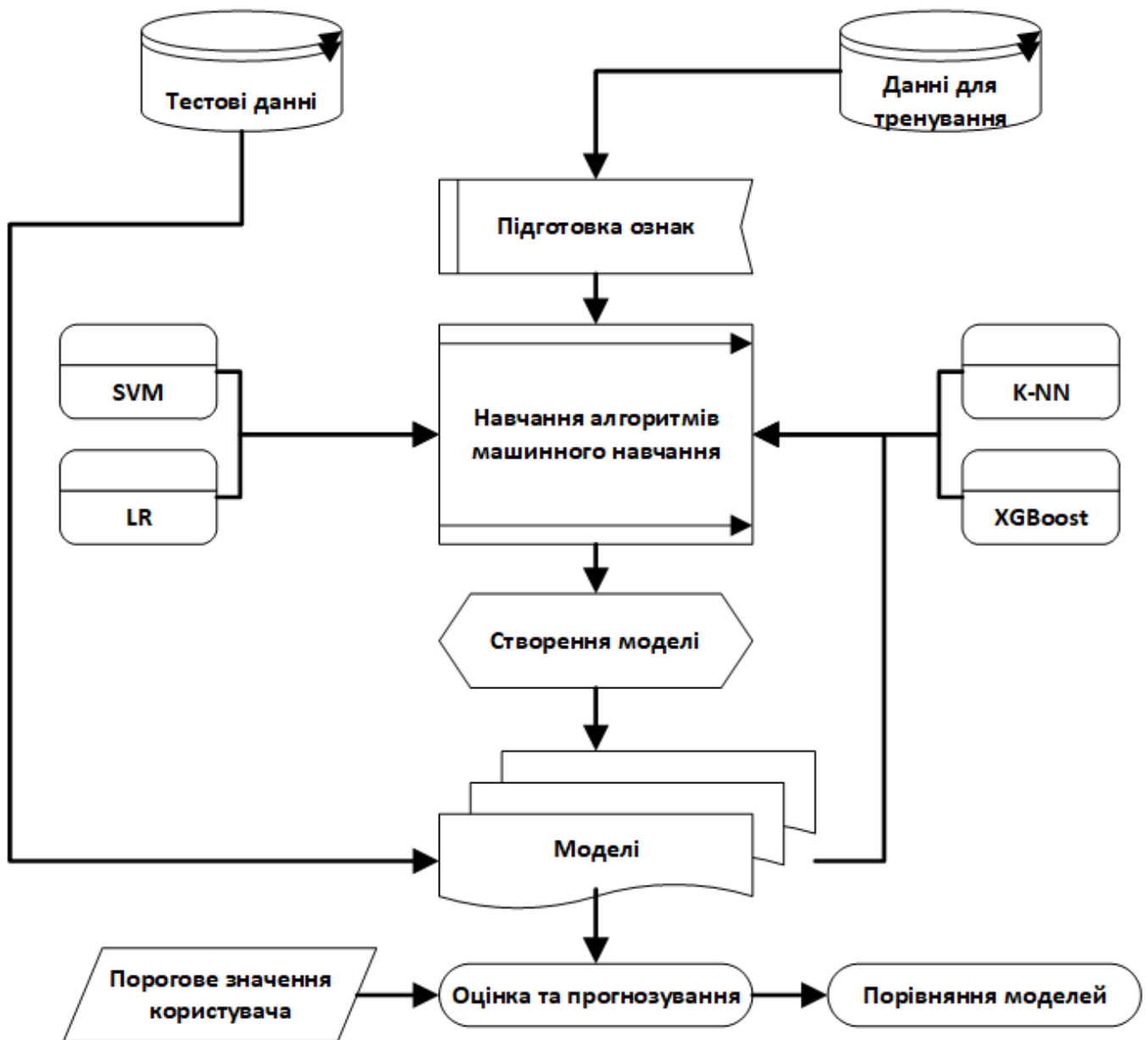
- [1] «https://uk.wikipedia.org/wiki/%D0%86%D1%81%D1%82%D0%BE%D1%80%D1%96%D1%8F_%D0%B4%D1%96%D0%B0%D0%B1%D0%B5%D1%82%D0%BE%D0%BB%D0%BE%D0%B3%D1%96%D1%97,» [Онлайновий].
- [2] «https://uk.wikipedia.org/wiki/%D0%A6%D1%83%D0%BA%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D0%B4%D1%96%D0%B0%D0%B1%D0%B5%D1%82,» [Онлайновий].
- [3] «<https://www.floris-sumy.com.ua/ua/cikave/novini/insulin-istoriya-vidkrittya-13-09-21.html>,» [Онлайновий].
- [4] «<https://pnu.edu.ua/depart/BioorganicChemistry/resource/file/W-09-11.pdf>,» [Онлайновий].
- [5] «https://www.rbc.ua/ukr/news/1-3-mln-ukraintsev-stradayut-saharnogo-diabeta-1542196774.html?utm_source=chatgpt.com,» [Онлайновий].
- [6] «Dr. Varma S.; Mitushi S. Diabetes Prediction using Machine Learning Techniques International Journal of Engineering Research & Technology (IJERT), September-2020. <http://www.ijert.org>,».
- [7] «Kaur, H.; Kumari, V. Predictive modelling and analytics for diabetes using a machine learning approach. Department of Computer Science and Engineering, School of Engineering Sciences and Technology, Jamia Hamdard, New Delhi, India, 19 December 2018. h».
- [8] «Deberneh, H.M.; Kim, I. Prediction of Type 2 Diabetes Based on Machine Learning Algorithm. Int. J. Environ. Res. Public Health 2021, 18, 3317. <https://doi.org/10.3390/ijerph18063317>,».
- [9] «Dritsas, E.; Trigka, M. Data-Driven Machine-Learning Methods for Diabetes Risk Prediction. Sensors 2022, 22, 5304. <https://doi.org/10.3390/s22145304>,».
- [10] «<https://alpha-medical.com.ua/uk/diagnostyka-ta-likuvannya-czukrovogo-diabetu-novi-pidhody-ta-tehnologiyi/>,» [Онлайновий].
- [11] «<https://invivo.ua/uk/blog-detail/proyavlenie-saharnogo-diabeta/>,» [Онлайновий].
- [12] «https://elartu.tntu.edu.ua/bitstream/lib/39410/1/Baran_S_O.pdf,» [Онлайновий].
- [13] «<https://ztu.edu.ua/site/files/1/docs/%D0%9D%D0%B0%D1%83%D0%BA%D0%B0/%D0%90%D1%81%D0%BF%D1%96%D1%80%D0%B0%D0%BD%D1%82%D1%83%D1%80%D0%B0%20%D1%82%D0%B0%20%D0%B4%D0%BE%D0%BA%D1%82%D0%BE%D1%80%D0%B0%D0%BD%D1%82%D1%83%D1%80%D0%B0/%D0%A0%D0%B0%D0%B7%D0%BE%D0%B2>,» [Онлайновий].
- [14] «<https://a4.com.ua/rozrobka-program-avtomatizacii-dlja-analizu-danih-ta-prognozuvannja/>,» [Онлайновий].
- [15] «<http://dspace.wunu.edu.ua/bitstream/316497/47261/1/%D0%93%D1%80%D0%B8%D0%BD%D1%8C%D0%BA%D1%96%D0%B2.pdf>,» [Онлайновий].
- [16] «<https://ela.kpi.ua/server/api/core/bitstreams/9f04330d-4748-4198-a60a-e481a85e0112/content>,» [Онлайновий].
- [17] «<https://gmdhsoftware.com/ua/ai-ml-forecasting-software/>,» [Онлайновий].
- [18] «<https://bimedis.net/latest-news/browse/778/suchasni-tehnologiyi-u-medicini-shtuchnij-intelekt>,» [Онлайновий].
- [19] «<https://umj.com.ua/uk/publikatsia-241221-rozvitok-shtuchnogo-intelektu-v-suchasnij-meditcini>,» [Онлайновий].
- [20] «<https://uifuture.org/publications/25217-medicine/>,» [Онлайновий].
- [21] «<https://uk.shaip.com/blog/real-world-applications-of-machine-learning-in-healthcare/>,» [Онлайновий].
- [22] «<https://med-expert.com.ua/news-uk/rol-data-science-ta-ai-u-transformaciyi-ohoroni-zdorovya>,» [Онлайновий].
- [23] «<https://repo.knmu.edu.ua/server/api/core/bitstreams/4558538d-c20f-4feb-877b-202fd1fdd662/content>,» [Онлайновий].
- [24] «<https://iqmed.com.ua/tsukrovyy-diabet-info/>,» [Онлайновий].
- [25] «<https://www.kaggle.com/code/fareedalwanwar/logistic-regression/input>,» [Онлайновий].
- [26] «https://moodle.znu.edu.ua/pluginfile.php/1194219/mod_resource/content/1/%D0%9B%D0%B5%D0%BA%D1%86%D1%96%D1%8F%20%D0%B4%D0%BE%20%D0%BC%D0%BE%D0%B4%D1%83%D0%BB%D1%8F%204.pdf,» [Онлайновий].
- [27] «[https://dspace.uzhnu.edu.ua/jspui/bitstream/lib/53297/1/%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%D1%87%D0%BA%D0%B0%20\(3\).pdf](https://dspace.uzhnu.edu.ua/jspui/bitstream/lib/53297/1/%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%D1%87%D0%BA%D0%B0%20(3).pdf),» [Онлайновий].
- [28] «<https://www.zfort.com.ua/blog/yak-pidgotuvati-nabir-danikh-mashinnogo-navchannya-za-7-krokv/>,» [Онлайновий].
- [29] «<https://kharkov-future.com.ua/uk/articles-yak-rozroblyayetsya-mashynne-navchannya-klyuchovi-etapy>,» [Онлайновий].
- [30] «<https://blog.plag.ai/uk/machine-learning-overview>,» [Онлайновий].
- [31] «<https://mindthegraph.com/blog/uk/machine-learning-in-science/>,» [Онлайновий].

- [32] «<https://www.it.ua/knowledge-base/technology-innovation/machine-learning>,» [Онлайновий].
- [33] «<https://devzone.org.ua/post/10-alhorytmiv-mashynnoho-navchannia-na-primyakh-z-realnoho-svitu>,» [Онлайновий].
- [34] «<https://pnn.com.ua/ua/blog/detail/lightweight-and-flexible-flask-fundamentals-for-software-web-development>,» [Онлайновий].
- [35] «<https://uk.sharpcodeblog.com/blog/creating-restful-apis-with-python-and-flask>,» [Онлайновий].
- [36] «<https://devzone.org.ua/post/stvorennia-servisu-dlia-zberihannia-fayliv-z-flask-rethinkdb-ta-vuejs-ch1>,» [Онлайновий].
- [37] «https://ela.kpi.ua/bitstream/123456789/43983/1/Chukhlib_bakalavr.pdf](https://ela.kpi.ua/bitstream/123456789/43983/1/Chukhlib_bakalavr.pdf),» [Онлайновий].
- [38] «<https://dan-it.com.ua/uk/blog/shho-take-git-ta-github-bazovi-nalashtuvannya-i-yak-z-nymy-praczuvaty>,» [Онлайновий].
- [39] «<https://git-scm.com/book/uk/v2/%D0%9D%D0%B0%D0%BB%D0%B0%D1%88%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F-Git-%D0%9A%D0%BE%D0%BD%D1%84%D1%96%D0%B3%D1%83%D1%80%D0%B0%D1%86%D1%96%D1%8F-Git>,» [Онлайновий].
- [40] «https://ir.nmu.org.ua/jspui/bitstream/123456789/159001/1/%D0%9F%D0%97_%D0%A1%D0%BE%D0%BB%D0%B4%D0%B0%D1%82%D0%BE%D0%B2_123-18%D1%81%D0%BA-1.pdf,» [Онлайновий].
- [41] «https://uk.wikipedia.org/wiki/Visual_Studio_Code,» [Онлайновий].
- [42] «<https://learn.microsoft.com/uk-ua/power-pages/configure/vs-code-extension>,» [Онлайновий].
- [43] «<https://science.lpnu.ua/sites/default/files/journal-paper/2024/nov/36464/241164maket-169-192.pdf>,» [Онлайновий].
- [44] «<https://science.lpnu.ua/sites/default/files/journal-paper/2024/aug/35679/maket2402951-294-325.pdf>,» [Онлайновий].
- [45] «<https://devzone.org.ua/post/10-alhorytmiv-mashynnoho-navchannia-na-primyakh-z-realnoho-svitu>,» [Онлайновий].
- [46] «<https://www.dstu.dp.ua/Portal/Data/74/72/3st13-17.pdf>,» [Онлайновий].
- [47] «<https://itwiki.dev/data-science/ml-reference/ml-glossary/xgboost>,» [Онлайновий].
- [48] «<https://ekmair.ukma.edu.ua/bitstreams/c315b222-4b7e-44b6-a306-06ebb90b7d5b/download>,» [Онлайновий].
- [49] «<https://ami-ejournal.cdu.edu.ua/article/download/4607/4853/11026>,» [Онлайновий].
- [50] «https://www.researchgate.net/publication/375040651_METODI_BUSTINGOVOGO_MASINNOGO_NAVCANNA_DLA_NESTACIONARNIH_CASOVIH_RADIV,» [Онлайновий].
- [51] «<https://itwiki.dev/data-science/ml-reference/ml-glossary/knn-models>,» [Онлайновий].
- [52] «<https://www.unite.ai/uk/a-quick-guide-to-knn-algorithm>,» [Онлайновий].
- [53] «https://essuir.sumdu.edu.ua/bitstream/123456789/92711/1/Kharchenko_mashynne_navchannia.pdf,» [Онлайновий].
- [54] «<https://foxminded.ua/freimvorky-python/>,» [Онлайновий].
- [55] «<https://alimbekov.com/machine-learning-flask-rest-api/>,» [Онлайновий].
- [56] «<https://ela.kpi.ua/bitstreams/d211d7a3-2b2e-4431-9bd8-b59f108ada6f/download>,» [Онлайновий].
- [57] «<https://foxminded.ua/vue-js/>,» [Онлайновий].
- [58] «<https://ua.vuejs.org/guide/introduction.html>,» [Онлайновий].
- [59] «<https://uk.wikipedia.org/wiki/HTTP>,» [Онлайновий].
- [60] «<https://goit.global.ua/articles/rest-api-shcho-tse-iak-pratsiuie-perevahy-i-primyary>,» [Онлайновий].
- [61] «https://www.espressif.com/sites/default/files/documentation/esp8266-technical_reference_en.pdf,» [Онлайновий].
- [62] «<https://alterstrategy.com/2021/05/19/connecting-the-esp8266-module-with-nodemcu-lua-firmware-to-home-assistant-server-by-http-2>,» [Онлайновий].
- [63] к. Ю. Б. С. В. Підпригора, «Автоматизована система прогнозування виявлення цукрового діабету на основі медичних даних пацієнтів», 2024.
- [64] «<https://empendium.com/ua/chapter/B27.13.1>,» [Онлайновий].
- [65] «http://om.univ.kiev.ua/users_upload/15/upload/file/pr_lecture_03.pdf,» [Онлайновий].
- [66] «<https://ekmair.ukma.edu.ua/server/api/core/bitstreams/1f16a596-e417-4cae-b2ba-ec9f6c2d6c67/content>,» [Онлайновий].
- [67] «<https://www.youtube.com/watch?v=YYOuTkqey7E>,» [Онлайновий].

Додаток А

**Функціональна схема автоматизованої системи прогнозування
виявлення цукрового діабету**

Додаток Б

Блок-схема для оцінки роботи алгоритму прогнозування виявлення цукрового діабету на основі медичних даних пацієнтів

Додаток В

Блок-схема алгоритму вибору найточнішого передбачення

