МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри інженерії програмного забезпечення, канд. техн. наук, доцент,

_____Є. О. Давиденко

«____»_____2024 р.

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

ІНФОРМАЦІЙНА СИСТЕМА МОНІТОРИНГУ ЙМОВІРНОСТІ ВИНИКНЕННЯ ЗАХВОРЮВАНЬ НА ОСНОВІ АНАЛІЗУ ФАКТОРІВ РИЗИКУ

Спеціальність «Інженерія програмного забезпечення»

121 – КРМ.1 − 608м.21610801

Здобувач _____В. О. Баришніков

підпис

«___» _____ 2024 р.

Керівник д-р. техн. наук, професор кафедри інженерії програмного забезпечення

_____А. В. Швед

підпис

«___» _____ 2024 р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра інженерії програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри інженерії програмного

забезпечення, канд.техн.наук, доцент,

_____Є. О. Давиденко

«___» _____ 2023 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи магістра

Видано студенту групи 608м факультету комп'ютерних наук

Баришнікову Вадиму Олександровичу.
(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

«Інформаційна система моніторингу ймовірності виникнення захворювань на основі аналізу факторів ризику».

Затверджена наказом по ЧНУ від «10» листопада 2023 р. № 234

2. Строк представлення кваліфікаційної роботи «__» _____ 2024 р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні

Вхідні дані до роботи − функціональні та нефункціональні вимоги до програмного забезпечення, якісна і / або кількісна оцінка факторів ризику виникнення захворювання. Результат – функціонуючий застосунок, що визначає ймовірність виникнення захворювання у пацієнта.

4. Перелік питань, що підлягають розробці:

- аналіз предметної сфери розробки програмного забезпечення інформаційних систем в ґалузі соціальної медицини;
- аналіз факторів ризику ймовірності виникнення захворювань;

– моделювання та проєктування програмного забезпечення інформаційної системи;
– програмна реалізація інформаційної системи.

5. Перелік графічних матеріалів:

Презентація.

Керівник роботи д-р. техн. наук, професор Швед Альона Володимирівна
(посада, прізвище, ім'я, по батькові)

_____
(підпис)

Завдання прийнято до виконання

Баришніков Вадим Олександрович
(прізвище, ім'я, по батькові студента)

_____
(підпис)

Дата видачі завдання «___» _____ 2023 р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: «Інформаційна система моніторингу ймовірності виникнення захворювань на основі аналізу факторів ризику» .

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КРМ	10.11.2023	20.11.2023	Виконано
2.	Огляд літератури за темою роботи	21.11.2023	28.11.2023	Виконано
3.	Складання календарного плану КРМ	28.11.2023	29.11.2023	Виконано
4.	Аналіз предметної області	01.12.2023	15.12.2023	Виконано
5.	Розробка проєктних рішень	16.12.2023	20.12.2023	Виконано
6.	Моделювання та конструювання ПЗ	21.12.2023	27.12.2023	Виконано
7.	Кодування, тестування та апробація розробленого ПЗ, аналіз результатів тестування, розробка керівництва користувача	02.01.2024	30.01.2024	Виконано
8.	Відгук керівника КРМ	05.02.2024	05.02.2024	Виконано
9.	Оформлення КРМ та презентації	06.02.2024	07.02.2024	Виконано
10.	Попередній захист	08.02.2024	08.02.2024	Виконано
11.	Рецензування	10.02.2024	15.02.2024	Виконано
12.	Завершення оформлення КРМ та презентації	16.02.2024	20.02.2024	Виконано
13.	Захист кваліфікаційної роботи	26.02.2024	27.02.2024	Виконано

Розробив студент Баришніков Вадим Олександрович _____
(прізвище, ім'я, по батькові студента) (*підпис*)

«___» _____ 2023 р.

Керівник роботи д-р. техн. наук, професор Швед А. В. _____
(посада, прізвище, ім'я, по батькові) (*підпис*)

«___» _____ 2023 р.

АНОТАЦІЯ

до кваліфікаційної роботи магістра

«Інформаційна система моніторингу ймовірності виникнення захворювань на основі аналізу факторів ризику»

Студент 608м гр.: Баришніков Вадим Олександрович

Керівник: д-р. техн. наук, професор Швед А. В.

Інформаційні технології використовуються в сучасній медицині для вдосконалення точності діагностування хвороб. Алгоритми машинного навчання та штучний інтелект надають нові можливості та інструменти для обробки великих масивів даних.

Об’єкт кваліфікаційної роботи – процес визначення ймовірності виникнення захворювання. Предмет кваліфікаційної роботи – методи та засоби створення моделі для визначення ймовірності виникнення захворювання. Метою кваліфікаційної роботи є вдосконалення процесу профілактики захворювань за рахунок використання алгоритмів машинного навчання.

В першому розділі представлено аналіз предметної сфери розробки програмного забезпечення в ґалузі соціальної медицини, здійснено специфікацію вимог та досліджено існуючі аналоги. У другому розділі обрано набір даних та алгоритм машинного навчання. В ході розробки третього розділу виконано проєктування та моделювання системи. В четвертому розділі представлено основні етапи програмної реалізації медичної інформаційної системи.

В результаті виконання кваліфікаційної роботи магістра реалізовано медичну інформаційну систему.

КРМ викладена на 46 сторінок, вона містить 4 розділи, 25 рисунків, 10 таблиць, 18 джерел в переліку посилань

Ключові слова: медична інформаційна система, машинне навчання, дерева рішень, Python, Telegram-бот.

ABSTRACT

of the Master’s Thesis

«Information system for monitoring the probability of disease occurrence based on the analysis of risk factors»

Student of group 608m: Baryshnikov Vadym Oleksandrovych

Supervisor: Dr. Sci. (Engin.), Professor Shved A. V.

Information technologies are used in modern medicine to improve the accuracy of diagnosing diseases. Machine learning algorithms and artificial intelligence provide new opportunities and tools for processing large amounts of data.

The object of the qualification work is the process of determining the probability of the occurrence of the disease. The subject of the qualification work is methods and means of creating a model for determining the probability of disease occurrence. The purpose of the qualification work is to improve the process of disease prevention through the use of machine learning algorithms.

In the first chapter, an analysis of the subject area of software development in the field of social medicine is presented, requirements are specified, and existing alternatives are investigated. In the second section, a data set and a machine learning algorithm are selected. During the development of the third section, the design and modeling of the system was performed. The fourth chapter presents the main stages of software implementation of the medical information system.

A medical information system was implemented as a result of the master's qualification work.

The work is 46 pages long and includes 4 sections, 25 illustrations, 10 tables, and 18 sources.

Keywords: *hospital information system, machine learning, decision tree, Python, Telegram-bot, scikit-learn.*

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

БД – база даних

ЕСМД – експертна система медичної діагностики

КРМ – кваліфікаційна робота магістра

МІС – медична інформаційна система

ПЗ – програмне забезпечення

ШІ – штучний інтелект

UML – Unified Modeling Language

ВСТУП

Сучасна медицина нерозривно пов’язана з інформаційними технологіями. Застосування програмного забезпечення для вдосконалення точності діагностування хвороб та є одним з найважливіших напрямків інновацій. Штучний інтелект та алгоритми машинного навчання надають нові можливості та інструменти для здійснення досліджень та обробки великих масивів даних. В той самий час медицина вирізняється вимогливістю до точності та науковості використаних розрахунків та алгоритмів. Більшість сучасних досліджень неможливі без окислювальної техніки.

Об’єктом є процес визначення ймовірності виникнення захворювання.

Предметом є методи та засоби створення моделі для визначення ймовірності виникнення захворювання.

Метою кваліфікаційної роботи є вдосконалення процесу профілактики захворювань за рахунок автоматизації процесу оцінки ймовірності виникнення захворювання на основі аналізу факторів ризику, що його обумовлюють методами машинного навчання.

Для досягнення поставленої мети необхідно виконати наступні *завдання*:

1) Аналіз предметної сфери розробки програмного забезпечення інформаційних систем в ґалузі соціальної медицини.
2) Аналіз існуючих аналогів та визначення їх переваг і недоліків.
3) Вибір набору даних.
4) Специфікація вимог.
5) Моделювання системи.
6) Вибір алгоритму машинного навчання.
7) Програмна реалізація інформаційної системи.

Актуальність теми визначена тим що вдосконалення процесу профілактики захворювань має важливу роль в охороні здоров’я. Запобігання хворобам набагато ефективніше за лікування запущених випадків. Технології для аналізу даних дозволяють обробляти великі медичні датасети для прогнозування, діагностики та досліджень, що має позитивний вплив на розвиток та вдосконалення медицини.

1 АНАЛІЗ ПРЕДМЕТНОЇ СФЕРИ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНИХ СИСТЕМ В ҐАЛУЗІ СОЦІАЛЬНОЇ МЕДИЦИНИ

1.1 Опис предметної області розробки інформаційних систем в галузі соціальної медицини

В сучасному світі медицина тісно пов’язана з інформаційними технологіями. Наприклад, біотехнології та біостатистика передбачають інтеграцію біології, інформаційних технологій та інженерії в межах одного проєкту або дослідження. Здійснення розрахунків та моделювання без використання обчислювальної техніки неможливе в ґалузі генетики, епідеміології та багатьох інших напрямків, які пов’язані з обробкою великих обсягів даних. Застосування інформаційних технологій в медицині забезпечує:

1) Створення нового або вдосконалення існуючого обладнання. В даному випадку передбачається розробка апаратного і програмного забезпечення. Точність та функціональність сучасного медичного обладнання постійно покращується завдяки інноваціям в сфері інформатики.

2) Оптимізоване збереження медичних даних. Завдяки збереженим даним вчені можуть отримати необхідні для дослідження датасети не лише цілеспрямовано збираючи нові дані, а і звертаючись до існуючих «архівів». Цифрова база даних спрощує співпрацю та дозволяє швидко обмінюватися інформацією на відміну від громіздких та невпорядкованих паперових документів. Big Data робить дослідження більш прозорими.

3) Аналіз даних. При підготовці до експерименту чи при розробці гіпотез вчені спираються на статистику про історії хвороб та інші дані про пацієнтів. Побудова математичних моделей допомагає проводити дослідження, прогнозувати результати лікування та перевіряти ефективність ліків. Big Data – це масив структурованої або неструктурованої інформації, яка має великий обсяг. Також Big Data включає в себе інструменти, підходи та методи зберігання «великих» даних [1].

4) IoMT (Internet of Medical Things). Як спеціалізоване обладнання в лікарні, так і звичайний смарт-годинник збирають та обробляють дані про стан здоров'я в режимі реального часу.

5) Управління та логістика. Спеціалізоване програмне забезпечення дозволяє управляти персоналом, інвентаризацією, документообігом тощо. Також існують системи які призначені для запису на прийом та отримання направлення онлайн (наприклад, «Helsi»). Чат-боти дозволяють оперативно надавати відповіді на найпоширеніші «організаційні» питання від пацієнтів та у такий спосіб заощаджують час персоналу лікарні.

Дана кваліфікаційна робота присвячена використанню аналізу даних та алгоритмів машинного навчання для моніторингу ймовірності виникнення захворювань на основі факторів ризику. Отже, має бути розроблена **експертна система медичної діагностики (ЕСМД)**. Метою ЕСМД є постановка діагнозу на основі даних про стан здоров'я пацієнта.

Починаючи з 1950-х вчені зацікавилися застосуванням обчислювальної техніки для здійснення регресійного аналізу в сфері медичних досліджень. В 1959 році в журналі «Science» була опублікована стаття під назвою «Reasoning Foundations of Medical Diagnosis» (Robert S. Ledley and Lee B. Lusted). Вказана робота містила математичну модель процесу діагностики захворювань. Більшість концепцій та термінів з даної статті досі актуальні, отже дату її публікації можна вважати початком існування біостатистики.

Однією з перших ЕСМД була INTERNIST-1, створена в 1970-х роках в Університеті Піттсбурга. Дана система спиралася лише на лінійну та логістичну регресію, часто неправильно діагностувала захворювання і вимагала великої кількості часу для постановки діагнозу. Ситуація ускладнювалася відсутністю графічного інтерфейсу та недоступністю ЕОМ для більшості лікарень, шпиталів та університетів. Отже, на той момент ЕСМД не набули популярність. У порівнянні з цим застосунком лікар був набагато кращою альтернативою [2].

В 1990-х почало з'являтися програмне забезпечення яке діагностувало захворювання якісніше та точніше за лікарів. Це стало можливо завдяки складним

алгоритмам машинного навчання. Часто в ЕСМД використовуються саме нейронні мережам. Нейронні мережі (або штучні нейронні мережі) – це різновид алгоритмів глибокого навчання, принцип дії яких полягає в імітації нейронних мереж людського мозку [3]. В 1990-х прогрес відбувався не лише в сфері розробки алгоритмів та математичних моделей, а і в ґалузі апаратного забезпечення. Комп'ютери на той момент набули значного поширення, а розробка доступного для пересічного користувача графічного інтерфейсу зробила їх використання повсюдним.

На даний момент сфера біостатистики та data science для медицини активно розвивається. Дослідження та інновації в сфері охорони здоров'я мають велику цінність для суспільства, тому даний напрямок є одним з найбільш важливих та актуальних. Але робота з медичними даними є однією з найскладніших та найбільш наукомістких галузей в data science. Справжні проєкти в даній сфері передбачають залучення великої кількості фахівців, які мають різну спеціалізації. Серед цих фахівців обов'язково мають бути:

1) Спеціалісти із захисту даних. Інформація про стан здоров'я є прикладом особистої інформації, оприлюднення або викрадення якої поставить під загрозу приватність та добробут людей, яких ці дані стосуються. Дотримання «лікарської таємниці» вимагається на законодавчому рівні у переважній більшості країн [4]. Саме тому належний захист даних є надзвичайно важливим при роботі з медичними даними.

2) Data Scientists, інженери програмного забезпечення тощо. Дані спеціалісти створюють математичні моделі та програмне забезпечення. Саме в ході аналізу даних та застосування машинного навчання можливо робити певні висновки на основі даних. Інженери та програмісти повинні мати певне уявлення про предмет дослідження, але у випадку медицини важливо враховувати відсутність належної медичної освіти. Співпраця з фахівцями є запорукою розробки якісного та безпечного програмного забезпечення для медичних цілей.

3) Фахівці в ґалузі медицини, біології або хімії. Лише медики мають право підтверджувати або спростовувати здійснену інформаційною системою

діагностику. На етапі тестування та відлагодження важливо враховувати поради та зауваження цих фахівців заради покращення якості ПЗ.

Попри значні успіхи штучного інтелекту (ШІ) для діагностики захворювань варто все одно залишити остаточне рішення стосовно діагнозу та планування подальшого лікування медикам. Інформаційна система має бути допоміжним засобом для лікаря, а не його заміною.

1.2 Класифікація медичних інформаційних систем

Існують різні підходи до класифікації медичних інформаційних систем (МІС). Якщо у системі збір даних відбувається без участі людини, то така система називається автоматичною. Якщо людина певною мірою бере участь у процесі роботи системи, то тоді така система називається автоматизованою.

Також система може оперувати даними або знаннями (експертна система). Наприклад, інформаційна система для запису в чергу до лікаря оперує даними, оскільки дата, час та інші аспекти відвідування лікарні не спираються на специфічні знання в ґалузі медицини. А от система яка здійснює діагностування на основі даних про стан здоров’я пацієнта є саме експертною системою.

В залежності від типу поставленого завдання МІС можна поділити на:

1. *Інформаційно-довідкові.* Найпоширеніший тип, який допомагає автоматизувати документообіг та пришвидшити процес пошуку необхідної інформації (з довідників, каталогів тощо).

2. *Інформаційно-логічні.* Дозволяють отримати нові знання в результаті прогнозування, моніторингу або діагностування.

3. *Автоматизовані системи управління.* Використовуються для прийняття керуючих рішень.

Медичні інформаційні системи можуть використовуватися на різних рівнях. Користувачами можуть бути як лікарі певної клініки, так і державні заклади та органи управління. Отже, область застосування залежить від поставленої задачі.

1.3 Аналіз сучасних медичних інформаційних систем

Аналогічне програмне забезпечення належить до двох окремих класів: діагностичні ІС та лікувально-діагностичні ІС. Крім того, програмне забезпечення може бути орієнтоване на лікарів (фахівців) або на пацієнтів (нефахівців).

Існують спеціалізовані програми (наприклад, ACDSee, AquilionONE, Infinix CS-I, Horizon SE та XIDF-QCA801), які використовуються в комплексах променевої та ультразвукової діагностики. Також існує програмне забезпечення для ультразвукової сонографії, комп'ютерної та магнітно-резонансної томографії, ангіографії, ехосонографії, ендоскопії тощо. Інтерфейс даних програм виконує калібрування детектора, захоплення та виведення зображення. Для підвищення інформативності зображень використовуються фільтри для управління контрастністю, виділення контурів, селективного підсилення або пригнічення сигналу. На основі створених математичних моделей розробляються експертні системи (наприклад, Гастрограф 1T, Stimul тощо), які обробляють отримані дані.

Лікувально-діагностичні ІС виконують аналіз даних з метою прийняття рішень під час вирішення проблеми на лікувальному або діагностичному етапі. У сучасній медицині процеси діагностування характеризуються високою складністю. Спостерігається тенденція ускладнення процесів прийняття рішень у зв'язку з розвитком процесів інтеграції та глобалізації в медичній практиці. Деякі великі медичні центри намагаються розробити «інтелектуальні» програми, що дають змогу встановити діагноз, звузити коло захворювань або визначити мінімальний набір діагностичних тестів (наприклад, ІС для постановки діагнозів Internist, BPLab, SpeseLabs Medical, Meditech, A&D, Omron, CardioVita тощо) [5].

Наведені вище приклади програмного забезпечення передбачають наявність специфічних інтерфейсів (томограф, комплекс для УЗД тощо) для «читання»даних про пацієнта. Їх розробка та тестування передбачали співпрацю з фахівцями в області медицини та потужне фінансування. Крім того, це спеціалізоване програмне забезпечення яким користуються лише фахівці.

Серед найбільш популярних «користувацьких» застосунків варто назвати «Apple Health». Даний застосунок підтримує як введення даних вручну, так і

синхронізацію з Apple Watch для автоматизованого читання даних про тривалість сну, температуру тіла, тиск тощо. На рисунках 1.1 та 1.2 наведено зразки інтерфейсу.

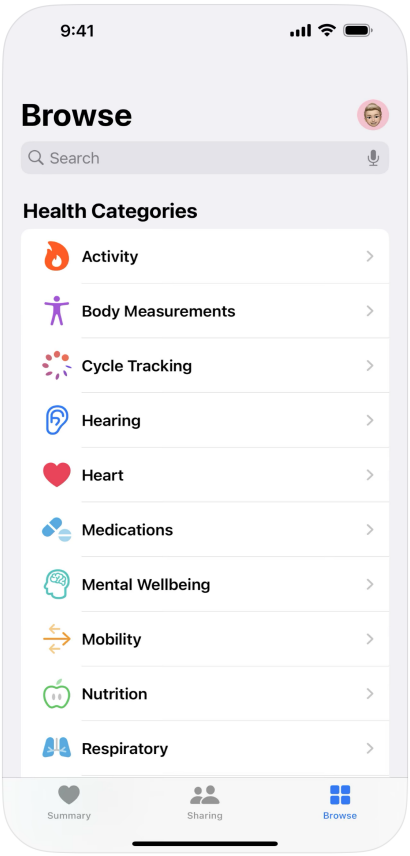


Рисунок 1.1 – Функції застосунку «Health»

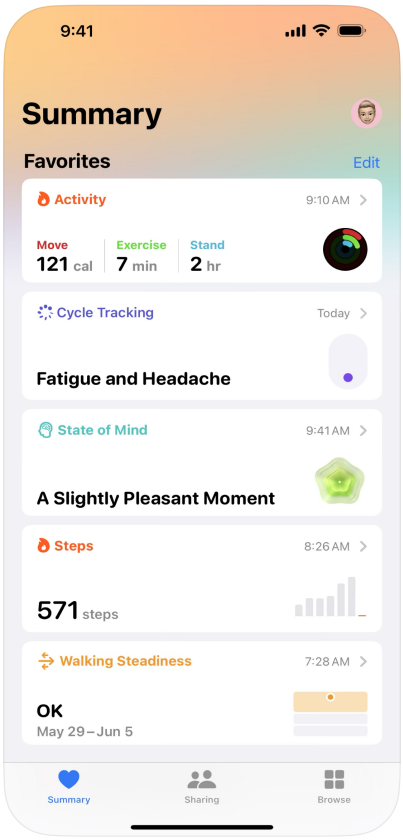


Рисунок 1.2 – Приклад інтерфейсу «Health»

Користувач отримує інтерактивні звіти про стан здоров'я, візуалізацію цих даних та може досліджувати тренди у своїх звичках та змінах стану. Даними звітами можна ділитися, наприклад, з лікарем. Наявність систематизованої інформації про зміни у стані здоров'я необхідна під час постановки діагнозу.

З появою смарт-годинників подібне програмне забезпечення стало актуальним та затребуваним. Його застосування вимагає все менше зусиль, оскільки більшість даних автоматично вводиться зі спеціалізованого пристрою, а не вручну. Інша подібна платформа – кросплатформний застосунок «Google Fit». Приклад даного застосунку та основні його функції представлено на рисунку 1.3.

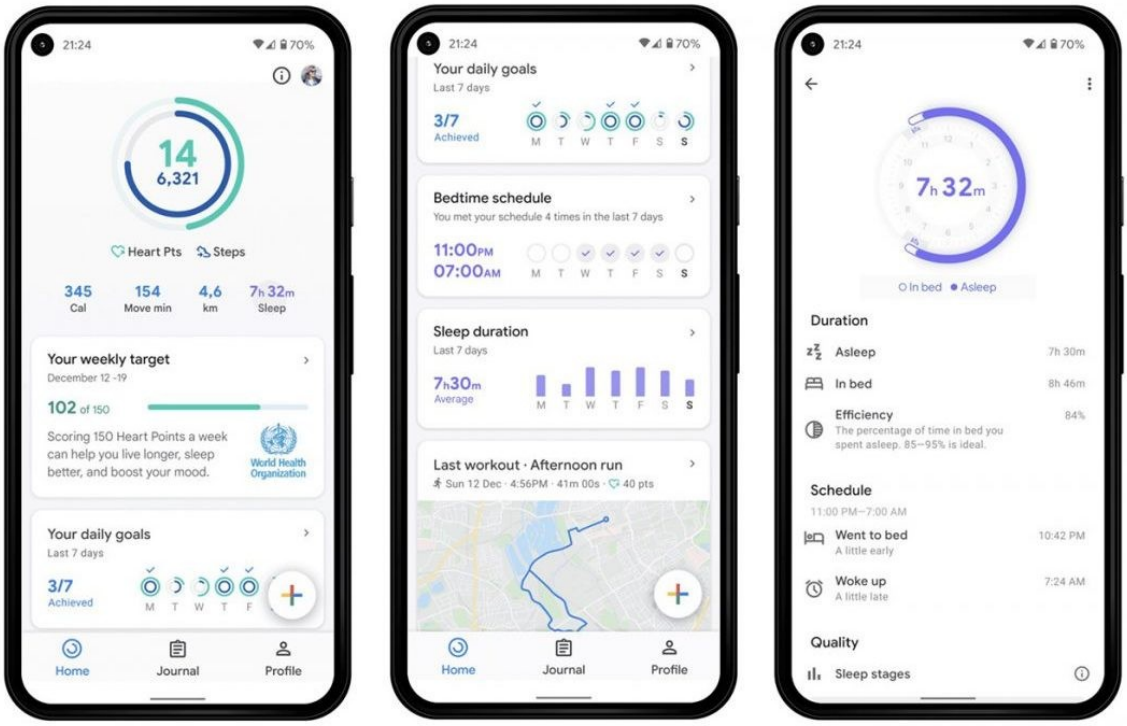


Рисунок 1.3 – Інтерфейс застосунку «Google Fit»

Всі наведені вище системи мають досить широку спеціалізацію. Також існують окремі застосунки для відстеження фітнес-цілей або прогресу у лікуванні діабету, онкології та інших хвороб. Також актуальним напрямком є «асистенти», які нагадують про необхідність прийняти пігулки, перевірити рівень цукру в крові чи звернутися до лікаря.

Також існують застосунки для підтримки ментального здоров’я. Формат мобільного застосунку є найбільш популярним не випадково. Саме смартфон у сучасному світі став однією з найбільш особистих речей, яка завжди поряд з людиною і часто виконує функцію щоденника.Чат-боти є навіть більш ефективним шляхом взаємодії з користувачем, оскільки за визначенням вони кросплатформні та мають знайомий інтерфейс. Одним з найяскравіших прикладів є український чат-бот першої психологічної допомоги «Друг. Перша допомога» [6]. Приклад інтерфейсу вказаного чат-боту наведено на рисунку 1.4.

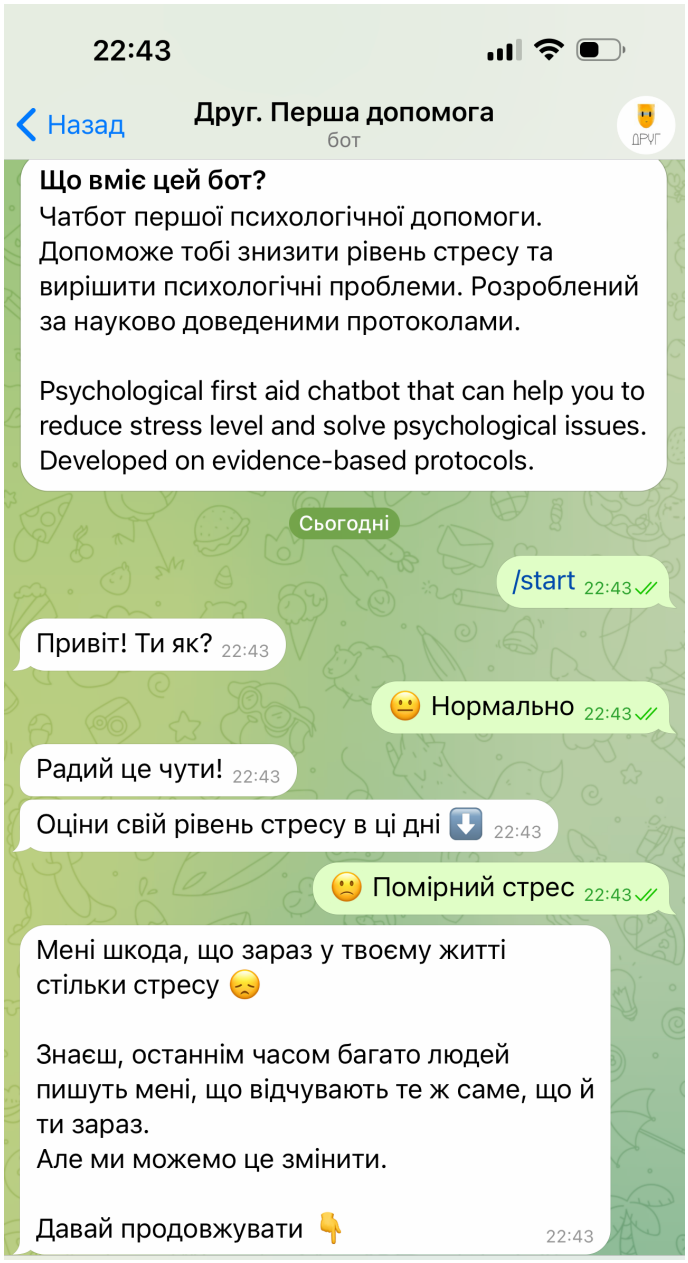


Рисунок 1.4 – Приклад інтерфейсу чат-боту «Друг. Перша допомога»

Отже, чат-бот – це найзручніший та найбільш універсальний формат користувацького застосунку для підтримки здоров’я.

Аналіз природних мов та штучний інтелект робить застосунки для моніторингу стану здоров’я гнучкими та інформативними. Штучний інтелект у застосунки впроваджується за допомогою машинного навчання.

Машинне навчання є галуззю штучного інтелекту (ШІ) та комп’ютерних наук, яка фокусується на використанні даних та алгоритмів для імітації процесу навчання, який здійснюється людським мозком [7]. В результаті машинного

навчання створюється модель, яка здатна роботи прогнози. Машинне навчання використовується в різних галузях людської діяльності, наприклад, у фінтеху, медицині, освіті та соціальних мережах. Розроблена модель має розпізнати загальні патерни та закономірності, але не надто прив’язуватись до наданого набору даних (рис. 1.5). І занадто проста модель, і занадто точна модель (overfitting) призводять до помилкових прогнозів.

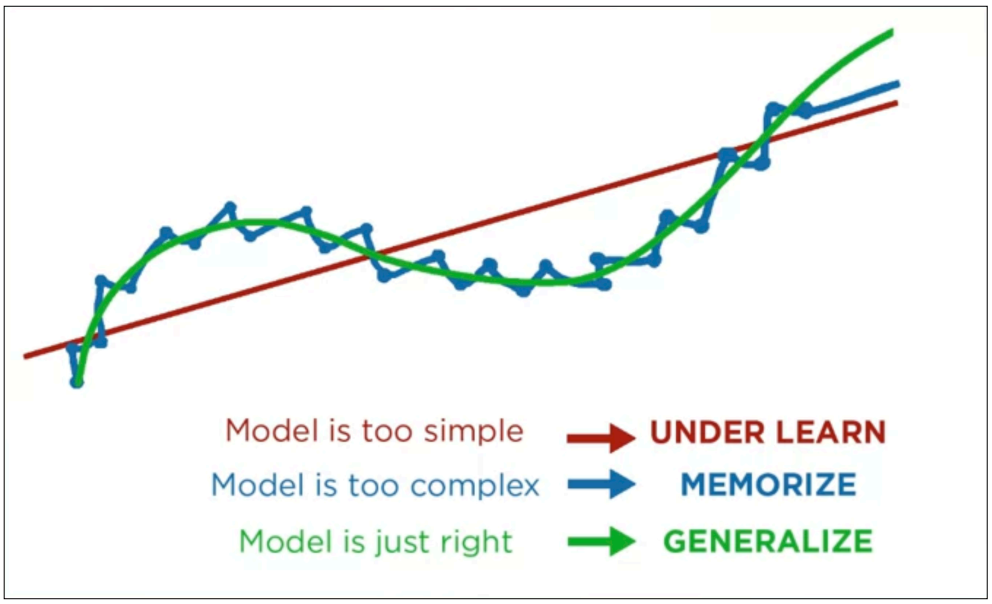


Рисунок 1.5 – Вимоги до моделі машинного навчання

Існують різні алгоритми машинного навчання, які можна застосувати для прогнозування ймовірності виникнення захворювань на основі факторів ризику. Для того щоб максимально вдало підібрати алгоритм потрібно дослідити як сам набір даних, так і існуючі варіанти алгоритмів. Даній задачі буде присвячено третій розділ кваліфікаційної роботи магістра.

1.4 Специфікація вимог до програмного забезпечення інформаційної системи

На основі попереднього аналізу предметної сфери можливо зазначити основні функціональні вимоги до інформаційної системи моніторингу ймовірності виникнення захворювання:

1. Авторизація та реєстрація (для розмежування прав доступу).
2. Управління даними про хворих на діабет (для медперсоналу).
3. Можливість додавання персональних даних та даних про поточні показники стану здоров’я (для пацієнтів).
4. Здійснення прогнозу стосовно захворювання на основі декількох біометричних параметрів.
5. Генерація статистики.

Що стосується нефункціональних вимог, то найважливішою є безпека. Медична інформація користувачів – це їх приватне надбання, яким не повинні заволодіти сторонні люди. Забезпечення конфіденційності та безпеки є одним з пріоритетних завдань при прогнозування застосунку.

Висновки до розділу 1

В результаті розробки першого розділу КРМ здійснено специфікацію вимог на основі аналізу предметної сфери. Інформаційна система має визначати ймовірність розвитку захворювання на основі аналізу факторів ризику. Дана автоматизована експертна інформаційно-логічна МІС повинна включати в себе алгоритми машинного навчання та ґрунтуватися на досягненнях доказової медицини.

Аналіз існуючих аналогів показав що при взаємодії з користувачем найкращим варіантом є або мобільний застосунок, або чат-бот. Отримання даних про стан здоров’я користувача не пов’язано з використанням вартісного медичного обладнання, тому впровадження даної системи не пов’язане зі значними витратами коштів.

2 АНАЛІЗ ФАКТОРІВ РИЗИКУ ЙМОВІРНОСТІ ВИНИКНЕННЯ ЗАХВОРЮВАНЬ

2.1 Аналіз шкал оцінки ризику виникнення захворювання

Існують наступні шкали оцінки ризику виникнення захворювання.

1. ***SCORE*** (Systematic Coronary Risk Evaluation) – це концепція розрахунку сумарного серцево-судинного ризику, сформульована в 90-х рр. XX століття. Дана система оцінює 5 чинників. Використовується тільки в первинній профілактиці і визначає ризик смертельного серцево-судинного захворювання протягом наступних 10 років. У своєму сучасному вигляді запропонована в 2003 році. За час існування знайшла відображення в посібниках і протоколах по артеріальній гіпертензії багатьох країн світу.

2. ***FINDRISC*** (шкала FINDRISC. Finnish Diabetes Risk Score) – це шкала, розроблена Фінською Асоціацією Діабету під керівництвом Jaakko Tuomilehto. Вона дозволяє оцінити 10-річний ризик цукрового діабету 2 типу, включаючи з 85% точністю.

3. ***MMSE*** (Mini-Mental State Examination) є найбільш широко поширеною методикою для скринінгу та оцінки тяжкості деменції. Являє собою опитування, що складається з 30 пунктів. З моменту розробки (1975 рік), в шкалу неодноразово вносилися зміни. Слід зазначити, що діагностична чутливість цієї методики не є абсолютною, тому остаточна інтерпретація залежить від фахівця.

4. ***HAS-BLED*** (Hypertension, Abnormal renal-liver function, Stroke, Bleeding history or predisposition, Labile international normalized ratio, Elderly, Drugs or alcohol concomitantly). Використовується для оцінки ризику великої кровотечі протягом 1 року. Шкала містить 9 пунктів, кожен з яких оцінюється в 1 бал. Якщо набрано більше 2 балів, то призначенні антикоагулянтів заборонене.

5. ***CHA2DS2-VASc*** і ***CHADS2.*** Антагоністи шкали HAS-BLED, вони використовуються для визначення ризику розвитку інсульту у пацієнтів. Останні дослідження підтверджують користь шкали CHA2DS2VASc, яка з’явилася відносно недавно і схоже незабаром повністю замінить CHA2DS2.

Використання системи бальної оцінки стану пацієнта відповідає всім принципам медицини та дозволяє не тільки проводити діагностику, але і оцінювати ефективність проведеного лікування [8]. Після збору даних від пацієнта необхідно інтерпретувати отримані результати за допомогою розрахунку кількості балів. Кількість балів за кожний варіант відповіді розрахована у такий спосіб щоб передавати вагу певного параметру (рис. 2.1).

1. Вік, роки	
До 45	0 балів
45–54	2 бали
55–64	3 бали
Старше 64 років	4 бали
2. ІМТ, кг/м^2	
Менше 25	0 балів
25–30	1 бал
Більше 30	3 бали
3. Окружність талії, см	
♂: <94 ♀: <80	0 балів
<94–102 <80-88	3 бали
>102 >88	4 бали
4. Чи приділяєте Ви кожного дня як мінімум 30 хвилин фізичної активності на роботі або під час відпочинку (включно зі звичайною повсякденною активністю)?	
Так	0 балів
Ні	2 бали
5. Як часто Ви вживаєте в їжу овочі, фрукти, ягоди?	
Щоденно	0 балів
Не кожен день	1 бал
6. Чи приймали Ви коли-небудь антигіпертензивні засоби?	
Ні	0 балів
Так	2 бали
7. Чи виявляли у Вас коли-небудь підвищений рівень глюкози у крові (наприклад, при диспансерному обстеженні, під час хвороби, у період вагітності)?	
Ні	0 балів
Так	5 балів
8. Чи був діагностований у когось з членів Вашої родини або найближчих родичів ЦД 1-го чи 2-го типу?	
Ні	0 балів
Так: дідусь, бабуся, тітка, дядько або кузени	3 бали
Так: батьки, брати, сестри або діти	5 балів
Сума балів	

Рисунок 2.1 – Опитування для оцінки ймовірності виникнення діабету

Наприклад, наявність хворих на діабет родичів та випадки виявлення високого рівня глюкози у крові суттєво підвищують ризик виникнення діабету (5 балів), тоді як прийом антигіпертензивних препаратів не має такого великого впливу (2 бали).

На рисунку 2.2 наведено приклад шкали для інтерпретації результатів опитування.

Сума балів	Очікуваний ризик
<7	Низький: розвиток ЦД можливий у 1 випадку зі 100
7–11	Незначно підвищений: розвиток ЦД можливий у 1 випадку з 25
12–14	Помірний: розвиток ЦД можливий у 1 випадку з 6
15–20	Високий: розвиток ЦД можливий в кожному третьому випадку
>20	Дуже високий: розвиток ЦД можливий в кожному другому випадку

Рисунок 2.2 – Інтерпретація отриманих результатів

Для інших захворювань існують відповідні шкали, але оскільки у роботі буде використано датасет про хворих на цукровий діабет в 2 розділі представлена саме така шкала оцінювання ризику ймовірності виникнення захворювання.

2.2 Формування вихідного набору факторів ризику, що впливають на виникнення захворювань

При виборі датасету є два можливих підходи: використання «тестового» набору даних та використання актуального набору даних (який піде в production).

Тестовий датасет дозволяє задати структуру даних та працювати над застосунком паралельно зі збором «справжніх» даних, на основі яких він має працювати в майбутньому. Пошук якісного набору даних є складною задачею. Найвищу якість мають відкриті дані. Відкриті дані – це інформація, що генерується державою та оприлюднюється в машиночитаній формі для її подальшого використання бізнесом, громадськістю, журналістами, дослідниками та органами влади [9]. В Україні існує спеціальний портал «Дія. Відкриті дані», звідки можна брати такі датасети.

Інший якісний ресурс – сайт «World Health Organization». Дана організація надає звітність про охорону здоров’я різних країн світу. Їх датасети більше орієнтовані на соціально-економічні, а не біометричні фактори ризику. WHO самостійно створює аналітичні звіти та візуалізацію даних.

На жаль, українські відкриті дані про пацієнтів та хвороби не вдалося знайти на вказаному вище порталі (рис. 2.3).

Рисунок 2.3 – Приклад відкритих даних

Отже, найбільш доречним підходом є використання тестового набору даних з заданою структурою для розробки та відлагодження застосунку. Готова до використання версія програми має працювати на основі подібного за структурою, але інакшого за вмістом набору даних. Цей датасет має бути зібраний фахівцями в ході дослідження та лікування пацієнтів.

В ході розробки застосунку буде використаний набір даних для прогнозування діабету з «Kaggle» [10]. На рисунку 2.4 представлено зразок даних з цього датасету.

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
0	Female	80.0	0	1	never	25.19	6.6	140	0
1	Female	54.0	0	0	No Info	27.32	6.6	80	0
2	Male	28.0	0	0	never	27.32	5.7	158	0
3	Female	36.0	0	0	current	23.45	5.0	155	0
4	Male	76.0	1	1	current	20.14	4.8	155	0

Рисунок 2.4 – Приклад даних з датасету

Обраний датасет має наступні поля:

1) *gender* – стать пацієнта.
2) *age* – вік пацієнта.
3) *hypertension* – наявність гіпертензії.

4) *heart_disease* – наявність серцевих захворювань.

5) *smoking_history* – досвід паління.

6) *bmi* – індекс маси тіла.

7) *HbA1c_level* – середній рівень глюкози за останні 2 – 3 місяці.

8) *blood_glucose_level* – поточний рівень глюкозу в крові.

9) *diabetes* – наявність діабету.

Отже, інформаційна система призначена для прогнозування ризику розвитку діабету на основі наведених факторів ризику. Цукровий діабет є хронічним захворюванням, яке пов’язане з недостатнім виробленням та неефективним використанням інсуліну. Інсулін відповідає за регулювання рівня цукру в кров та забезпечує клітини енергією. Існують два основних типи цукрового діабету:

1) *Діабет першого типу.* Це аутоімунне захворювання, за якого імунна система організму атакує та знищує клітини, відповідальні за вироблення інсуліну в підшлунковій залозі. Люди з діабетом 1-го типу повинні отримувати інсулін шляхом ін’єкцій для утримання нормального рівня цукру в крові.

2) *Діабет другого типу.* Цей тип діабету виникає коли організм не виробляє достатньо інсуліну або не може його ефективно використати. Діабет 2-го типу зазвичай розвивається внаслідок генетичних факторів, способу життя та інших чинників. Лікування може включати в себе дієту, фізичну активність, спеціальні ліки та, в окремих випадках, інсулін.

Спільними симптомами цукрового діабету є підвищений рівень цукру в крові, слабкість, спрага тощо. Невідкладна діагностика та лікування діабету є важливими для уникнення ускладнень.

Оскільки фахівець з медичною освітою не надає рекомендації та не здійснює перевірку в ході розробки системи важливо спиратися на медичну інформацію з надійного джерела. Система буде надавати поради стосовно подальших дій після здійснення прогнозування ймовірності виникнення захворювання, тому важливо щоб ці поради базувалися на достовірних відомостях та досягненнях доказової медицини. Наприклад, варто спиратися на рекомендації

стосовно лікування діабету від організації «National Institute of Diabetes and Digestive and Kidney Diseases» (США) [11].

Візуальне представлення інформації, рекомендації та кількісна оцінка ризику виникнення або наявності хвороби сприяє тому що користувач краще розуміє стан свого здоров'я та подальші кроки для покращення самопочуття.

2.3 Метод дерева рішень

Дерева рішень використовуються для здійснення класифікації та передбачення значення цільової змінної на основі декількох вхідних параметрів (факторів). У даному випадку використовується ієрархічна структура для представлення множини можливих рішень (рис. 2.5).

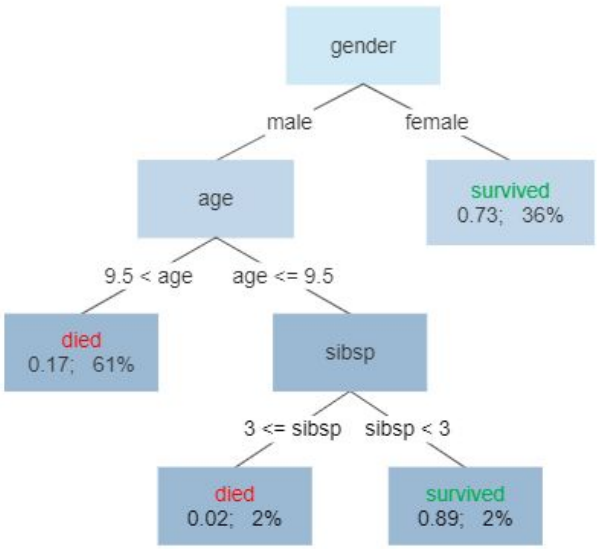


Рисунок 2.5 – Приклад дерева рішень [12]

Відповідний граф дещо подібний до блок-схеми алгоритму, оскільки демонструє умови переходу від одного рішення до іншого.

Якщо величина яка прогнозується приймає дискретні значення (наприклад, «має захворювання» та «не має захворювання»), то виконується задача *класифікації (classification tree)*.

При прогнозуванні значення безперервної величини (наприклад, вартість квартири на основі кількості кімнат, площі тощо) виконується задача *чисельного прогнозування (regression tree)*.

Дерево рішень складається з двох типів вузлів: умови (*branches*) та результати (*leaves*).

2.4 Алгоритм побудови нечітких дерев рішень

Коефіцієнт P_i^N – це відношення прикладів $D_j \in S^N$ вузла N для цільового значення i:

$$P_i^N = \sum_{S^N} min(\mu_N(D_j), \mu_i(D_j)) \tag{2.1}$$

де $\mu_N(D_j)$ – ступінь приналежності прикладу D_j до вузла N,

$\mu_i(D_j)$ – ступінь приналежності прикладу відносно цільового значення i,

S^N – множина всіх прикладів вузла N.

Далі розраховуються коефіцієнт P^N, що позначає загальні характеристики прикладів вузла N.

Для нечітких дерев обраховують відношення прикладів вузла N для цільового значення i (P_i^N) до узагальненого значення коефіцієнта прикладів вузла N (P^N). Вираз:

$$E(S^N) = -\sum_i \frac{P_i^N}{P^N} \cdot \log_2 \frac{P_i^N}{P^N} \tag{2.2}$$

дає оцінку середньої кількості інформації для визначення класу об’єкту із множини P^N.

На наступному кроці побудови нечіткого дерева рішень алгоритм розраховує ентропію для розбиття за атрибутом A зі значеннями a_j:

$$E(S^N, A) = -\sum_j \frac{P^{N|j}}{P^N} \cdot E(S^{N|j}) \tag{2.3}$$

де вузол $N \mid j$ – дочірній для вузла N.

Алгоритм обирає атрибут A' з максимальним приростом інформації

$$G(S^N, A) = E(S^N) - E(S^N, A) \tag{2.4}$$

$$A' = arg\ max_A\ G(S, A) \tag{2.5}$$

Вузол N розбивається на декілька під вузлів $N \mid j$.

Ступінь приналежності прикладу D_k вузла $N \mid j$ обраховується покроково з вузла N як:

$$\mu_{N|j}(e_k) = min(\mu_{N|j}(D_k), \mu_{N|j}(D_k, a_j)) \quad (2.6)$$

де $\mu_{N|j}(D_k, a_j)$– ступінь приналежності D_kдо атрибуту a_j.

Підвузол $N \mid j$ видаляється якщо всі приклади в ньому мають нульову ступінь приналежності.

Алгоритм повторюється доки всі приклади вузла не будуть класифіковані , або доки не будуть використані для розбиття (класифікації) всі атрибути.

Приналежність до цільового класу для нового запису (прикладу) визначається за формулою:

$$\sigma_j = \frac{\sum_l \sum_k P_k^l \cdot \mu_l(D_j) \cdot \chi_k}{\sum_l (\mu_l(D_j) \cdot \sum_k P_k^l)} \quad (2.7)$$

де P_k^l – коефіцієнт співвідношення прикладів листа дерева l для значення цільового класу класу k,

$\mu_l(D_j)$ – ступінь приналежності прикладу до вузла l,

χ_k – приналежність значення цільового класу k до позитивного результату класифікації.

2.5 Приклад визначення ймовірності захворювання на основі аналізу факторів ризику

Нехай наведено дані про 7-х пацієнтів лікарні: вік, індекс маси тіла та наявність діабету другого типу. Необхідно побудувати нечітке дерево рішень для визначення ймовірності виникнення діабету другого типу.

Припустимо, що атрибут (лінгвістична змінна) «вік» може приймати значення «дитячій», «дорослий» та «похилий», а атрибут (лінгвістична змінна) «індекс маси тіла» – значення «недостатній», «нормальний» та «надмірний». У таблиці 2.1 представлено відповідні дані.

Таблиця 2.1 – Навчальні приклади для побудови нечіткого дерева рішень

№	вік	індекс маси тіла	наявність діабету
D1	14	18	0.05
D2	19	20	0.1
D3	45	33	1
D4	44	23	0.2
D5	85	36	1
D6	70	27	0.7
D7	62	32	0.9

Таким чином маємо:

1. Для «вік»
   - універсальна множина U = {0; 85};
   - множина термів T = {«дитячий», «дорослий», «похилий»};
2. Для «індекс маси тіла»:
   - універсальна множина U = {0; 37};
   - множина термів T = { «недостатній», «нормальний», «надмірний»}.

Для формалізації атомарних термів використаємо нечітку множину, що має лінійну функцію приналежності з параметрами:

1. Для «вік»:
   - «дитячий» (0, 0, 16, 18);
   - «дорослий» (18, 36, 55, 65);
   - «похилий» (65, 70, 75, 85);
2. Для «індекс маси тіла»:
   - «недостатній» (14, 16, 17, 18);
   - «нормальней» (18, 22, 24, 25);
   - «надмірний» (25, 28, 30, 37).

Для задання трапецієподібної функції приналежності використовується четвірка чисел (a, b, c, d). За вказаною формулою розраховано значення з таблиці 2.2.

$$\mu(x) = \begin{cases} 1 - \dfrac{b-x}{b-a}, & a \le x \le b; \\ 1, & b \le x \le c; \\ 1 - \dfrac{x-c}{d-c}, & c \le x \le d; \\ 0, & \textit{в інших випадках.} \end{cases} \quad (2.8)$$

Таблиця 2.2 – Ступінь приналежності кожного прикладу до значень атрибутів

№	вік			індекс маси тіла		
	дитячий	дорослий	похилий	недостатній	нормальний	надмірний
D1	1	0	0	0	0	0
D2	0	0.06	0	0	0.5	0
D3	0	1	0	0	0	0.6
D4	0	1	0	0	1	0
D5	0	0	0	0	0	0.14
D6	0	0	1	0	0	0.7
D7	0	0.3	0	0	0	0.7
SUM	**1**	**2.36**	**1**	**0**	**1.5**	**2.4**

Розрахунок загальної ентропії у відповідності до формули 2.2 враховуючи те що $P_{yes} = 4$ та $P_{no} = 3$:

$$E(S^N) = -\left(\frac{4}{7} \cdot \log_2 \frac{4}{7} + \frac{3}{7} \cdot \log_2 \frac{3}{7}\right) = 0.99$$

Таблиця 2.3 – Результат розрахунків нечітких показників для атрибуту «вік»

	дитячий	дорослий	похилий
P_{yes}	0.05	1.56	0.7
P_{no}	0.95	0.96	0.3
E	0.286	0.958	0.881

Приріст інформації для різних факторів становить:

$$G(S^N, age) = 0.499$$

$$G(S^N, bmi) = 0.649$$

Максимальний приріст інформації забезпечує атрибут «індекс маси тіла», тому розбиття почнеться з нього. На наступному кроці алгоритму необхідно для кожного запису розрахувати ступінь приналежності до кожного нового вузла за формулою 2.6. Результат розрахунків представлено у таблиці 2.4.

Таблиця 2.4 – Приналежність до нових вузлів дерева

IMT	недостатній			нормальний			надмірний		
Вік	дитячий	дорослий	похилий	дитячий	дорослий	похилий	дитячий	дорослий	похилий
D1	0	0	0	0	0	0	0	0	0
D2	0	0	0	0	0.06	0	0	0	0
D3	0	0	0	0	0	0	0	0.6	0
D4	0	0	0	0	1	0	0	0	0
D5	0	0	0	0	0	0	0	0	0
D6	0	0	0	0	0	0	0	0	0.7
D7	0	0	0	0	0	0	0	0.3	0

Розрахунок нечітких показників атрибута «IMT – нормальний – вік - дорослий»:

$\mathbf{P_{так}}$ = min(0,05;0) + min(0,1;0,06) + min(1;0) + min (0,2;1) + min(1;0) + min(0,7;0) + min(0,9;0) = 0 + 0,06 + 0 + 0,2 + 0 + 0 +0 = 0,26

$\mathbf{P_{ні}}$= min(0,95;0) + min(0,9;0,06) + min(0;0) + min (0,8;1) + min(0;0) + min(0,3;0) + min(0,1;0) = 0 + 0,06 + 0 + 0,8 + 0 + 0 +0 = 0,86

Розрахунок нечітких показників атрибута «IMT – надмірний – вік – дорослий»

$\mathbf{P_{так}}$ = min(0,05;0) + min(0,1;0) + min(1;0.6) + min (0,2;0) + min(1;0) + min(0,7;0) + min(0,9;0.3) = 0 + 0 + 0.6 + 0 + 0 + 0 +0.3 = 0,9

$\mathbf{P_{ні}}$= min(0,95;0) + min(0,9;0) + min(0;0.6) + min (0,8;0) + min(0;0) + min(0,3;0) + min(0,1;0,3) = 0 + 0 + 0 + 0 + 0 + 0 + 0 +0,1 = 0,1

Розрахунок нечітких показників атрибута «IMT – надмірний – вік – похилий»

$\mathbf{P_{так}}$ = min(0,05;0) + min(0,1;0) + min(1;0) + min (0,2;0) + min(1;0) + min(0,7;0,7) + min(0,9;0) = 0 + 0 + 0 + 0 + 0 + 0,7 +0 = 0,7

$\mathbf{P_{ні}}$= min(0,95;0) + min(0,9;0) + min(0;0) + min (0,8;0) + min(0;0) + min(0,3;0,7) + min(0,1;0) = 0 + 0 + 0 + 0 + 0 + 0,3 +0 = 0,3

Графічна ілюстрація розробленого нечіткого дерева рішень представлена на рисунку 2.6.

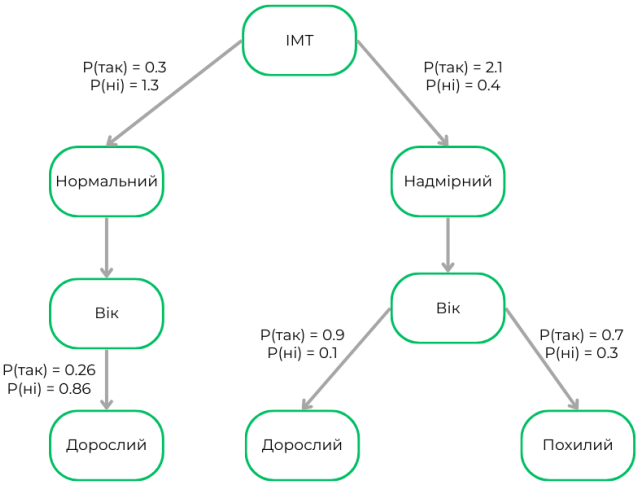


Рисунок 2.6 – Розроблене дерево рішень

Якщо необхідно здійснити прогнозування для людини віком 35 років та ІМТ 28, то йдеться про «дорослий» вік та «надмірний ІМТ».

Гілка (ІМТ= надмірний) та (вік = дорослий)

$\mathbf{P_{так}} = 0,9$ $\chi_{так} = 1,0$

$\mathbf{P_{ні}} = 0,1$ $\chi_{ні} = 0,0.$

$\mu = 0,95$

$$\delta = \frac{0,9 \times 0,95 \times 1,0 + 0,1 \times 0,95 \times 0,0}{(0,9 + 0,1) \times 0,95} = 0,9 = 90\,\%$$

Отже, зазначений пацієнт з 90% ймовірністю знаходиться у групі ризику через діабет другого типу. Текст програми для розрахунку числових значень у п.2.5 наведено у додатку А.

Висновки до розділу 2

У другому розділі проаналізовано існуючі шкали для оцінювання ризику виникнення захворювання, обрано набір даних та алгоритм машинного навчання.

3 МОДЕЛЮВАННЯ ТА ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Діаграма діяльності системи

На рисунку 3.1 представлена діаграма діяльності (*activity diagram*). Діаграми даного типу показують послідовність процесів у системі та умови переходу від одного процесу до іншого.

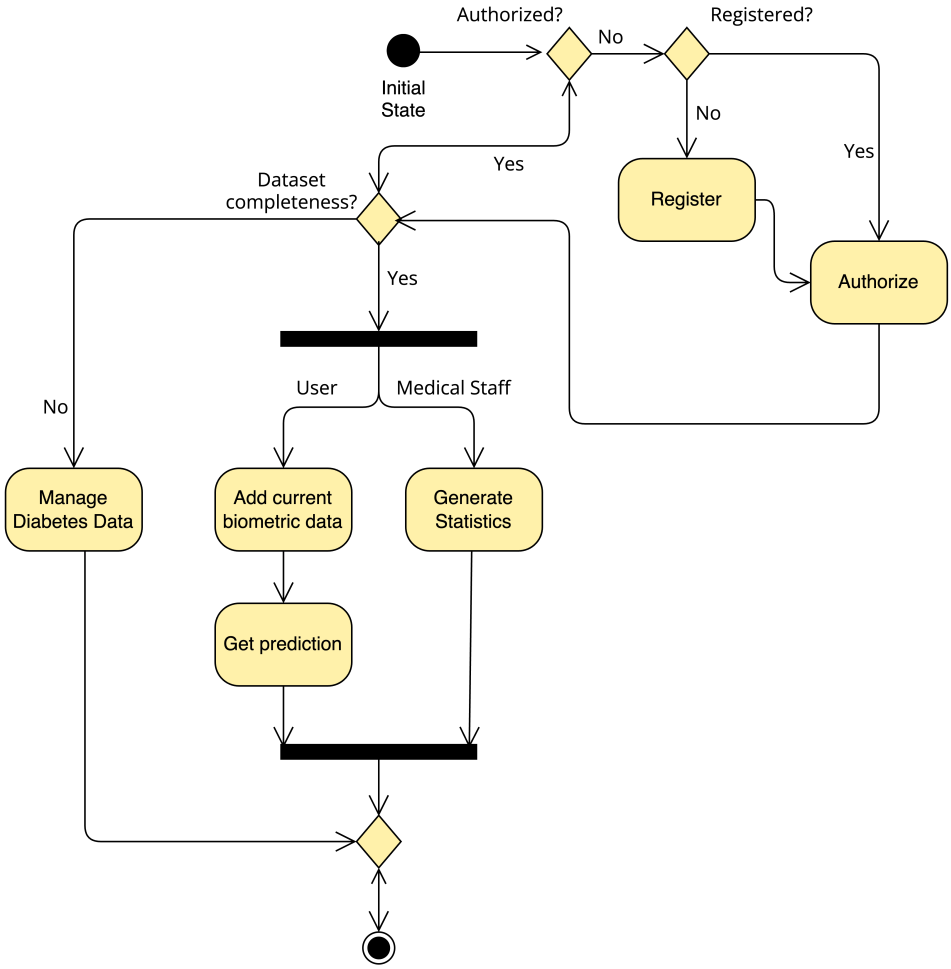


Рисунок 3.1 – Діаграма діяльності для ІС

Отже, після авторизації та реєстрації можливі декілька варіантів подальших дій. Система може бути доповнена медичними даними або використана за призначенням у поточному стані. Пацієнт може отримати прогноз стосовно розвитку захворювання, а медперсонал може генерувати звітність.

На рисунку 3.2 представлена діаграма послідовності (*sequence diagram*), яка показує впорядковані за часом взаємодії між об’єктами. В даному випадку йдеться про найголовніший процес у системі – отримання користувачем прогнозу стосовно ймовірності виникнення захворювання.

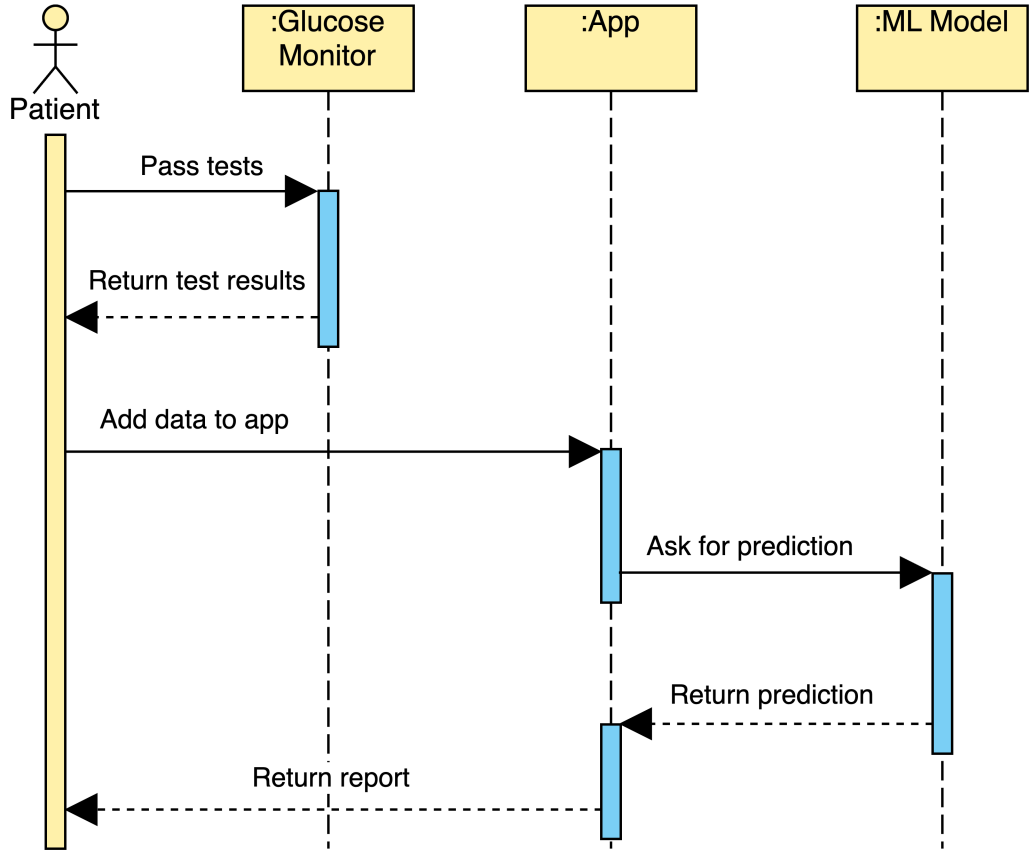


Рисунок 3.2 – Діаграма послідовності для ІС

На діаграмі послідовності «Glucose Monitor» є умовним позначенням як для домашнього пристрою для вимірювання поточного рівня глюкози, так і для лабораторного обладнання для вимірювання рівня HbA1c. В будь-якому випадку користувач має ввести в інформаційну систему результати аналізу крові. Інформаційна система використовує створену за допомогою машинного навчання модель, але ця навчена модель є окремим програмним продуктом який придатний для використання в будь-якому іншому застосунку.

3.2 Аналіз варіантів використання системи

Для документування вимог розроблено діаграму варіантів використання, яка представлена на рисунку 3.3. Основними дійовими особами є незареєстрований користувач, медперсонал та пацієнти.

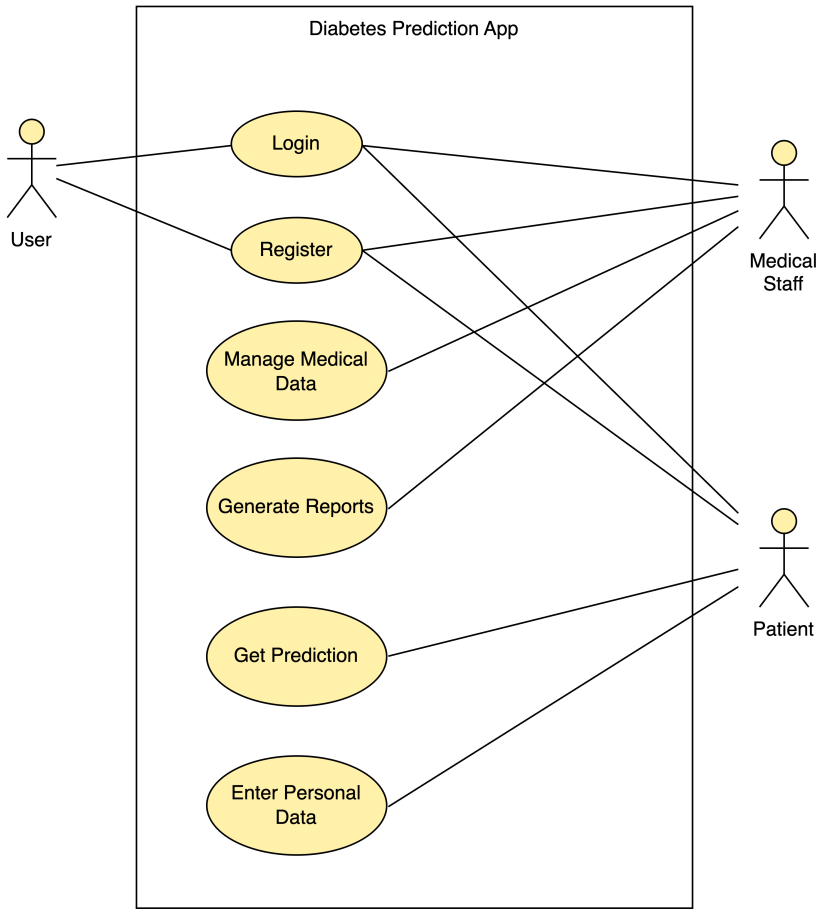


Рисунок 3.3 – Use Case Diagram

Створена діаграма показує як розмежування прав доступу, так і основні цілі акторів. Для розробки діаграм для другого розділу КРМ використано онлайн-ресурс «Visual Paradigm» та UML.

Таблиця 3.1 – Варіант використання «Login»

Use Case 1: Login	
Актори:	Неавторизований корситувач та інофрмаційна система.
Мета:	Авторизуватися в системі.
Передумова:	Користувач не авторизований, але зареєстрований в системі.
Успішний сценарій: 1. Користувач переходить до форми авторизації. 2. Користувач вводить дані. 3. Система здійснює валідацію даних. 4. Користувач авторизований в системі.	
Результат:	Користувач авторизувався в системі.
Розширення:	
**а*	Користувач не зареєстрований в системі. Система здійснює переадресацію на сторінку реєстрації.
1а	Введені дані не пройшли валідацію. Користувач отримує повідомлення про помилку та наступну спробу авторизуватися.
1б	Користувач обрав опцію «відновити пароль». Система здійснює переадресацію до форми відновлення пароля.

Таблиця 3.2 – Варіант використання «Register»

Use Case 2: Register	
Актори:	Неавторизований користувач, інформаційна система
Мета:	Зареєструватися в системі.
Передумова:	Користувач не зареєстрований в системі.
Успішний сценарій: 1. Користувач переходить до форми реєстрації. 2. Користувач заповнює форму реєстрації. 3. Інформаційна система здійснює валідацію та збереження даних. 4. Користувач зареєстрований та авторизований в системі.	
Результат:	Користувач зареєструвався в системі.
Розширення:	
**a*	Помилка з'єднання з базою даних. Користувач отримує повідомлення про те що сталася помилка і варто спробувати пізніше.
1a	Користувач вже має обліковий запис, який прив'язаний до цього email. Система пропонує відновити пароль.

Таблиця 3.3 – Варіант використання «Manage Medical Data»

Use Case 3: Manage Medical Data	
Актори:	Медпрацівник та інформаційна система.
Мета:	Підвищити якість датасету за допомогою додавання, редагування та видалення даних.
Передумова:	Медпрацівник вже авторизований в системі.
Успішний сценарій: 1. Медпрацівник переходить до форми редагування набору даних. 2. Медпрацівник вносить зміни в датасет. 3. Інформаційна система здійснює валідацію та збереження введених даних. 4. Інформаційна система оновлює модель для прогнозування.	
Результат:	Медпрацівник здійснив управління даними.
Розширення:	
**a*	Помилка з’єднання з базою даних. Користувач отримує повідомлення про те що сталася помилка і варто спробувати пізніше.
1a	Додані або відредаговані дані не пройшли валідацію. Користувач отримує повідомлення та не може зберегти зміни поки не виправить помилку.

Таблиця 3.4 – Варіант використання «Generate Reports»

Use Case 4: Generate Reports	
Актори:	Медпрацівник та інформаційна система.
Мета:	Створити звітність про ймовірність виникнення захворювання.
Передумова:	Медпрацівник авторизований в системі.
Успішний сценарій: 1. Медпрацівник переходить до форми створення звіту. 2. Медпрацівник обирає тип звіту. 3. Медпрацівник налаштовує параметри звіту. 4. Інформаційна система формує звіт. 5. Медпрацівник отримує звіт.	
Результат:	Медпрацівник успішно створив звіт.
Розширення:	
**a*	Помилка з’єднання з базою даних. Користувач отримує повідомлення про те що сталася помилка і варто спробувати пізніше.
1a	Задані параметри фільтрації призвели до того що в звіт не потрапив жоден запис. Система виводить повідомлення про помилку та пропонує спробувати інші параметри.

Таблиця 3.5 – Варіант використання «Get Prediction»

Use Case 5: Get Prediction	
Актори:	Пацієнт та інформаційна система.
Мета:	Отримати прогноз стосовно ймовірності виникнення захворювання.
Передумова:	Пацієнт авторизований в системі.
Успішний сценарій: 1. Пацієнт переходить до форми для прогнозування ймовірності виникнення захворювання. 2. Користувач вводить власні біометричні дані. 3. Інформаційна система здійснює валідацію даних. 4. Інформаційна система здійснює прогнозування.	
Результат:	Пацієнт отримав прогноз стосовно ймовірності виникнення захворювання.
Розширення:	
**a*	Помилка з’єднання з базою даних. Користувач отримує повідомлення про те що сталася помилка і варто спробувати пізніше.
1a	Введені пацієнтом дані не пройшли валідацію. Система виводить повідомлення про помилку та вимагає виправити введені дані.

Таблиця 3.6 – Варіант використання «Enter Personal Data»

Use Case 6: Enter Personal Data	
Актори:	Пацієнт та інформаційна система.
Мета:	Додати персональні дані.
Передумова:	Пацієнт авторизований в системі.
Успішний сценарій: 1. Пацієнт переходить до форми додавання персональних даних. 2. Пацієнт вводить персональні дані. 3. Інформаційна система здійснює валідацію та збереження даних.	
Результат:	Пацієнт оновив свої персональні дані в системі.
Розширення:	
**a*	Помилка з’єднання з базою даних. Користувач отримує повідомлення про те що сталася помилка і варто спробувати пізніше.
1a	Введені пацієнтом персональні дані не пройшли валідацію. Система виводить повідомлення про помилку та вимагає виправити введені дані.

В системі розділено процес додавання поточних даних про рівень глюкози в крові (UC-5) та більш стабільну інформацію про вагу, зріст, вік тощо (UC-6).

3.3 Проєктування інтерфейсу користувача

Для зручності взаємодії з застосунком варто спроєктувати інтерфейс у форматі чат-боту. При створенні чат-ботів є декілька опцій: створення за допомогою програмування або створення з використанням конструктора. Створення за допомогою конструктора є гарним варіантом для тих випадків коли відбувається автоматизація стандартних процедур (наприклад, чат-бот служби підтримки чи чат-бот для запису в чергу). Якщо є потреба в унікальних та вузькоспеціалізованих функціях, то краще запрограмувати застосунок. В даній роботі йдеться про машинне навчання, тому для впровадження відповідного алгоритму та гнучкого налаштування системи варто розробити чат-бот самостійно. В якості месенджера обрано ***Telegram***. Немає потреби схематично зображати загальновідомий та вже розроблений GUI обраного месенджера, але можна створити діаграму, яка відображає порядок демонстрації повідомлень (*bot conversation flow diagram*).

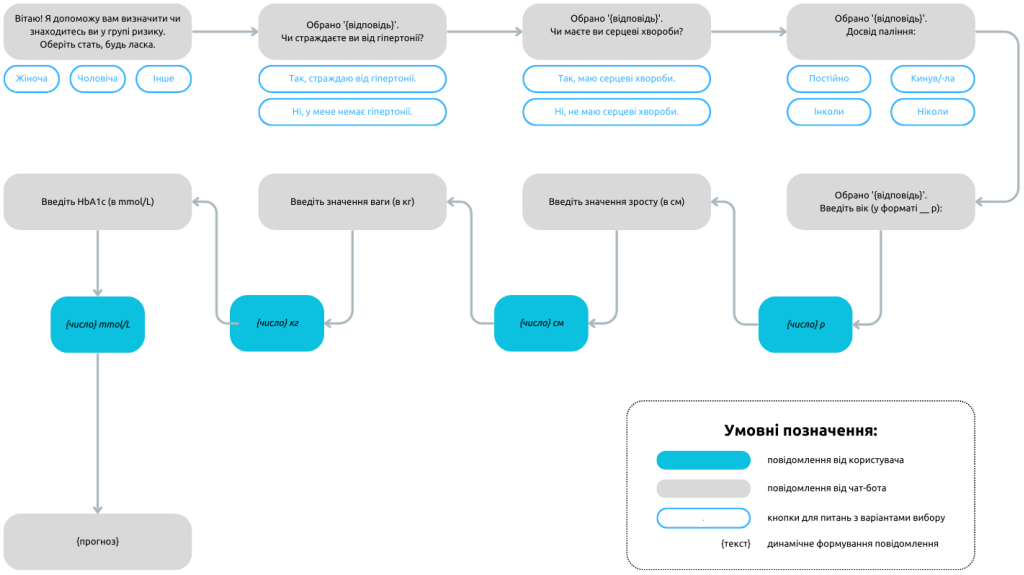


Рисунок 3.4 – Скрипти повідомлень чат-бота

Зручність чат-ботів полягає в тому що користувачі взаємодіють з вже знайомим та заздалегідь встановленим інтерфейсом. Це є перевагою оскільки не потрібно марно витрачати обчислювальні ресурси на завантаження та виконання зайвих застосунків. Також не потрібно витрачати час на вивчення нової програми.

3.4 Обробка даних у інформаційній системі

На рисунку 3.5 представлена *Data Flow Diagram (DFD)*. В даній медичній інформаційній системі дані походять з багатьох джерел, тому варто детальніше продемонструвати як саме відбувається процес роботи з даними.

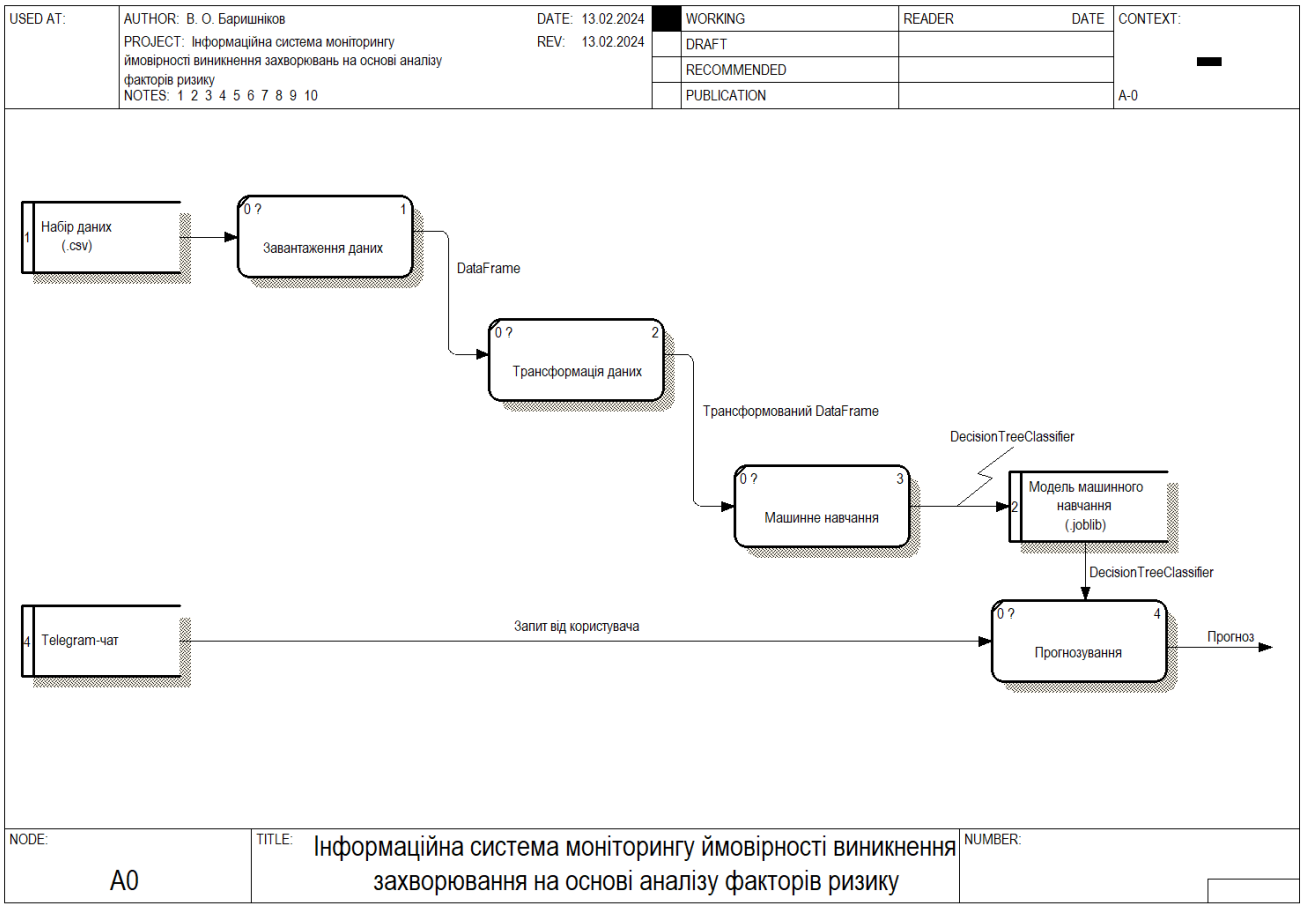


Рисунок 3.5 – Data Flow Diagram

Розроблена модель машинного навчання може бути використана в інших проєктах разом з іншими алгоритмами для розробки необхідного функціоналу.

Висновки до розділу 3

В даному розділі охарактеризовано процес проєктування та моделювання системи за допомогою UML-діаграм, DFD та мапи скриптів чат-бота. Визначено вимоги до програмного забезпечення. Наступним кроком необхідно здійснити програмну реалізацію застосунка.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ МЕДИЧНОЇ ІНФОРМАЦІЙНОЇ СИСТЕМИ

4.1 Вибір засобів розробки програмного забезпечення інформаційної системи

В якості мови програмування обрано ***Python***. Дана мова програмування характеризується динамічною типізацією, використанням інтерпретатора та механізму garbage collection. Python має бібліотеки для машинного навчання (*scikit-learn*), аналізу даних (*pandas*) та розробки чат-ботів (*python-telegram-bot*). Отже, є все необхідне для програмної реалізації медичної інформаційної системи. Набір бібліотек для data science вирізняється розмаїттям (рис. 4.1).

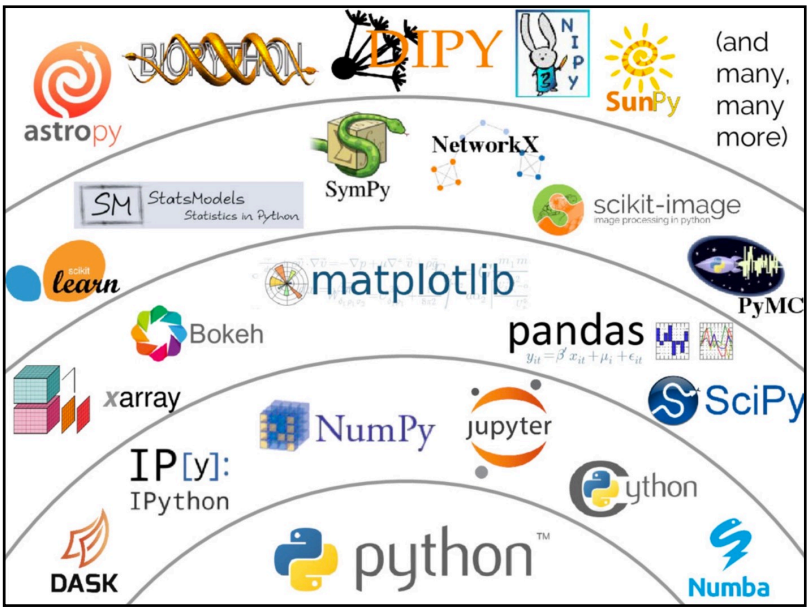


Рисунок 4.1 – Екосистема Python [13]

Обраний стек технологій має наступні переваги:

1) Читабельність коду та легкість розробки. Мова програмування Python у порівнянні з багатьма іншими мовами програмування може вважатися найбільш простою для розуміння та вивчення. Текст програми максимально наближений до розмовної англійської мови та базується на правилах, більшість з яких є інтуїтивно зрозумілими. До всіх бібліотек та функцій створена якісна документація. Це досить популярна мова програмування, тому велика спільнота дозволяє отримати підтримку та консультації в ході розробки.

2) Бібліотеки. Як стандартна бібліотека, так і завантажені додаткові інструменти пропонують велику кількість можливостей (розробка вебзастосунків, чат-ботів, автоматизація, машинне навчання, аналіз даних тощо).

3) Кросплатформність. Важливо щоб застосунок був сумісний з більшістю популярних платформ. Код мовою Python працює на macOS, Windows та Linux.

Серед недоліків Python варто зазначити:

1) Performance. У порівнянні з C та C++ код виконується досить повільно. Саме тому Python не використовується для тих задач, у яких мінімізація часу виконання є важливою.

2) Залежності. Існують різні версії використаних бібліотек. Інколи виникає конфлікт версій. Чим більше залежностей, тим вища ймовірність виникнення проблем з сумісністю. Застосування віртуального оточення дозволяє вирішити дану проблему.

3) Не є популярною опцією для розробки мобільних застосунків та майже не застосовується при розробці комп'ютерних ігор попри існування відповідних бібліотек. Через проблеми з часом виконання не користується популярністю.

Після аналізу всіх переваг на недоліків можна зрозуміти що Python дозволяє виконати поставлені завдання у найбільш ефективний спосіб. Python застосовується для програмування backend (логіка роботи бота). В якості frontend обрано Telegram.

4.2 Підготовка вихідного набору даних

Програмна реалізація починається з аналізу даних та машинного навчання. Для виконання цих задач зручно використати інтерактивне середовище, яке дозволяє поєднувати процес виконання коду та створення звітності.

Для даного проєкту обрано ***Google Colaboratory***. Вказаний сервіс надає можливість покроково виконувати та тестувати код. Додавання фрагментів тексту, рівнянь та зображень підвищує читабельність та доступність розробленого програмного забезпечення.

Перш за все необхідно дослідити структуру завантаженого набору даних. У бібліотеці pandas є всі необхідні засоби для завантаження, обробки та трансформації даних. Отже, завантажений датасет складається з ***9*** стовпчиків та ***100 000*** рядків (рис. 4.2).

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 9 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   gender               100000 non-null  object 
 1   age                  100000 non-null  float64
 2   hypertension         100000 non-null  int64  
 3   heart_disease        100000 non-null  int64  
 4   smoking_history      100000 non-null  object 
 5   bmi                  100000 non-null  float64
 6   HbA1c_level          100000 non-null  float64
 7   blood_glucose_level  100000 non-null  int64  
 8   diabetes             100000 non-null  int64  
dtypes: float64(3), int64(4), object(2)
memory usage: 6.9+ MB
```

Рисунок 4.2 – Характеристика датасету

У наборі даних є категоріальні змінні, які представлено множиною рядків (наприклад, стать як «Female», «Male» та «Other»). У ***DecisionTreeClassifier*** використовуються лише числові параметри, тому варто закодувати рядки у вигляді певних чисел за допомогою методів класу ***LabelEncoder***. Інформація про тютюнопаління має бути представлена у бінарному форматі («так» або «ні») без зайвих градацій. На рисунку 4.3 представлені відповіді методи для попередньої обробки набору даних. Пусті рядки у датасеті відсутні.

```
def smoking_history_binary(smoking_history):
    yes = ['current', 'former', 'ever']
    no = ['never', 'No Info',  'not current']

    if smoking_history in yes:
        return 1
    else:
        return 0

df['smoking_history'] = df.apply(lambda row: smoking_history_binary(row['smoking_history']), axis=1)
```

```
label_encoder = LabelEncoder()

# Encode "gender" parameter
df["gender"] = label_encoder.fit_transform(df["gender"])
mapping = dict(zip(label_encoder.classes_, range(len(label_encoder.classes_))))
print(f"Gender dict: {mapping}")
```

```
Gender dict: {'Female': 0, 'Male': 1, 'Other': 2}
```

Рисунок 4.3 – Трансформація набору даних

Після попередньої обробки можливо перейти до створення моделі машинного навчання.

Перш ніж створювати відповідну модель варто дослідити кореляцію між параметрами та з’ясувати що саме є найбільш характерною ознакою наявності діабету. Після створення кореляційної матриці можна зробити висновок що такими параметрами є два майже ідентичні показники: рівень глюкози в крові та рівень глікованого гемоглобіну в крові (HbA1c). Відповідна матриця представлена на рисунку 4.4.

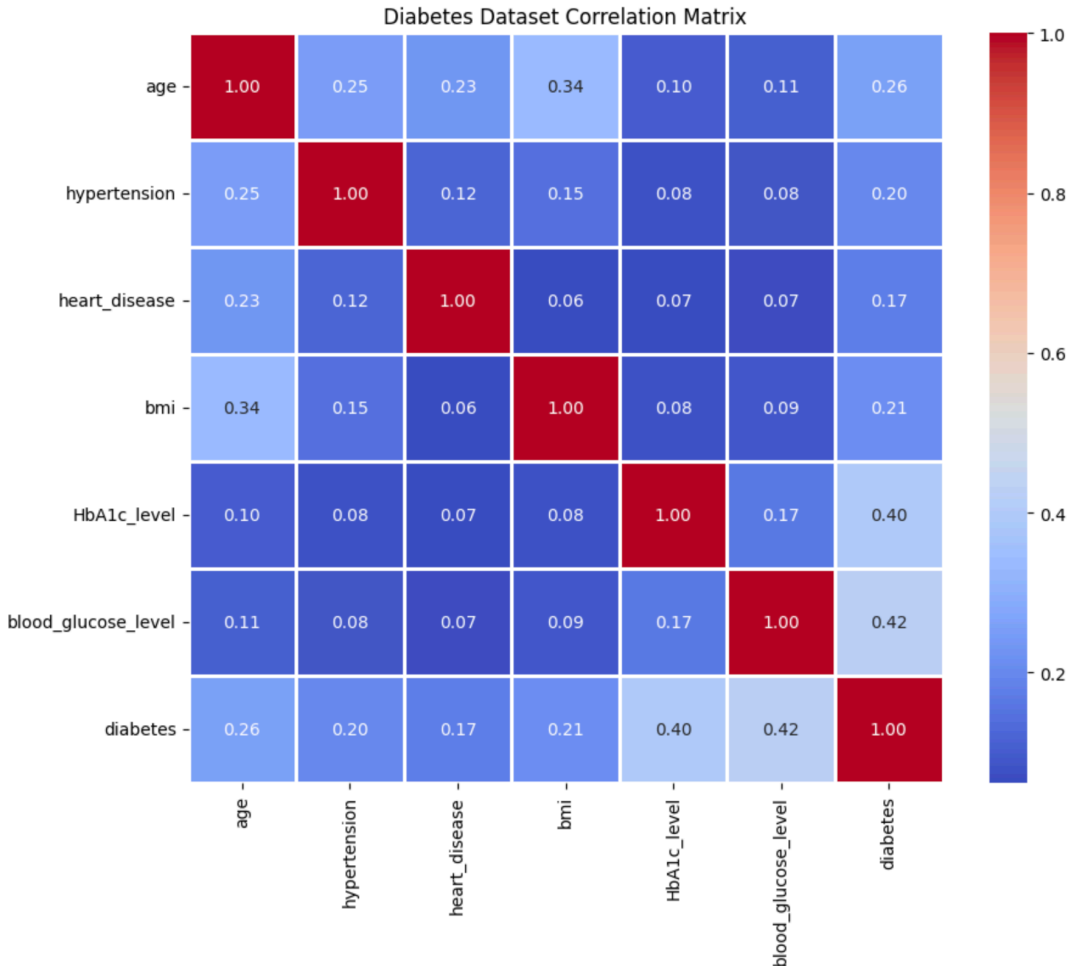


Рисунок 4.4 – Кореляційна матриця

При дослідженні предметної сфери з’ясовано що рівень глюкози в крові не є надійним показником. Навіть у здорових людей цей показник може виходити за межі норми у випадку незбалансованого харчування або відразу після прийому їжі. Тому спиратися на цей параметр для діагностики недоцільно. Саме тому розроблена модель буде враховувати саме лікований гемоглобін для визначення ймовірності виникнення захворювання.

До здійснення машинного навчання варто розділити датасет на тестовий та тренувальний набір. Це необхідно для визначення точності прогнозування за допомогою методу accuracy_score з бібліотеки scikit-learn. Також в даній бібліотеці реалізовано механізм підбору оптимальних параметрів (GridSearchCV). Після обрання найкращих налаштувань можливо виконати навчання створеної моделі. Відповідний текст програми представлено на рисунку 4.5.

Рисунок 4.5 – Здійснення машинного навчання

Необхідно обчислити точність прогнозування та зберегти створене дерево рішень для подальшого використання у backend (рис. 4.6). Точність прогнозування становить ***95.34%***.

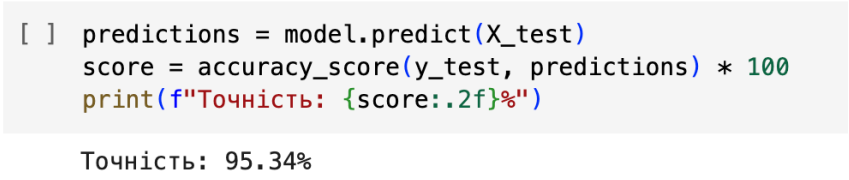


Рисунок 4.6 – Визначення точності прогнозування

Отже, на даному етапі розв’язано всі задачі, які стосуються обробки даних. Надалі задача спрощується до розробки чат-боту. Текст програми для попередньої обробки вихідного набору даних та аналізу якості побудованої моделі наведено у додатку Б.

4.3 Налаштування чат-бота для прогнозування ймовірності виникнення захворювання

Для створення токену необхідно використати чат-бот @BotFather та за допомогою простих команд налаштувати його назву, опис та інші параметри. Приклад налаштування представлено на рисунку 4.7.

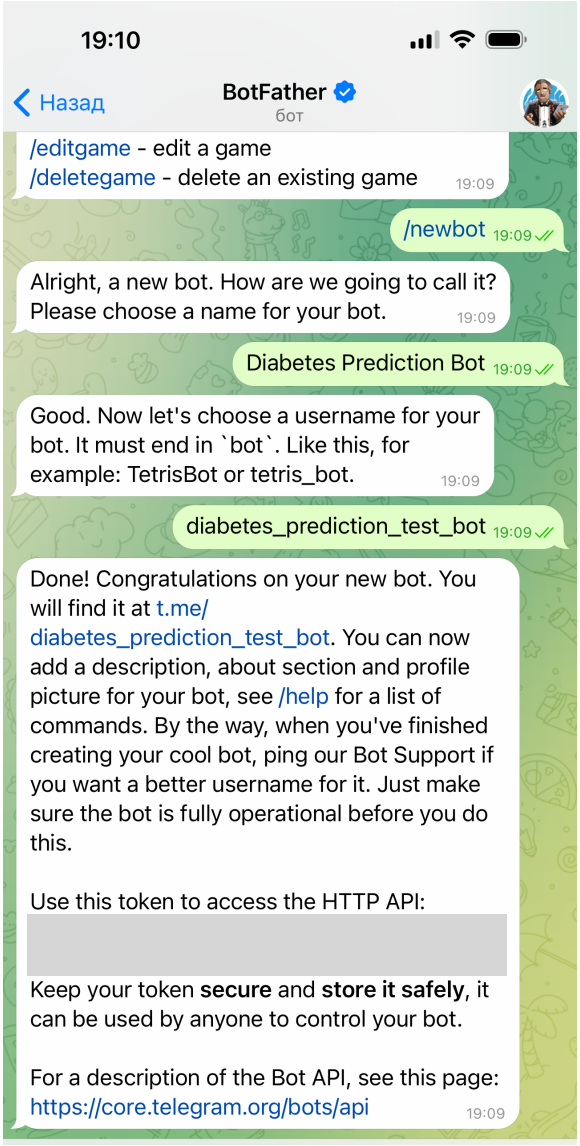


Рисунок 4.7 – Створення чат-бота

Чат-бот створено, але через відсутність функціонуючого backend він поки що не може виконувати жодні задачі та команди. Наступним кроком необхідно розробити backend з урахуванням скриптів повідомлень, які розроблено в попередньому розділі.

При створенні backend використано бібліотеку *python-telegram-bot.* Для запуску застосунку необхідно виконати наступний код:

```
if __name__ == "__main__":
    model = joblib.load("diabetes_prediction.joblib")
    user = User(model)
    start_bot(user)
```

Розроблено функції для обробки отримання текстового повідомлення та натискання на кнопки у випадку відповіді на питання з заздалегідь визначеними варіантами відповіді:

```
def start_bot(user):
    print("Bot is running...")

    updater = Updater(TOKEN, use_context=True)
    updater.dispatcher.add_handler(CommandHandler("start", start))
    updater.dispatcher.add_handler(MessageHandler(Filters.text, partial(text,
user=user)))
    updater.dispatcher.add_handler(CallbackQueryHandler(partial(button_click,
user=user)))
    updater.dispatcher.add_error_handler(error)

    updater.start_polling()
    updater.idle()
```

Важливо зазначити що у деякі функції передається об’єкт класу User. Даний об’єкт дозволяє зберегти та структурувати отримані від користувача дані, а також здійснити прогнозування за допомогою відповідного методу. Прогнозування здійснює заздалегідь навчене дерево рішень:

```
    def predict_diabetes(self):
        user_data = pd.DataFrame({
            "gender": [self.gender],
            "age": [self.age],
            "hypertension": [self.hypertension],
            "heart_disease": [self.heart_disease],
            "smoking_history": [self.smoking_history],
            "bmi": [self.weight / math.pow((self.height / 100), 2)],
            "HbA1c_level": [self.HbA1c]
        })

        return model.predict(user_data)[0]
```

В результаті прогнозування модель повертає число 1 (можлива наявність діабету 2-го типу) або 0 (людина не знаходиться в групі ризику). Текст програми представлено у додатку В.

4.4 Інструкції користувача інформаційної системи

Для запуску застосунка необхідно виконати команду «**/start**». Після виконання даної команди з’являються або питання з варіантами відповідей, або текстові повідомлення у відповідь на які треба написати текст. Кожне повідомлення містить інструкцію стосовно формату відповіді. Валідація значень відбувається за допомогою регулярних виразів, тому важливо правильно вказувати одиниці вимірювання. Приклади роботи застосунка продемонстровано на рисунках 4.8 – 4.9.

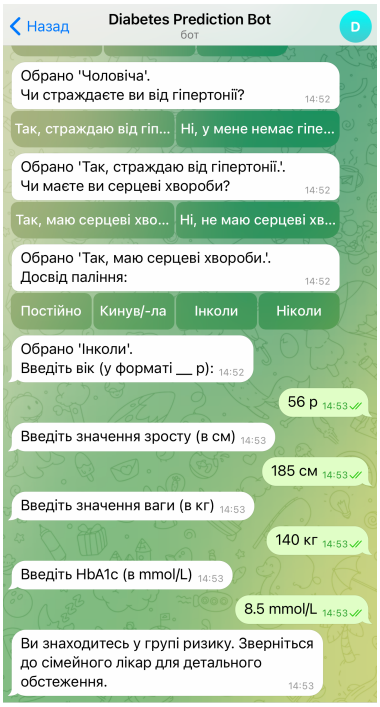


Рисунок 4.8 – Позитивний прогноз

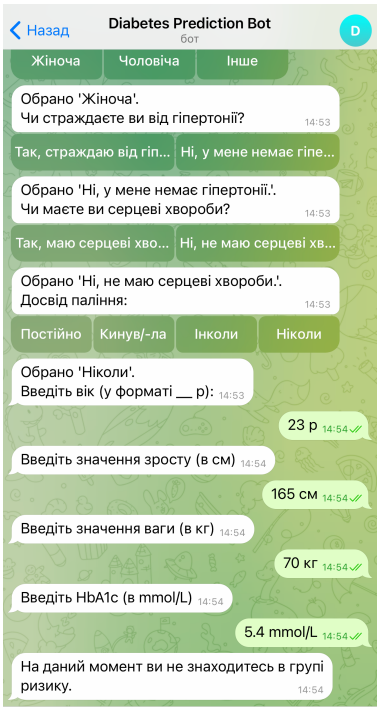


Рисунок 4.9 – Негативний прогноз

При помилках у форматі відповіді система повідомляє про помилку. Варто повторно ввести дані у правильному форматі для продовження роботи з ботом.

Висновки до розділу 4

В ході розробки четвертого розділу КРМ охарактеризовано процес програмної реалізації МІС. Система протестована та доповнена керівництвом користувача. Наведено приклади використання застосунка.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи магістра здійснено визначення ймовірності виникнення захворювання на основі факторів ризику. Досліджено та застосовано на практиці методи та засоби створення дерева рішення. Розроблено чат-бот для прогнозування ймовірності виникнення цукрового діабету другого типу. Отримання попередження про перебування у групі ризику мотивує звернутися до фахівця та здійснити Отже, вдалося вдосконалити процес профілактики захворювань за рахунок використання алгоритмів штучного інтелекту. Для досягнення поставленої мети виконано наступні завдання:

1) Проаналізовано предметну сферу розробки програмного забезпечення в ґалузі соціальних медицини.
2) Досліджено існуючи аналоги.
3) Обрано набір даних.
4) Здійснено специфікацію вимог.
5) Виконано моделювання системи за допомогою UML-діаграм.
6) Обрано алгоритм машинного навчання (дерева рішень).
7) Реалізовано програмне забезпечення МІС (Telegram-бот).

Практичне значення отриманих результатів полягає у тому що застосування чат-ботів для автоматизації медичних опитувань сприяє економії часу при ознайомленні зі скаргами пацієнтів при записі до лікаря. Застосування штучного інтелекту для прогнозування ймовірності виникнення захворювань сприяє вчасності профілактичних заходів. Розроблений застосунок може бути вдосконалений за допомогою більш детального набору даних для тренування моделі машинного навчання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Що таке Big Data? : Kyivstar Business Hub URL: https://hub.kyivstar.ua/articles/shho-take-big-data (дата звернення: 07.01.2024).

2. Kumar V. Healthcare Analytics Made Simple: Techniques in healthcare computing using machine learning and Python. Birmingham : Packt Publishing, 2018. 268 p.

3. What is a neural network? : IBM URL: https://www.ibm.com/topics/neural-networks (дата звернення: 07.01.2024).

4. Пацієнт має право: Що треба знати про право на медичну таємницю : Офіційний сайт МОЗ URL: https://moz.gov.ua/article/health/pacient-mae-pravo-scho-treba-znati-pro-pravo-na-medichnu-taemnicju (дата звернення: 07.12.2023).

5. Мельникова Н. І. Аналітичний огляд засобів програмного забезпечення в медичній галузі // Мельникова Н. І., Шаховська Н. Б. // Вісник національного університету «Львівська політехніка». – 2010. – №673. – С.146 – 154.

6. Чат-бот «Друг. Перша допомога» URL: https://t.me/friend_first_aid_bot (дата звернення: 07.01.2024).

7. What is machine learning? : IBM URL: https://www.ibm.com/topics/machine-learning (дата звернення: 09.01.2024).

8. Формалізована оцінка стану хворого за допомогою шкал при основних внутрішніх хворобах URL: http://dspace.zsmu.edu.ua/bitstream/123456789/938/1/Shkaly,%20Smtik%202015.pdf (дата звернення: 05.02.2024).

9. Що таке відкриті дані : «Дія. Відкриті дані» URL: https://diia.data.gov.ua/info-center/wodi (дата звернення: 07.01.2024).

10. Diabetes prediction dataset : Kaggle URL: https://www.kaggle.com/datasets/iammustafatz/diabetes-prediction-dataset (дата звернення: 08.01.2024).

11. Insulin, Medicines, & Other Diabetes Treatments : NIH URL: https://www.niddk.nih.gov/health-information/diabetes/overview/insulin-medicines-treatments (дата звернення: 09.01.2024).

12. Дерево: Wikipedia URL: https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_(%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2) (дата звернення: 05.02.2024).

13. Why Scientists Should Use Python for Scientific Computing: Hugo Bowne-Anderson URL: https://www.datacamp.com/blog/the-case-for-python-in-scientific-computing (дата звернення: 06.02.2024).

ДОДАТОК А

Програмна реалізація нечіткого дерева рішень

```
import pprint
import math
```

Навчальні приклади для побудови нечіткого дерев рішень:

```
diabetes_data = {
    "D1": {"age": 14, "bmi": 18, "diabetes": 0.05},
    "D2": {"age": 19, "bmi": 20, "diabetes": 0.1},
    "D3": {"age": 45, "bmi": 33, "diabetes": 1},
    "D4": {"age": 44, "bmi": 23, "diabetes": 0.2},
    "D5": {"age": 85, "bmi": 36, "diabetes": 1},
    "D6": {"age": 70, "bmi": 27, "diabetes": 0.7},
    "D7": {"age": 62, "bmi": 32, "diabetes": 0.9},
}
```

Множина факторів:

```
standard = {
    "age": {"дитячий": [0, 0, 16, 18],
            "дорослий": [18, 36, 55, 65],
            "похилий": [65, 70, 75, 85]},
    "bmi": {
          "недостатній": [14, 16, 17, 18],
          "нормальний": [18, 22, 24, 25],
          "надмірний": [25, 28, 30, 37],
    }
}
```

Формула для визначення приналежності прикладів до атрибутів:

```
def adherence(dataset, standard):
  keys = standard.keys()
  res = dict()

  for key in keys:
    category_res = dict()
    for category in standard[key]:
      a, b, c, d = standard[key][category]

      element_result = dict()
      for element_key in dataset.keys():
        x = dataset[element_key][key]

        mu = 0
        if x >= a and x <= b:
          try:
            mu = 1 - (b - x) / (b - a)
          except:
            pass
        if x >= b and x <= c:
          mu = 1
        if x >= c and x <= d:
          try:
            mu = 1 - (x - c)/(d - c)
          except:
            pass

        element_result[element_key] = mu
      category_res[category] = element_result
    res[key] = category_res

  return res
```

Застосування даної формули до наданого прикладу:

```
pp = pprint.PrettyPrinter(indent=2)
adherence_rate = adherence(diabetes_data, standard)
pp.pprint(adherence_rate )
```

```
    { 'age': { 'дитячий': { 'D1': 1,
                            'D2': 0,
                            'D3': 0,
                            'D4': 0,
                            'D5': 0,
                            'D6': 0,
                            'D7': 0},
               'дорослий': { 'D1': 0,
                             'D2': 0.05555555555555558,
                             'D3': 1,
                             'D4': 1,
                             'D5': 0,
                             'D6': 0,
                             'D7': 0.30000000000000004},
               'похилий': { 'D1': 0,
                            'D2': 0,
                            'D3': 0,
                            'D4': 0,
                            'D5': 0.0,
                            'D6': 1,
                            'D7': 0}},
      'bmi': { 'надмірний': { 'D1': 0,
                              'D2': 0,
                              'D3': 0.5714285714285714,
                              'D4': 0,
                              'D5': 0.1428571428571429,
                              'D6': 0.6666666666666667,
                              'D7': 0.7142857142857143},
               'недостатній': { 'D1': 0.0,
                                'D2': 0,
                                'D3': 0,
                                'D4': 0,
                                'D5': 0,
                                'D6': 0,
                                'D7': 0},
               'нормальний': { 'D1': 0.0,
                               'D2': 0.5,
                               'D3': 0,
                               'D4': 1,
                               'D5': 0,
                               'D6': 0,
                               'D7': 0}}}
```

Розрахунок загальної ентропії:

$$E(S^N) = -(\frac{4}{7} \cdot \log_2 \frac{4}{7} + \frac{3}{7} \cdot \log_2 \frac{3}{7}) = 0.99$$

```
def general_e(p_yes, p_no, p):
  return (-1) * (p_yes / p * math.log2(p_yes / p) + p_no / p * math.log2(p_no / p))
```

```
p_yes = 0
p_no = 0

for element_key in diabetes_data:
  data = diabetes_data[element_key]["diabetes"]
  p_yes += data
  p_no += 1 - data

p = p_yes + p_no
g_e = general_e(p_yes, p_no, p)

print(f"Значення P_yes: {p_yes}")
print(f"Значення P_no: {p_no}")
print(f"Загальне значення P: {p}")
print(f"Загальна ентропія: {g_e}")
```

```
    Значення P_yes: 3.9499999999999997
    Значення P_no: 3.0500000000000003
    Загальне значення P: 7.0
    Загальна ентропія: 0.988042611964058
```

Розрахунок нечітких показників різних атрибутів:

```
for factor in adherence_rate.keys():
  categories = adherence_rate[factor]

  e_factor = 0
  for category in categories.keys():
    print(category)
    p_yes = 0
    p_no = 0

    sum = 0

    for element_key in categories[category]:
      rate = diabetes_data[element_key]["diabetes"]
      mu = categories[category][element_key]

      p_yes += min(rate, mu)
      p_no += min(1 - rate, mu)
      sum += mu

    print(f"p_yes = {p_yes}")
    print(f"p_no = {p_no}")

    try:
      e = general_e(p_yes, p_no, p_yes + p_no)
      print(f"e_category = {e}")
    except:
      e = 0

    print(f"sum = {sum}")

    e_factor += sum / p * e

    print()

  print(f"Приріст інформації для '{factor}': {g_e - e_factor}\n")
```

```
дитячий
p_yes = 0.05
p_no = 0.95
e_category = 0.28639695711595625
sum = 1

дорослий
p_yes = 1.5555555555555556
p_no = 0.9555555555555556
e_category = 0.9584161691555606
sum = 2.355555555555555

похилий
p_yes = 0.7
p_no = 0.30000000000000004
e_category = 0.8812908992306927
sum = 1.0

Приріст інформації для 'age': 0.49871541365901473

недостатній
p_yes = 0.0
p_no = 0.0
sum = 0.0

нормальний
p_yes = 0.30000000000000004
p_no = 1.3
e_category = 0.6962122601251458
sum = 1.5

надмірний
p_yes = 2.0952380952380953
p_no = 0.4
e_category = 0.6350395330941944
sum = 2.0952380952380953

Приріст інформації для 'bmi': 0.6487744102627883
```

ДОДАТОК Б

Програмна реалізація методу дерева рішень на прикладі визначення ризику діабету другого типу

Імпорт необхідних бібліотек

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
import joblib
```

1. Попередній аналіз даних

```
df = pd.read_csv("diabetes_prediction_dataset.csv")
```

1.1 Приклад завантажених даних:

```
df.head()
```

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_g
0	Female	80.0	0	1	never	25.19	6.6	
1	Female	54.0	0	0	No Info	27.32	6.6	
2	Male	28.0	0	0	never	27.32	5.7	
3	Female	36.0	0	0	current	23.45	5.0	
4	Male	76.0	1	1	current	20.14	4.8	

1.2 Загальна інформація про набір даних:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100000 entries, 0 to 99999
Data columns (total 9 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   gender               100000 non-null  object 
 1   age                  100000 non-null  float64
 2   hypertension         100000 non-null  int64  
 3   heart_disease        100000 non-null  int64  
 4   smoking_history      100000 non-null  object 
 5   bmi                  100000 non-null  float64
 6   HbA1c_level          100000 non-null  float64
 7   blood_glucose_level  100000 non-null  int64  
 8   diabetes             100000 non-null  int64  
dtypes: float64(3), int64(4), object(2)
memory usage: 6.9+ MB
```

1.3 Кількість пустих комірок:

```
df.isna().sum()
```

```
gender                 0
age                    0
hypertension           0
heart_disease          0
smoking_history        0
bmi                    0
HbA1c_level            0
blood_glucose_level    0
diabetes               0
dtype: int64
```

```
df = df.dropna()
```

1.4 Унікальні значення

```
df["smoking_history"].unique()
```

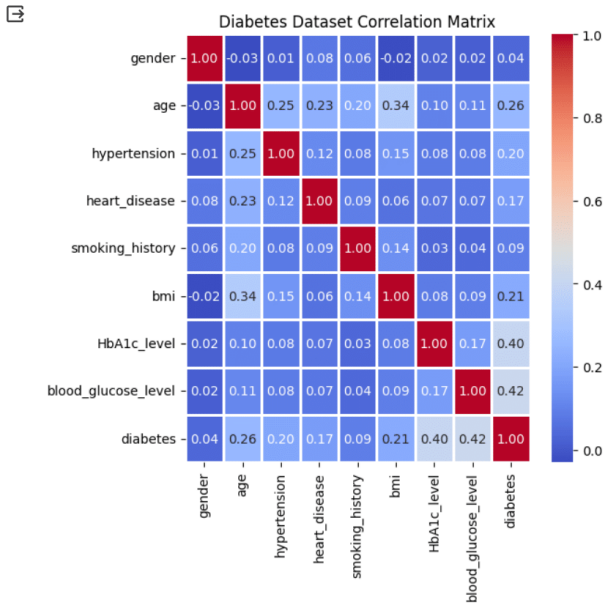
```
array(['never', 'No Info', 'current', 'former', 'ever', 'not current'],
      dtype=object)
```

```
df["gender"].unique()
```

```
array(['Female', 'Male', 'Other'], dtype=object)
```

1.5 Кореляція між параметрами

```
correlation_matrix = df.corr()
plt.figure(figsize=(5, 5))
sb.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=1)
plt.title('Diabetes Dataset Correlation Matrix')
plt.show()
```



2. Трансформація даних

```
def smoking_history_binary(smoking_history):
  yes = ['current', 'former', 'ever']
  no = ['never', 'No Info',  'not current']

  if smoking_history in yes:
      return 1
  else:
      return 0

df['smoking_history'] = df.apply(lambda row: smoking_history_binary(row['smoking_history']), axis=1)
```

```
label_encoder = LabelEncoder()

# Encode "gender" parameter
df["gender"] = label_encoder.fit_transform(df["gender"])
mapping = dict(zip(label_encoder.classes_, range(len(label_encoder.classes_))))
print(f"Gender dict: {mapping}")
```

```
Gender dict: {'Female': 0, 'Male': 1, 'Other': 2}
```

```
df.head()
```

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_g
0	0	80.0	0	1	0	25.19	6.6	
1	0	54.0	0	0	0	27.32	6.6	
2	1	28.0	0	0	0	27.32	5.7	
3	0	36.0	0	0	1	23.45	5.0	
4	1	76.0	1	1	1	20.14	4.8	

3. Машинне навчання

3.1 Розділення даних на навчальний та тестовий набір

```
X = df[["gender", "age", "hypertension", "heart_disease", "smoking_history", "bmi", "HbA1c_level"]]
y = df["diabetes"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

3.2 Підбір оптимальних гіперпараметрів

```
param_grid = {
    'max_depth': range(7, 20),
    'random_state': [20]
}

clf = DecisionTreeClassifier()
grid_search = GridSearchCV(estimator=clf, param_grid=param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

print("Best parameters:", grid_search.best_params_)
```

```
Best parameters: {'max_depth': 7, 'random_state': 20}
```

3.3 Навчання моделі

```
model = grid_search.best_estimator_
model.fit(X_train, y_train)
```

```
DecisionTreeClassifier(max_depth=7, random_state=20)
```

3.4 Обчислення точності моделі

```
predictions = model.predict(X_test)
score = accuracy_score(y_test, predictions) * 100
print(f"Точність: {score:.2f}%")
```

```
Точність: 95.49%
```

3.5 Прогнозування

```
def predict_diabetes(gender, age, hypertension, heart_disease, smoking_history, bmi, HbA1c_level):
  user_data = pd.DataFrame({
    "gender": [gender],
    "age": [age],
    "hypertension": [hypertension],
    "heart_disease": [heart_disease],
    "smoking_history": [smoking_history],
    "bmi": [bmi],
    "HbA1c_level": [HbA1c_level],
  })

  prediction = model.predict(user_data)[0]

  if prediction:
    return "Yes"
  else:
    return "No"
```

```
predict_diabetes(1, 65, 1, 1, 1, 30, 8.5)
```

```
'Yes'
```

3.6 Збереження створеної моделі

```
joblib.dump(model, "diabetes_prediction.joblib")
```

```
['diabetes_prediction.joblib']
```

ДОДАТОК В

Текст програми чат-бота

```
import math
import re
import pandas as pd
import joblib
from functools import partial
from telegram import InlineKeyboardMarkup, InlineKeyboardButton
from telegram.ext import Updater, CommandHandler, MessageHandler, Filters,
CallbackQueryHandler
from settings import *

# Dictionary
genders = {"Жіноча": 0, "Чоловіча": 1, "Інше": 2}
hypertension = {"Так, страждаю від гіпертонії.": 1, "Ні, у мене немає
гіпертонії.": 0}
heart_disease = {"Так, маю серцеві хвороби.": 1, "Ні, не маю серцеві хвороби.": 0}
smoking = {"Постійно": 1, "Кинув/-ла": 0, "Інколи": 1, "Ніколи": 0}

# MENU
gender_keyboard = [[InlineKeyboardButton(key, callback_data=key) for key in
genders]]
hypertension_keyboard = [[InlineKeyboardButton(key, callback_data=key) for key in
hypertension]]
heart_disease_keyboard = [[InlineKeyboardButton(key, callback_data=key) for key in
heart_disease]]
smoking_keyboard = [[InlineKeyboardButton(key, callback_data=key) for key in
smoking]]


# Збереження даних про користувача
class User:
    def __init__(self, model):
        self.model = model

        self.gender = None
        self.age = None
        self.hypertension = None
        self.heart_disease = None
        self.smoking_history = None
        self.height = None
        self.weight = None
        self.HbA1c = None

    def predict_diabetes(self):
        user_data = pd.DataFrame({
            "gender": [self.gender],
            "age": [self.age],
            "hypertension": [self.hypertension],
            "heart_disease": [self.heart_disease],
            "smoking_history": [self.smoking_history],
            "bmi": [self.weight / math.pow((self.height / 100), 2)],
            "HbA1c_level": [self.HbA1c]
        })

        return model.predict(user_data)[0]
```

```
def start(update, context):
    first_message = "Вітаю! Я допоможу вам визначити чи знаходитесь ви у групі
ризику " \
                    "\nОберіть стать, будь ласка:"
    update.message.reply_text(first_message,
                              reply_markup=InlineKeyboardMarkup(gender_keyboard))


def button_click(update, context, user=None):
    query = update.callback_query
    query.answer()
    answer = query.data

    message = None
    markup = None

    if answer in genders.keys():
        user.gender = genders[answer]
        message = f"Обрано '{answer}'.\nЧи страждаєте ви від гіпертонії?"
        markup = InlineKeyboardMarkup(hypertension_keyboard)
    elif answer in hypertension.keys():
        user.hypertension = hypertension[answer]
        message = f"Обрано '{answer}'.\nЧи маєте ви серцеві хвороби?"
        markup = InlineKeyboardMarkup(heart_disease_keyboard)
    elif answer in heart_disease.keys():
        user.heart_disease = heart_disease[answer]
        message = f"Обрано '{answer}'.\nДосвід паління:"
        markup = InlineKeyboardMarkup(smoking_keyboard)
    elif answer in smoking.keys():
        user.smoking_history = smoking[answer]
        message = f"Обрано '{answer}'.\nВведіть вік (у форматі __ р):"

    update.callback_query.message.reply_text(message, reply_markup=markup)


def text(update, context, user=None):
    answer = update.message.text
    answer_numeric = float(re.findall(r'\d+\.\d+|\d+', answer)[0])
    message = None

    age = re.compile("^[0-9]{1,3} р$")
    if age.match(answer):
        user.age = answer_numeric
        message = "Введіть значення зросту (в см)"

    height = re.compile("^[0-9]{1,3} см$")
    if height.match(answer):
        user.height = answer_numeric
        message = "Введіть значення ваги (в кг)"

    weight = re.compile("^[0-9]{1,3} кг$")
    if weight.match(answer):
        user.weight = answer_numeric
        message = "Введіть HbA1c (в mmol/L)"

    glucose = re.compile("^[+-]?([0-9]*[.])?[0-9]+ mmol/L$")
    if glucose.match(answer):
```

```
        user.HbA1c = answer_numeric
        prediction = user.predict_diabetes()

        if prediction:
            message = "Ви знаходитесь у групі ризику. " \
                      "Зверніться до сімейного лікар для детального обстеження."
        else:
            message = "На даний момент ви не знаходитесь в групі ризику."

    if not(age.match(answer) or weight.match(answer) or height.match(answer) or
glucose.match(answer)):
        message = "Неприпустимий формат відповіді!"

    update.message.reply_text(message)


def error(update, context):
    print(f"Error! {context.error}")


def start_bot(user):
    print("Bot is running...")

    updater = Updater(TOKEN, use_context=True)
    updater.dispatcher.add_handler(CommandHandler("start", start))
    updater.dispatcher.add_handler(MessageHandler(Filters.text, partial(text,
user=user)))
    updater.dispatcher.add_handler(CallbackQueryHandler(partial(button_click,
user=user)))
    updater.dispatcher.add_error_handler(error)

    updater.start_polling()
    updater.idle()


if __name__ == "__main__":
    model = joblib.load("diabetes_prediction.joblib")
    user = User(model)
    start_bot(user)
```