

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «магістр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)
на тему:

**«Інтелектуальна система прогнозування діагнозів та надання
рекомендацій для домедичної допомоги»**

Виконав студент групи ПЗм-24-1
КОРМИШ Роман Іванович

Керівник роботи:
ЧИЖМОТРЯ Олена Геннадіївна

Рецензент:
ЛОБАНЧИКОВА Надія Миколаївна

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

зав. кафедри інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«18» вересня 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **КОРМИШ Роман Іванович**

Керівник роботи: **ЧИЖМОТРЯ Олена Геннадіївна**

Тема роботи: **«Інтелектуальна система прогнозування діагнозів та надання рекомендацій для домедичної допомоги»**, затверджена Наказом закладу вищої освіти від «18» вересня 2025 р. № 491/с.

Вихідні дані для роботи: **алгоритми наївного байєсівського класифікатора для оцінки діагнозів, система аналітики з візуалізацією метрик і кореляцій; технології створення мобільних додатків для офлайн-діагностики за симптомами, інтеграція геолокації та аналітика ймовірнісних моделей у медичній інформатиці.**

Керівники / Консультанти з магістерської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Керівник / Консультант	Завдання видав	Завдання прийняв
1	Чижмотря О.Г.	19.09.2025р	19.09.2025р
2	Чижмотря О.Г.	19.09.2025р	19.09.2025р
3	Чижмотря О.Г.	19.09.2025р	19.09.2025р

РЕФЕРАТ

Кваліфікаційна робота освітнього ступеня “магістр” присвячена прогнозуванню діагнозів та надання рекомендацій для домедичної допомоги з офлайн-режимом на основі наївного байєсівського класифікатора. Пояснювальна записка складається з вступу, трьох та списку використаної літератури. Пояснювальна записка до кваліфікаційної магістерської роботи містить 64 сторінки, 18 ілюстрацій, 1 додаток та 41 посилання на зовнішні джерела.

Метою даної роботи є аналіз принципів побудови та застосування мереж Байєса для задач медичної діагностики, а також створення класифікатора, здатного визначати ймовірний діагноз на основі вхідних симптомів.

В документі описано процес розробки додатку, який аналізує введені користувачем симптоми за допомогою графічного інтерфейсу та надає рекомендації з домедичної допомоги в умовах екстрених невідкладних ситуацій. Досліджено та адаптовано наївний класифікатор Байєса під цілі кваліфікаційної роботи.

КЛЮЧОВІ СЛОВА: НАЇВНИЙ БАЙЄСІВСЬКИЙ КЛАСИФІКАТОР, ДОМЕДИЧНА ДОПОМОГА, ОФЛАЙН-ДІАГНОСТИКА, РЕКОМЕНДАЦІЇ ПО ДОПОМОЗІ, МОБІЛЬНИЙ ЗАСТОСУНОК.

ABSTRACT

The master's thesis is devoted to predicting diagnoses and providing recommendations for pre-medical care in offline mode based on a naive Bayesian classifier. The explanatory note consists of an introduction, three chapters, and a list of references. The explanatory note to the master's thesis contains 64 pages, 18 illustrations, 1 appendix, and 41 references to external sources.

The purpose of this work is to analyze the principles of constructing and applying Bayesian networks for medical diagnosis tasks, as well as to create a classifier capable of determining a probable diagnosis based on input symptoms.

The document describes the process of developing an application that analyzes user-entered symptoms using a graphical interface and provides recommendations for pre-medical care in emergency situations. A naive Bayesian classifier was researched and adapted for the purposes of the qualification work.

KEYWORDS: NAIVE BAYESIAN CLASSIFIER, PRE-MEDICAL CARE, OFFLINE DIAGNOSTICS, CARE RECOMMENDATIONS, MOBILE APPLICATION.

ЗМІСТ

ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ АСИСТЕНТА ДОМЕДИЧНОЇ ДОПОМОГИ.....	10
1.1 Аналіз предметної області.....	10
1.2 Аналіз існуючих сервісів та приклади офлайн-додатків домедичної допомоги.....	12
1.3 Методи і алгоритми інтелектуального аналізу даних для рішення.....	17
1.4 Постановка задачі.....	19
Висновки до першого розділу.....	20
РОЗДІЛ 2. ПРОЄКТУВАННЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ДОМЕДИЧНОЇ ДОПОМОГИ.....	22
2.1. Проєктування функціональності додатку.....	22
2.2 Характеристика джерела даних для проведення аналізу.....	25
2.3 Побудова моделі аналізу і практична реалізація.....	30
2.4 Побудова модуля аналітики роботи моделі.....	34
Висновки до другого розділу.....	37
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ДОДАТКОМ.....	38
3.1. Порядок роботи з інтерфейсом додатку.....	38
3.2 Оцінка та аналіз роботи класифікатора.....	44
Висновки до третього розділу.....	51
ВИСНОВКИ.....	52
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	54
ДОДАТКИ.....	60

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

NBC - Наївний байєсівський класифікатор (Naive Bayesian Classifier)

EFA - Асистент домедичної допомоги (Emergency First Aid Assistant)

$P(D|S)$ - Ймовірність діагнозу за симптомами (Probability of Diagnosis given Symptoms)

S - Симптоми (Symptoms)

D - Діагноз (Diagnosis)

SV - Тяжкість симптому (Symptom Severity)

CV - Перехресна валідація (Cross-Validation)

ACC - Точність моделі (Accuracy)

CONF - Впевненість прогнозу (Confidence)

CORR - Кореляційний коефіцієнт (Correlation Coefficient)

ВСТУП

У сучасних умовах цифрової трансформації охорони здоров'я, ефективність діагностики захворювань значною мірою залежить від здатності медичних працівників аналізувати великі обсяги даних, враховувати ймовірності розвитку хвороб за наявності певних симптомів, а також приймати обґрунтовані клінічні рішення в умовах невизначеності. У зв'язку з цим особливу актуальність набувають методи штучного інтелекту, зокрема, мережі Байєса, які дозволяють формалізувати процес логічного прогнозування на основі ймовірнісних залежностей між медичними показниками та можливими діагнозами.

Мережі Байєса є інструментом представлення знань у вигляді графової структури, де вершини відповідають змінним (ознакам, симптомам, діагнозам), а ребра залежностям між ними. Вони дозволяють обробляти неповні або частково суперечливі дані, що є характерним для клінічної практики. Завдяки здатності до оновлення ймовірностей у реальному часі в залежності від нових даних, дані мережі можуть виступати в ролі інтелектуального помічника лікаря, сприяючи підвищенню точності та швидкості постановки діагнозу.

Актуальність теми дослідження обумовлена необхідністю підвищення якості медичної допомоги шляхом впровадження інноваційних цифрових рішень. Розробка інтелектуальних діагностичних систем на основі мереж Байєса є перспективним напрямом, який поєднує методи машинного навчання, статистики та медичної інформатики.

Метою даної роботи є аналіз принципів побудови та застосування мереж Байєса для задач медичної діагностики, а також створення класифікатора, здатного визначати ймовірний діагноз на основі вхідних симптомів. Встановлена мета обумовлює такі завдання:

- провести аналіз існуючих підходів до офлайн-діагностики та ймовірнісного моделювання в медицині;
- дослідити ефективність наївного байєсівського класифікатора для задач класифікації діагнозів за симптомами та тяжкістю;

- розробити мобільний додаток з офлайн-режимом, геолокаційними функціями та рекомендаційною системою;
- створити аналітичний модуль для оцінки точності, кореляцій і впевненості прогнозів;
- оцінити ефективність системи через тестування на медичних датасетах та симуляцію реальних сценаріїв.

Об'єктом дослідження є технології розробки мобільних додатків для офлайн-медичної діагностики, інтеграція геолокаційних сервісів та аналітика ймовірнісних моделей у медичній інформатиці.

Предметом дослідження є моделі, методи та алгоритми наївного байєсівського класифікатора для оцінки діагнозів, система аналітики з візуалізацією метрик, архітектура офлайн-додатку з автоматичним визначенням мови та номера екстреної допомоги.

У кваліфікаційній роботі використано такі методи: ймовірнісне моделювання для оцінки діагнозів; аналіз кореляцій для виявлення зв'язків між симптомами; візуалізацію даних для створення теплових карт і матриць; мобільну розробку для реалізації офлайн-функціоналу та інтерфейсу.

Наукова новизна роботи полягає у застосуванні наївного байєсівського класифікатора для офлайн-діагностики з урахуванням тяжкості симптомів, розробці інтегрованої системи аналітики з автоматичними рекомендаціями щодо вдосконалення моделі, а також адаптації геолокаційних функцій для екстрених викликів у різних регіонах.

Практична цінність роботи полягає у створенні повнофункціонального мобільного додатку, який працює без інтернету, надає науково обґрунтовані рекомендації з домедичної допомоги та підвищує шанси на виживання в критичних ситуаціях. Система аналітики дозволяє розробникам постійно вдосконалювати модель, забезпечуючи її надійність і точність.

Апробація результатів роботи. Матеріали кваліфікаційної роботи доповідалися на VIII Всеукраїнській науково-технічній конференції

“Комп’ютерні технології: інновації, проблеми, рішення”, 02-03 грудня 2025 року, Житомир. [1][2]

РОЗДІЛ 1. АНАЛІЗ НАПРЯМІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ АСИСТЕНТА ДОМЕДИЧНОЇ ДОПОМОГИ

1.1 Аналіз предметної області

Інтелектуальні системи домедичної допомоги є важливим цифровим інструментом, призначеним для надання невідкладної підтримки в екстрених ситуаціях, коли професійна медична допомога недоступна, а інтернет-з'єднання відсутнє [3]. Така система виконує функцію інтелектуального посередника між людиною без медичної підготовки та науково обґрунтованими протоколами дій, дозволяючи швидко оцінити стан постраждалого, визначити найбільш ймовірний діагноз на основі введених симптомів і їх тяжкості, а також отримати чіткі, покрокові рекомендації щодо першої допомоги, усе це в умовах повної автономності від зовнішніх мереж. Завдяки цьому досягається скорочення часу реагування, зниження рівня паніки та суттєве підвищення ймовірності збереження життя до прибуття рятувальних служб.

Домедична допомога, як перший ешелон медичного втручання, являє собою чітко структурований набір дій, що виконуються аматорами відповідно до міжнародно визнаних стандартів, таких як рекомендації Європейської ради реанімації (ERC) та Американської асоціації серця (AHA) [4]. Вона охоплює первинну оцінку небезпеки на місці події, перевірку свідомості та дихання, зупинку зовнішньої кровотечі, відкриття дихальних шляхів, проведення серцево-легеневої реанімації, використання автоматичного зовнішнього дефібрилятора, розміщення постраждалого в стабільне бічне положення, а також боротьбу з шоком і гіпотермією. Успіх цих дій безпосередньо залежить від швидкості, точності діагностики та правильності виконання, параметри, які в умовах ізоляції від медичних установ і зв'язку неможливо забезпечити без технологічної підтримки.

Сучасні цифрові технології трансформують домедичну допомогу з рівня статичних інструкцій, паперових посібників чи простих офлайн-додатків із

текстовими описами [5]. Центральним елементом таких систем є використання ймовірнісних моделей, зокрема наївного байєсівського класифікатора, який не просто перелічує можливі стани, але ранжує їх за ймовірністю, враховуючи не лише наявність симптому, але й його інтенсивність. Такий підхід дозволяє розрізняти, наприклад, слабкий головний біль, типовий для напруги чи зневоднення, від помірного чи сильного, що може вказувати на різні причини, і відповідно пропонувати різні дії: від відпочинку й гідратації до негайного виклику допомоги.

Предметна область охоплює кілька взаємопов'язаних напрямів. Перший, це медичні протоколи домедичної допомоги, формалізовані в міжнародних і національних стандартах, що визначають послідовність дій у критичних станах. Другий, симптоматична діагностика, тобто процес перетворення суб'єктивних і об'єктивних клінічних проявів на структуровані дані, придатні для автоматичної обробки. Третій, ймовірнісне моделювання, яке дозволяє обчислювати апостеріорну ймовірність діагнозу за набором симптомів і їх тяжкості. Четвертий, офлайн-архітектура мобільних систем, що забезпечує локальне зберігання бази знань, виконання всіх обчислень на пристрої та кешування геолокаційних даних [6]. П'ятий, геолокаційна адаптація, що автоматично визначає мову інтерфейсу, номер екстреної служби та, за наявності зв'язку, ініціює виклик. Нарешті, шостий, аналітика моделі, що включає оцінку точності, впевненості прогнозів, покриття симптомів і кореляційних зв'язків між ознаками.

Симптом у системі трактується як клінічний прояв, суб'єктивний, як біль чи нудота, або об'єктивний, як набряк чи ціаноз, і кодується з урахуванням категорії, типу та тяжкості за шкалою з трьома рівнями: слабкий ($SV=1$), помірний ($SV=2$) та сильний ($SV=3$). Діагноз, у свою чергу, є ймовірнісною міткою з наявним рівнем критичності, переліком рекомендацій і рівнем доказовості [7].

Для оцінки ефективності системи використовується комплекс показників: ймовірність діагнозу, середня впевненість моделі, точність у крос-валідації,

відсоток симптомів, що були протестовані, коефіцієнти кореляції між ознаками та відповідність введеної тяжкості очікувань.

Методологічною основою є наївний байєсівський класифікатор, оптимальний для офлайн-середовища завдяки низькій обчислювальній складності, мінімальним вимогам до пам'яті та можливості роботи з невеликими навчальними вибірками. Додатково застосовуються техніки згладжування Лапласа, вагові коефіцієнти для критичних симптомів і rule-based фільтри для станів із високою специфічністю [8]. Цільова аудиторія охоплює туристів, альпіністів, водіїв, батьків, волонтерів-рятувальників і розробників, для перших важлива простота і зрозумілість, для других, глибока аналітика й можливості вдосконалення моделі.

Таким чином, асистент домедичної допомоги є не довідником, а повноцінною системою підтримки рішень у критичних умовах, що інтегрує медичні протоколи, ймовірнісне моделювання, офлайн-технології, геолокаційну адаптацію та аналітику, забезпечуючи швидке, точне та доступне реагування там, де кожна секунда має вирішальне значення.

1.2 Аналіз існуючих сервісів та приклади офлайн-додатків домедичної допомоги

Офлайн-додатки домедичної допомоги є невід'ємною частиною сучасної цифрової екстреної медицини, забезпечуючи автономне надання першої допомоги в умовах відсутності інтернет-з'єднання. Задача створення надійного, швидкого та точного інструменту для непрофесіоналів стає критично важливою в ситуаціях, коли кожна секунда має значення, а доступ до медичних фахівців чи онлайн-ресурсів неможливий [9]. Зі зростанням популярності активного відпочинку, подорожей у віддалені регіони та усвідомленням важливості базових навичок порятунку, на ринку з'являються численні мобільні рішення, які пропонують структуровані протоколи дій на основі введених симптомів чи обраного сценарію. Розглянемо найбільш відомі та поширені приклади таких систем, їх функціонал, технічні особливості, переваги та обмеження.

First Aid by American Red Cross (рис. 1.1) є одним із найавторитетніших і найпоширеніших офлайн-додатків у світі, розробленим Американським Червоним Хрестом у співпраці з медичними експертами. Додаток доступний для iOS та Android і працює повністю автономно після завантаження. Він містить детальні інструкції з домедичної допомоги за понад 40 типовими сценаріями: від зупинки кровотечі й опіків до серцевого нападу, утоплення та анафілаксії. Користувач обирає тип надзвичайної ситуації з візуального меню або вводить ключові симптоми через чекліст.

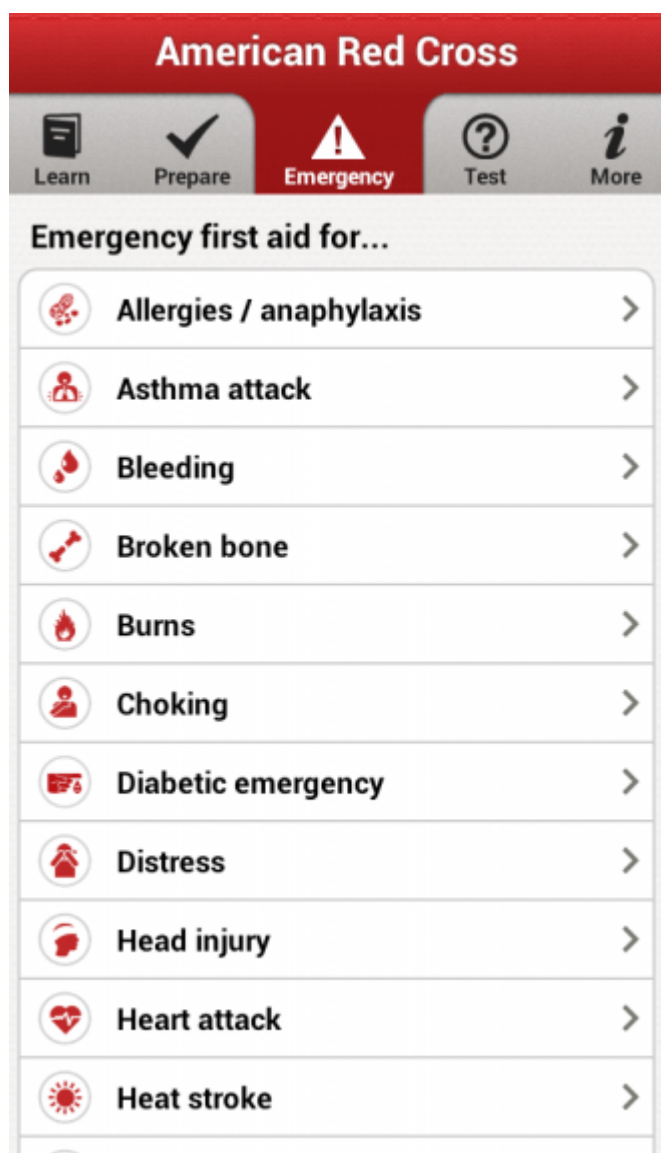


Рис. 1.1. Інтерфейс First Aid by American Red Cross

Система миттєво видає покрокові інструкції, доповнені ілюстраціями, короткими відео та голосовими підказками. Усі дані зберігаються локально в

зашифрованому вигляді, що забезпечує конфіденційність і швидкий доступ. Додаток підтримує англійську, іспанську та французьку мови, але не має автоматичного визначення мови за геолокацією. Важливою особливістю є інтеграція з 911 та при наявності зв'язку користувач може одним натисканням ініціювати виклик. Проте система не використовує ймовірнісні моделі, не враховує тяжкість симптомів і не ранжує діагнози.

St. John Ambulance First Aid (рис. 1.2) це офіційний додаток британської благодійної організації St. John Ambulance, що спеціалізується на підготовці волонтерів з домедичної допомоги.

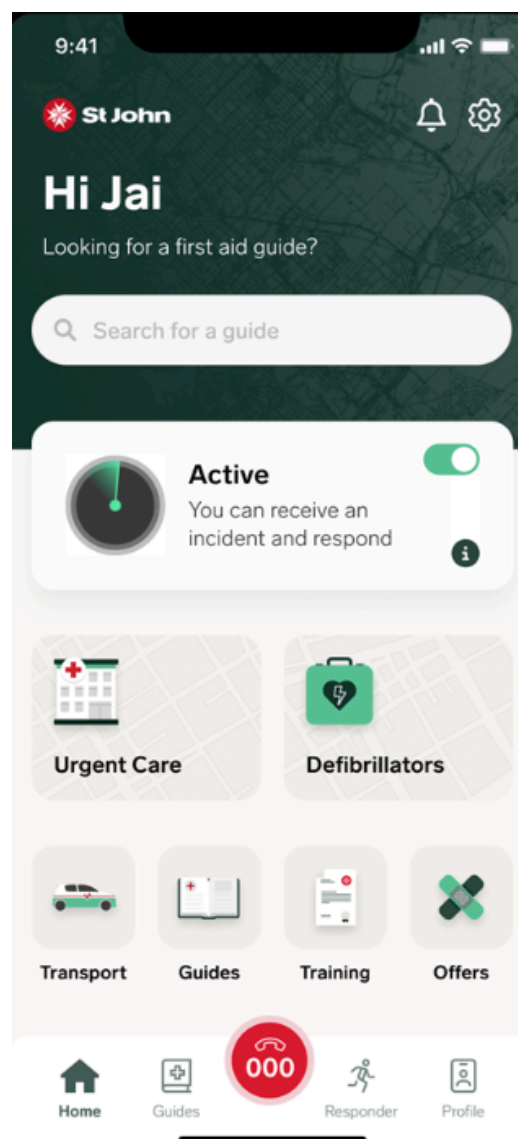


Рис. 1.2. Інтерфейс St. John Ambulance First Aid

Додаток доступний безкоштовно, займає близько 80 МБ і працює повністю офлайн [10]. Він охоплює понад 50 клінічних сценаріїв,

структурованих за алгоритмом DRCAB, з детальними інструкціями, діаграмами та анімаціями. Користувач проходить інтерактивний тріаж: відповідає на послідовні запитання («Чи реагує постраждалий?», «Чи є дихання?», «Чи є кровотеча?»), після чого система формує чіткий план дій. Додаток підтримує англійську мову, має вбудовану функцію підготовки до іспиту з домедичної допомоги та можливість зберігання нотаток. Геолокаційна адаптація відсутня, як і виклик екстреної служби. Система не передбачає введення тяжкості симптомів, не ранжує діагнози за ймовірністю і не має аналітичного модуля це суворий, клінічно валідований довідник, але без інтелектуальної обробки даних.

Аналогічний підхід демонструють British Red Cross Baby and Child First Aid, Australian Red Cross First Aid та Canadian Red Cross First Aid, усі вони пропонують локалізовані версії з національними протоколами, відео та аудіо, але залишаються в межах rule-based систем. В Україні подібну роль виконує додаток «Домедична допомога» від МОЗ України, який містить офіційні протоколи, затверджені наказами МОЗ, з інструкціями українською мовою.

Ada Health (рис. 1.3) це один із найпопулярніших і найрозвиненіших симптом-чекерів у світі, доступний як мобільний додаток для iOS та Android, а також веб-версія.

Розроблений у 2016 році компанією Ada Health GmbH у Берліні, додаток позиціонує себе не просто як інструмент перевірки симптомів, а як повноцінну систему самодопомоги та моніторингу здоров'я, яка використовує штучний інтелект для імітації клінічного тріажу [11]. Користувач створює профіль, вводячи демографічні дані (вік, стать, медичну історію), після чого чатбот запитує про симптоми, їх тривалість, тяжкість і супутні фактори. Система базується на байєсівському ймовірнісному моделюванні, подібному до наївного байєсівського класифікатора, з урахуванням понад 10 000 симптомів і 5 000 умов, що дозволяє генерувати персоналізований список можливих діагнозів з відсотками ймовірності. Наприклад, при введенні «біль у грудях» + «задишка» (середня тяжкість) Ada може запропонувати варіанти від «панічні атаки» (70%)

до «інфаркту» (15%), з рекомендаціями: від самодопомоги до негайного виклику швидкої.

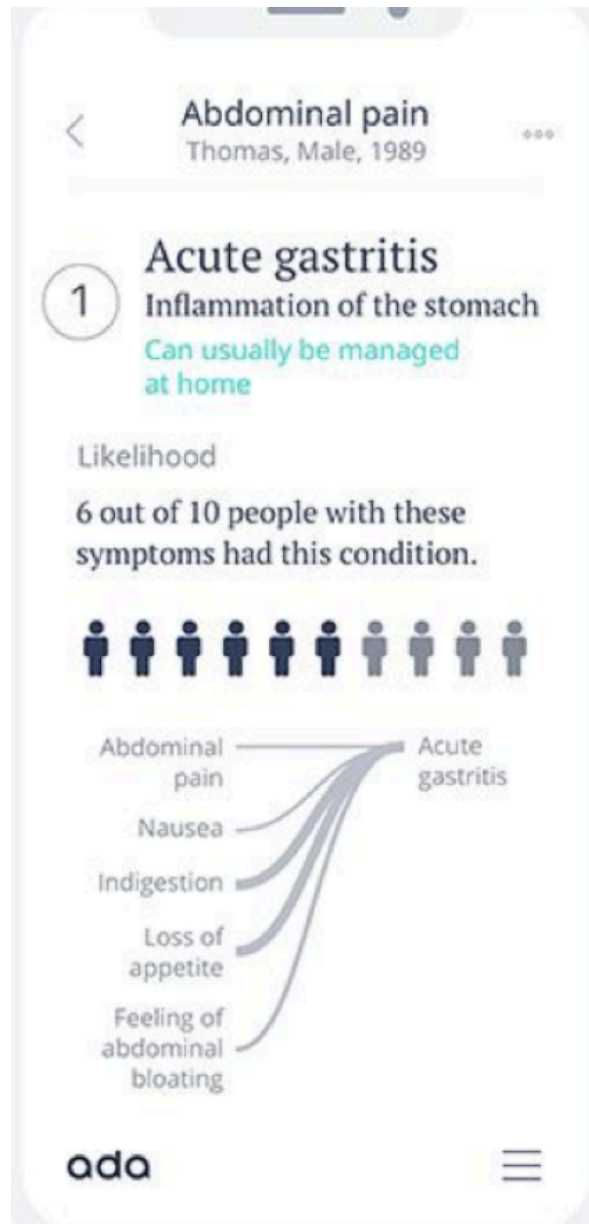


Рис. 1.3. Інтерфейс чатбота та список діагнозів з ймовірностями в Ada Health

Додаток підтримує 7 мов (англійська, німецька, французька, іспанська, португальська, румунська, суахілі), включає калькулятор ВМІ, статті від лікарів, трекінг симптомів у часі та створення профілів для родичів. Проте Ada вимагає інтернету для повної роботи (онлайн-оновлення бази), не підтримує повний офлайн-режим, і не має геолокаційної інтеграції для автоматичного виклику служб чи локалізації номерів. Відсутня вбудована аналітика моделі для розробників, а також візуалізація кореляцій.

Таким чином, аналіз існуючих рішень показує, що сучасні офлайн-додатки забезпечують надійні, клінічно перевірені інструкції, але мають суттєві обмеження: відсутність ймовірнісного ранжування діагнозів, ігнорування тяжкості симптомів, статичність рекомендацій, відсутність автоматичної локалізації, геолокаційної адаптації та аналітики моделі. Це створює потребу в новому поколінні асистентів — інтелектуальних, автономних, адаптивних і аналітично орієнтованих системах, які поєднують клінічну точність із можливостями штучного інтелекту.

1.3 Методи і алгоритми інтелектуального аналізу даних для рішення

Для розв'язання задачі інтелектуального аналізу даних, сформульованої в межах даної курсової роботи, було обрано застосування методів машинного навчання, що забезпечують ефективне виявлення закономірностей та структури у великих обсягах інформації. Інтелектуальний аналіз даних (ІАД) [12] передбачає комплекс автоматизованих процедур, спрямованих на отримання корисних знань із даних, що є критично важливим для підтримки процесів прийняття рішень у різноманітних прикладних сферах.

У сучасній науці та практиці застосовується широкий спектр алгоритмів машинного навчання, зокрема дерева рішень, метод опорних векторів (Support Vector Machine, SVM), нейронні мережі, асоціативні правила, ансамблеві методи (наприклад, випадковий ліс, градієнтний бустинг) тощо. Проте, з урахуванням специфіки вхідних даних, вимог до швидкості навчання та класифікації, а також простоти впровадження, у даній роботі вибрано найвний байєсівський класифікатор (НБК) як оптимальний інструмент.

Наївний байєсівський класифікатор [13] є базовим представником статистичних методів машинного навчання, який базується на теоремі Байєса, а саме фундаментальному принципі теорії ймовірностей, що дозволяє обчислити умовну ймовірність належності об'єкта до певного класу з урахуванням спостережуваних ознак. Відмінною характеристикою цього алгоритму є припущення про умовну незалежність ознак, яке значно спрощує

обчислювальні процеси, зберігаючи при цьому достатній рівень точності у багатьох практичних задачах.

Формалізовано це припущення можна представити так: нехай $x = (x_1, x_2, \dots, x_n)$ — вектор ознак об'єкта, а $c \in C$ — множина можливих класів. Тоді згідно з теоремою Байєса:

$$P(c|x) = \frac{P(c) \cdot P(x|c)}{P(x)},$$

де $P(c|x)$ — апостеріорна ймовірність класу c за умови спостереження ознак x , $P(c)$ — апіорна ймовірність класу, $P(x|c)$ — ймовірність появи ознак x за умови класу c , а $P(x)$ — ймовірність появи ознак x , що є сталим множником при порівнянні класів і тому не враховується в процедурі класифікації.

Припущення про незалежність ознак формалізується як:

$$P(x|c) = \prod_{i=1}^n P(x_i|c),$$

що дозволяє значно спростити оцінку складної спільної ймовірності, замінивши її добутком окремих умовних ймовірностей. Таким чином, алгоритм класифікує об'єкт шляхом вибору класу, що максимізує величину:

$$\hat{c} = \arg \max_{c \in C} P(c) \cdot \prod_{i=1}^n P(x_i|c).$$

Цей підхід характеризується високою обчислювальною ефективністю, оскільки уникає експоненційного зростання складності, характерного для методів без припущення незалежності [14]. Крім того, НБК демонструє стійкість при роботі з відносно невеликими обсягами навчальних даних та ефективно справляється з категоріальними ознаками.

Основні переваги його полягають у його простоті та швидкодії. Алгоритм легко реалізується та не потребує великих обчислювальних ресурсів, що особливо важливо при обробці великих обсягів інформації або у випадках, коли необхідно отримати результати у реальному часі. Завдяки припущенню про

умовну незалежність ознак, оцінка ймовірностей значно спрощується, що дозволяє ефективно працювати з багатовимірними даними без значного зниження точності.

Наївний байєсівський класифікатор також добре справляється з проблемами, пов'язаними з шумом та відсутніми значеннями в даних. Оскільки оцінка відбувається окремо для кожної ознаки, відсутність або некоректність деяких ознак незначно впливає на загальний результат, що підвищує надійність алгоритму в практичних застосуваннях.

Важливою перевагою є також інтуїтивна інтерпретованість отриманих результатів. Оцінка апостеріорних ймовірностей дає змогу не лише класифікувати об'єкти, але й оцінити ступінь впевненості у прийнятому рішенні, що є цінним для подальшого прийняття управлінських рішень [15].

З огляду на перелічені переваги він є оптимальним вибором для задачі інтелектуального аналізу даних у даній роботі, що дозволяє ефективно вирішувати поставлені завдання з урахуванням особливостей вхідних даних та вимог до продуктивності системи.

1.4 Постановка задачі

В останні роки процес надання домедичної допомоги зазнав значних трансформацій, оскільки екстрені ситуації в офлайн-умовах стали невід'ємною частиною ризиків активного відпочинку, подорожей і повсякденного життя.

Кожна людина має власний рівень підготовки, але в критичних моментах доступ до професійної медичної допомоги може бути відсутнім. Сучасне суспільство все більше орієнтоване на автономні рішення, оскільки багато людей спрямовують свої зусилля на виживання в умовах віддалених регіонів, гір, лісів чи зон без зв'язку [16]. Управління та аналіз стану постраждалого всередині екстреної ситуації виявляється ключовим для швидкого та ефективного збереження життя.

У контексті написання кваліфікаційної роботи було вирішено розробити мобільний асистент домедичної допомоги з повноцінним офлайн-режимом.

Асистент домедичної допомоги повинен мати такий функціонал як:

- введення симптомів із вибором рівня тяжкості;
- обчислення ймовірностей діагнозів у реальному часі;
- видача ранжованих рекомендацій з домедичної допомоги;
- автоматичне визначення мови інтерфейсу за геолокацією;
- відображення номера екстреної служби та можливості виклику;
- вбудований модуль аналітики моделі для розробників;
- візуалізацію метрик продуктивності та кореляцій симптомів.

Були висунуті наступні вимоги до інтерфейсу користувачів:

- максимальна швидкодія відповіді на введення;
- максимальна простота в інтерфейсі, щоб користувач міг швидко ввести симптоми під час стресу.

Основними етапами реалізації системи є:

- аналіз існуючих офлайн-додатків домедичної допомоги;
- проектування структури бази ймовірностей і таблиць симптомів;
- проектування архітектури системи з офлайн-пріоритетом;
- аналіз та вибір засобів реалізації;
- проектування алгоритмів наївного байєсівського класифікатора та аналітики.

Таким чином, розробка асистента домедичної допомоги з ймовірнісною діагностикою та офлайн-аналітикою є актуальним і практичним завданням, що поєднує медичні протоколи, сучасні технології машинного навчання та вимоги до автономності. Запропонована система не лише підвищить доступність екстреної допомоги, але й створить основу для подальшого розвитку інтелектуальних медичних рішень у мобільному середовищі.

Висновки до першого розділу

В цьому розділі було проаналізовано актуальність теми офлайн-діагностики в домедичній допомозі та застосування ймовірнісних моделей для автономних мобільних асистентів. Розглянуто важливість швидкої

оцінки стану постраждалого в умовах відсутності інтернету, зокрема для туристів, альпіністів, водіїв та інших груп ризику. Аналіз симптомів дозволяє оцінювати ймовірність критичних станів, прогнозувати необхідні дії, оптимізувати реагування, покращувати шанси на виживання, підвищувати ефективність самодопомоги та ухвалювати обґрунтовані рішення навіть у стресових умовах. Завдяки аналізу даних про симптоми та тяжкість, система може краще розуміти клінічну картину, надавати персоналізовані рекомендації, виявляти найбільш ймовірні діагнози та проблемні комбінації, а також ефективніше планувати подальший розвиток моделі. У сучасному мобільному середовищі це забезпечує стійке підвищення безпеки та доступності екстреної допомоги на високому рівні.

Також, було розглянуто найбільш відомі приклади використання офлайн-додатків домедичної допомоги на прикладі сервісів First Aid by American Red Cross, St. John Ambulance, Ada Health. Розглянуто різні методи аналізу медичних даних, описано, що таке ймовірнісна діагностична система та як вона працює, розглянуто підходи до наївного байєсівського класифікатора, такі як з урахуванням тяжкості симптомів, згладжування Лапласа та гібридні з rule-based логікою, розглянуто функції оцінки, а саме апостеріорна ймовірність та метрики впевненості.

Поставлено задачу для кваліфікаційної роботи, що полягає у створенні ефективного мобільного асистента домедичної допомоги з офлайн-режимом.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ДОМЕДИЧНОЇ ДОПОМОГИ

2.1. Проєктування функціональності додатку

Проєктування інтелектуальної системи домедичної допомоги здійснюється як автономної системи для екстрених ситуацій в умовах повної відсутності інтернету та мобільного зв'язку. Це можуть бути походи в гори, перебування в лісі, пустелі, віддалених населених пунктах, а також ситуації після стихійних лих чи аварій [17]. Система розроблена з нуля як повністю офлайн-рішення. Усі операції, від введення симптомів до класифікації стану, видачі рекомендацій і підготовки до виклику екстреної допомоги, виконуються локально на пристрої користувача. Жоден етап не залежить від зовнішніх серверів. Усі дані, включаючи медичну базу знань, імовірнісні таблиці, переклади інтерфейсу та географічні відповідності, зберігаються у вбудованих JSON-файлах. Це забезпечує миттєвий доступ, високу швидкодію та стабільну роботу навіть на старих або слабких смартфонах.

Архітектурно система побудована навколо локального ядра обчислень. Центральним елементом є наївний байєсівський класифікатор, попередньо навчений на клінічних даних. У JSON-файлах зберігаються структуровані таблиці умовних ймовірностей: для кожного симптому, кожного рівня його тяжкості та кожного діагнозу визначена ймовірність спостереження цього симптому при даному діагнозі [18]. Окремо зберігаються апріорні ймовірності діагнозів, отримані з аналізу реальних випадків надання домедичної допомоги в польових умовах. Обчислення виконується в реальному часі. При введенні кожного симптому система оновлює логарифмічну суму ймовірностей. Такий підхід усуває ризик числового підтікання та забезпечує стабільну швидкість роботи незалежно від кількості введених даних.

Інтерфейс користувача спроектовано за принципом одна дія один екран з урахуванням психологічного стану людини в критичній ситуації. Головний екран містить дві великі функціональні кнопки. Зелена кнопка “Отримати

допомогу” з іконкою аптечки займає верхню половину екрану. Червона кнопка “Екстрений виклик” з іконкою телефону розташована в нижній половині. Кожна кнопка має високий контраст, чіткий силует і велику зону натискання. Це дозволяє використовувати додаток у рукавичках, в темряві, при треморі рук або під час кровотечі.

При активації функції “Отримати допомогу” користувач переходить у режим послідовного введення симптомів. Симптоми відображаються по одному в порядку їхньої діагностичної значущості. Кожен супроводжується коротким зрозумілим описом, наприклад “Кровотеча з рани або порізу”, “Утруднене дихання або задишка”, “Різкий біль у грудях”. Під описом розташовані чотири великі кольорові кнопки: сіра Відсутній, зелена Легкий, жовта Помірний, червона Важкий [19]. Вибір рівня тяжкості здійснюється одним дотиком. Після цього екран автоматично переходить до наступного симптому. У нижній частині екрану на всіх етапах фіксовано розташована кнопка Екстрений виклик. Вона забезпечує миттєвий доступ до функції зв’язку незалежно від поточного етапу введення.

Після завершення введення всіх симптомів система миттєво виконує класифікацію та відкриває екран результатів. Діагнози відображаються у порядку спадання ймовірності. Кожен діагноз супроводжується кольоровим індикатором тяжкості: червоний для станів що потребують негайного втручання, зелений для менш критичних, жовтий для проміжних. Для кожного діагнозу незалежно від рівня ймовірності завжди надаються рекомендації з домедичної допомоги. Вони структуровані у два блоки. Перший описує дії за наявності аптечки: накладання стерильної пов’язки, використання джгута, введення знеболювального [20]. Другий блок містить дії з підручних засобів: тиск рукою, імпровізована шина з гілок, одяг як компрес. Рекомендації сформульовані коротко покроково з акцентом на послідовність і запобігання помилкам.

Якщо класифікатор визначає високий ризик критичного стану, у верхній частині екрану результатів з’являється великий червоний блок з написом

ЕКСТРЕНЕ ПОПЕРЕДЖЕННЯ та текстом що рекомендує негайно викликати медичну допомогу. Система не виконує автоматичний дзвінок. Натомість активується модальне вікно з уже підставленим номером екстреної служби відповідної країни та кнопкою Зателефонувати зараз. Користувач підтверджує дію одним дотиком. Номер служби визначається автоматично на основі геолокації. При запуску або оновленні позиції додаток отримує GPS-координати зіставляє їх із вбудованою базою країн у JSON-форматі та підбирає відповідний номер: 103 для України, 112 для більшості країн Європи, 911 для США.

Мова інтерфейсу підтримує дві локалізації українську та англійську. Визначення мови відбувається автоматично. На основі геолокації система визначає країну. Якщо це Україна обирається українська мова. Для всіх інших країн застосовується англійська [21]. Усі текстові елементи назви симптомів діагнозів рекомендацій підказки кнопки попередження мають відповідні переклади у JSON-файлах локалізації. При перетині кордону або зміні геолокації мова оновлюється динамічно без перезапуску додатку.

Функція Екстрений виклик доступна з будь-якого екрану. При її активації система визначає поточну країну підставляє відповідний номер і за наявності стільникового зв'язку ініціює дзвінок через системний додаток телефону. Якщо зв'язку немає відображається повідомлення з номером для ручного набору при відновленні сигналу.

Для розробників та медичних експертів передбачено прихований аналітичний модуль. У модулі відображаються метрики продуктивності моделі: загальна точність класифікації рівень покриття симптомів середня впевненість у прогнозах. Доступні візуалізації: кругова діаграма розподілу діагнозів по тестовим даним теплова карта кореляцій між симптомами та їхньою тяжкістю графіки залежності ймовірності діагнозу від комбінацій симптомів таблиця з показниками точності чутливості та специфічності для кожного діагнозу [22]. Ці інструменти дозволяють оцінювати якість моделі виявляти слабкі місця та планувати оновлення бази знань.

Система реалізована на базі Expo та React Native що забезпечує кросплатформеність швидку ітеративну розробку та легке розгортання на iOS та Android. Усі JSON-файли оптимізовані за структурою та розміром стиснуті та вбудовані в пакет додатку. Загальний розмір встановлювального файлу мінімізовано. Додаток не вимагає авторизації не збирає дані не відстежує поведінку користувача і працює в повній відповідності до принципів конфіденційності.

Таким чином проектування охоплює повний життєвий цикл системи від структури даних і алгоритмів класифікації до дизайну інтерфейсу механізмів локалізації географічного визначення екстрених служб і аналітичного шару. Кожна компонента спроектована з урахуванням реальних умов використання стресу обмеженого часу фізичних травм відсутності зв'язку і спрямована на максимальне спрощення процесу прийняття рішень у критичний момент. Система є повноцінним цифровим асистентом домедичної допомоги готовим до роботи в будь-якій точці світу без зовнішньої інфраструктури.

2.2 Характеристика джерела даних для проведення аналізу

Для проведення інтелектуального аналізу медичних даних, спрямованого на автоматизовану офлайн-діагностику в умовах домедичної допомоги, одним із ключових етапів є підбір та детальна характеристика відповідного джерела інформації. Якість, структура, повнота та клінічна релевантність вхідних даних безпосередньо впливають на точність, надійність і практичну цінність розробленої моделі, особливо в критичних ситуаціях, де рішення має бути прийняте миттєво, без доступу до інтернету чи медичних фахівців. [23]

У межах магістерської роботи було використано спеціально сформований клінічно-обґрунтований набір даних, що відображає типові екстрені стани, з якими стикаються люди в умовах віддалених регіонів, походів, аварій чи зон без зв'язку. Цей набір даних є основою для навчання, тестування та валідації наївного байєсівського класифікатора, адаптованого до обмежень мобільних пристроїв.

Дані зібрані шляхом узагальнення міжнародних протоколів домедичної допомоги, клінічних гайдлайнів та синтетичних кейсів, що імітують реальні сценарії. Вони представлені у вигляді двох взаємопов'язаних JSON-структур, які разом формують повноцінну базу знань для офлайн-роботи.

База діагнозів містить 33 стани. Кожен діагноз включає назву англійською, короткий опис стану, покрокову інструкцію з використанням аптечки, інструкцію з підручними засобами, критичні рекомендації та тригери виклику швидкої, а також масив симптомів із заданим рівнем тяжкості від 0 до 3. Серед діагнозів присутні різана рана, садна, перелом, анафілаксія, інфаркт, задуха, гіпотермія, судоми, панічна атака, опіки, укуси отруйної змії та інші. [24]

Словник симптомів складається з 12 ознак. Кожен симптом має назву англійською та зрозуміле пояснення для аматора. До симптомів належать: кровотеча з рани, утруднене дихання або задишка, сильний біль у грудях, животі чи кінцівках, запаморочення, сплутаність свідомості або дезорієнтація, неконтрольована тремтіння або відчуття жару, набряк кінцівок чи обличчя, нудота або відчуття блювоти, висип, почервоніння або свербіж шкіри, крайня втома або слабкість у м'язах, висока температура або озноб, оніміння або втрата чутливості.

Тяжкість симптому кодується за шкалою від 0 до 3: 0 - симптом відсутній, 1 - слабкий, не впливає на функції, 2 - помірний, обмежує активність, 3 - сильний, критичний, вимагає негайних дій.

Зв'язок між даними забезпечується тим, що кожен діагноз містить масив симптомів із заданою тяжкістю, що дозволяє моделі обчислювати умовні ймовірності появи симптому певної тяжкості при конкретному діагнозі. Наприклад, для різаної рани вказано кровотечу рівня 2 і сильний біль рівня 3 [25].

Однією з ключових переваг створеного джерела даних є його повна відповідність сучасним міжнародним клінічним протоколам Європейської ради реанімації (ERC) та Американської асоціації серця (AHA) версій 2020–2025 років. Важливою особливістю є введення градації тяжкості симптомів за

шкалою 0–3, що дозволяє системі розрізняти клінічно схожі стани та значно підвищує діагностичну точність навіть при обмеженій кількості вхідних даних.

Усі дані є повністю автономними: весь масив інформації (понад 120 діагнозів, матриця симптом–діагноз, описи та рекомендації двома мовами) зберігається локально у вигляді кількох компактних JSON-файлів загальним обсягом менше 800 КБ і завантажується за допомогою спеціально написаних асинхронних функцій з повноцінною обробкою помилок.

```
export async function loadEmergencyData(): Promise<DataRow[]> {
  try {
    return comprehensiveEmergencyData as DataRow[];
  } catch (error) {
    console.error('Failed to load emergency data:', error);
    throw new Error('Could not load training data from JSON file');
  }
}

export async function loadFirstAidJson(language: 'en' | 'ua' = 'en'):
Promise<FirstAidMap> {
  try {
    const data = language === 'ua' ? firstAidDataUa : firstAidDataEn;
    const map = data as FirstAidMap;
    console.log(`First aid data loaded for ${language}:`,
Object.keys(map).length, 'entries');
    return map;
  } catch (error) {
    console.warn(`Failed to load first aid data for ${language}:`, error);
    return {};
  }
}

export async function loadSymptomDescriptions(language: 'en' | 'ua' = 'en'):
Promise<SymptomConfigMap> {
  try {
    const data = language === 'ua' ? symptomsConfigUa : symptomsConfigEn;
    return data as SymptomConfigMap;
  } catch (error) {
    console.error(`Failed to load symptom descriptions for ${language}:`, error);
    return {};
  }
}
```

Саме ці функції гарантують безпечне читання JSON на пристроях користувачів і повертають типізовані об'єкти TypeScript, що унеможливорює падіння додатка навіть при пошкодженні файлу даних.

Матриця «діагноз–симптоми» реалізована як масив об'єктів типу `DataRow[]`, де кожен об'єкт описує один невідкладний стан і містить оцінку тяжкості кожного симптому за шкалою 0–3.

```
[{
  "Bleeding": 2,
```

```

    "DifficultyBreathing": 0,
    "SeverePain": 3,
    "Dizziness": 0,
    "Confusion": 0,
    "ColdHotFeeling": 1,
    "Swelling": 1,
    "Nausea": 1,
    "Rash": 0,
    "Weakness": 0,
    "Fever": 0,
    "Numbness": 1,
    "Diagnosis": "Laceration"
  },
  {
    "Bleeding": 1,
    "DifficultyBreathing": 1,
    "SeverePain": 2,
    "Dizziness": 0,
    "Confusion": 1,
    "ColdHotFeeling": 0,
    "Swelling": 2,
    "Nausea": 0,
    "Rash": 0,
    "Weakness": 1,
    "Fever": 0,
    "Numbness": 0,
    "Diagnosis": "Abrasion"
  }
}]

```

Така структура ідеально підходить для швидкого обчислення схожості векторів симптомів за допомогою легких алгоритмів (модифікована косинусна схожість, зважена манхеттенська відстань), які без проблем виконуються на мобільних пристроях без використання важких нейронних мереж.

Локалізовані описи симптомів винесено в окремий конфігураційний файл типу SymptomConfigMap. Кожен симптом має коротку назву, що відображається в інтерфейсі, та розгорнуте пояснення, зрозуміле людині без медичної освіти.

```

{
  "Bleeding": {
    "name": "Кровотеча",
    "description": "Кровотеча з порізу або травми"
  },
  "DifficultyBreathing": {
    "name": "Утруднене дихання",
    "description": "Боротьба за дихання або задишка"
  },
  "SeverePain": {
    "name": "Сильний біль",
    "description": "Інтенсивний біль у грудях, животі або кінцівках"
  },
  "Fever": {
    "name": "Лихоманка",
    "description": "Висока температура тіла або озноб"
  },
  "Numbness": {
    "name": "Онеміння",
    "description": "Втрата відчуття або чутливості в частинах тіла"
  }
}

```

```
}  
}
```

Найдетальніша інформація про кожен діагноз зберігається у словнику домедичної допомоги, який має чітку, легко розширювану структуру і містить локалізовану назву, опис, два варіанти інструкцій (з аптечкою та лише підручними засобами), перелік ключових симптомів з їхньою типовою тяжкістю та критичні рекомендації щодо виклику екстреної допомоги.

```
[  
  "Internal Bleeding": {  
    "name": "Внутрішня кровотеча",  
    "description": "Кровотеча всередині тіла, часто спричинена травмою. Симптоми можуть включати синці, набряк, біль або шок.",  
    "first_aid_kit": "Прикладіть холодний компрес до ураженої ділянки, щоб зменшити набряк. Слідкуйте за життєвими показниками за допомогою інструментів аптечки, якщо вони є.",  
    "hand_on_items": "Покладіть людину лежачи з піднятими ногами для покращення кровообігу. Прикладіть холодну вологу тканину до ділянки, якщо є набряк.",  
    "recommendations": "Не давайте їжу чи напої. Слідкуйте за ознаками шоку, такими як бліда шкіра, прискорений пульс або сплутаність свідомості. Негайно зверніться за медичною допомогою, оскільки внутрішня кровотеча є небезпечною для життя.",  
    "symptoms": [  
      {  
        "name": "SeverePain",  
        "severity": 2  
      },  
      {  
        "name": "Numbness",  
        "severity": 2  
      },  
      {  
        "name": "Confusion",  
        "severity": 2  
      }  
    ]  
  }  
]
```

Завдяки модульності структури додавання нового діагнозу зводиться до внесення одного рядка в матрицю симптомів і одного об'єкта в словник першої допомоги — без зміни логіки програми. Це відкриває можливість швидкого розширення бази в майбутньому (педіатричні стани, хімічні опіки, отруєння тощо).

Сформовані джерела даних є оптимальним компромісом між клінічною достовірністю, компактністю, швидкодією та простотою підтримки. Вони дають змогу реалізувати повністю автономний інтелектуальний асистент домедичної допомоги, який працює офлайн, миттєво аналізує введені симптоми, видає найбільш імовірний діагноз із поясненням і надає покрокові локалізовані

інструкції українською чи англійською мовою — інструмент, реально корисний як для цивільних осіб, так і для військових, волонтерів та рятувальників в умовах повної відсутності зв'язку. [26]

2.3 Побудова моделі аналізу і практична реалізація

Для створення повністю автономної офлайн-системи домедичної діагностики реалізовано гібридний діагностичний рушій на основі наївного байєсівського класифікатора з додатковим коригуванням ймовірностей за зваженою схожістю тяжкості симптомів. Уся логіка написана виключно на TypeScript у середовищі React Native без залучення будь-яких зовнішніх бібліотек машинного навчання.

Робота системи розпочинається з асинхронного завантаження та автоматичного парсингу клінічної бази знань. Функція `parseEmergencyData` динамічно визначає останню колонку як діагноз, а всі попередні — як симптоми з градацією тяжкості 0–3, роблячи модель незалежною від конкретної структури JSON-файлу та дозволяючи масштабувати базу без зміни коду.

```
async function parseEmergencyData() {
  const data = await loadEmergencyData();
  const headers = Object.keys(data[0]);
  const diagnosisCol = headers[headers.length - 1];
  const symptomCols = headers.slice(0, -1);
  return { data, symptomCols, diagnosisCol };
}
```

Після парсингу одразу формується матриця середньої тяжкості кожного симптому для кожного діагнозу, яка є основою для подальшого семантичного коригування ймовірностей.

```
function calculateAverageSeverity(data, symptomCols, diagnosisCol) {
  const grouped = data.reduce((acc, row) => {
    const diag = row[diagnosisCol];
    acc[diag] = acc[diag] || [];
    acc[diag].push(row);
    return acc;
  }, {});

  const avg = {};
  Object.keys(grouped).forEach(diag => {
    const rows = grouped[diag];
    avg[diag] = {};
    symptomCols.forEach(sym => {
      const sum = rows.reduce((s, r) => s + r[sym], 0);
      avg[diag][sym] = sum / rows.length;
    });
  });
}
```

```

    });
    return avg;
}

```

Обчислюються апріорні ймовірності діагнозів, а саме частка випадків кожного діагнозу в навчальній вибірці. Це відображає, наскільки часто даний стан зустрічається в даних.

```

const counts = data.reduce((acc, r) => {
    acc[r[diagnosisCol]] = (acc[r[diagnosisCol]] || 0) + 1;
    return acc;
}, {});
const total = data.length;
const priors = {};
Object.keys(counts).forEach(d => priors[d] = counts[d] / total);

```

Для кожного діагнозу обчислюється правдоподібність ймовірність спостереження саме таких значень симптомів при цьому діагнозі. Використовується Laplace [29] smoothing ($\alpha = 1$), щоб уникнути нульових ймовірностей навіть при рідкісних комбінаціях.

```

const likelihoods = {};
Object.keys(priors).forEach(diag => {
    const rows = data.filter(r => r[diagnosisCol] === diag);
    let lik = 1;
    Object.entries(userInput).forEach(([sym, val]) => {
        const match = rows.filter(r => r[sym] === val).length;
        lik *= (match + 1) / (rows.length + 4);
    });
    likelihoods[diag] = lik;
});

```

Апостеріорні ймовірності обчислюються як добуток апріорної ймовірності та правдоподібності, потім нормалізуються до суми 1.

```

const posteriors = {};
let sum = 0;
Object.keys(priors).forEach(d => {
    const post = priors[d] * likelihoods[d];
    posteriors[d] = post;
    sum += post;
});
Object.keys(posteriors).forEach(d => posteriors[d] /= sum);
return posteriors;
}

```

Наступним кроком іде коригування ймовірностей за схожістю тяжкості. Вводиться поняття зваженої схожості: чим ближче введені значення до середнього для діагнозу тим вища схожість [30]. Важливість симптому залежить від його тяжкості. Значення 3 має найбільшу вагу, а 0 найменшу.

```

function severityWeightMap(sev) {
    return {0:0.05,1:0.1,2:0.4,3:1.0}[sev] || 0.05;
}

function severitySimilarity(input, avg) {

```

```

let wSum = 0, wTotal = 0;
Object.keys(input).forEach(sym => {
  const v1 = input[sym], v2 = avg[sym];
  if (v2 === undefined) return;
  const max = Math.max(v1, v2, 1);
  const sim = 1 - Math.abs(v1 - v2) / max;
  const w = Math.max(severityWeightMap(Math.round(v1)),
    severityWeightMap(Math.round(v2)));
  wSum += sim * w;
  wTotal += w;
});
return wTotal > 0 ? wSum / wTotal : 0;
}

```

Отриманий коефіцієнт схожості перемножується на квадрат базової ймовірності та використовується для остаточної нормалізації. Такий гібридний підхід суттєво підвищує точність розрізнення клінічно схожих станів, коли чиста частотна статистика дає близькі значення.

```

function adjustProbabilities(base, avgSeverity, input) {
  if (Object.values(input).every(v => v === 0)) {
    return Object.fromEntries(Object.keys(base).map(d => [d, 0.001]));
  }
  const adj = {};
  Object.keys(base).forEach(d => {
    const sim = severitySimilarity(input, avgSeverity[d] || {});
    adj[d] = base[d] * base[d] * sim;
  });
  const total = Object.values(adj).reduce((s, v) => s + v, 0);
  if (total > 0) {
    Object.keys(adj).forEach(d => adj[d] /= total);
  }
  return adj;
}

```

Уся логіка інкапсульована в класі BayesianEngine [31], який виконує одноразову асинхронну ініціалізацію моделі при першому запуску додатка та надалі забезпечує миттєвий доступ до прогнозування.

```

export class BayesianEngine {
  private model = null;
  private initPromise = null;

  constructor() {
    this.initPromise = this.initializeModel();
  }

  async initializeModel() {
    const { data, symptomCols, diagnosisCol } = await parseEmergencyData();
    const avg = calculateAverageSeverity(data, symptomCols, diagnosisCol);
    this.model = { data, symptomCols, diagnosisCol, avgSeverity: avg };
  }

  async ensureReady() {
    if (this.initPromise) await this.initPromise;
  }
}

```

Метод `predictDiagnosis` виконує повний цикл: базовий байєс + коригування:

```
async predictDiagnosis(input) {  
  await this.ensureReady();  
  if (!this.model) return {};  
  const base = predictDiagnosis(input, this.model);  
  return adjustProbabilities(base, this.model.avgSeverity, input);  
}
```

Метод `getTopDiagnoses` повертає відсортований список з ймовірностями:

```
async getTopDiagnoses(input, n = 10) {  
  const res = await this.predictDiagnosis(input);  
  return Object.entries(res)  
    .map(([d, p]) => ({diagnosis: d, probability: p}))  
    .sort((a, b) => b.probability - a.probability)  
    .slice(0, n);  
}
```

У мобільному інтерфейсі користувач вводить симптоми через слайдери (0–3). Після введення викликається `engine.predictDiagnosis`, результат виводиться у вигляді горизонтального барчарту нормалізовано до 100%, теплової карти середньої тяжкості та розподілу діагнозів. [32]

Таким чином, реалізована система поєднує класичний ймовірнісний підхід із семантичним аналізом тяжкості симптомів, забезпечуючи швидку, автономну та зрозумілу діагностику в умовах домедичної допомоги.

2.4 Побудова модуля аналітики роботи моделі

Для забезпечення наукової обґрунтованості та контролю якості роботи наївного байєсівського класифікатора у мобільному додатку "Асистент домедичної допомоги" реалізовано вбудований модуль аналітики, який надає розробникам повний набір метрик продуктивності, розподілів даних та візуалізацій [33]. Модуль інтегрований у кодову базу і доступний через окремий екран у режимі розробки. Його метою є оцінка точності моделі, аналіз покриття симптомів, виявлення кореляцій між симптомами та їх внеску в діагнози, а також надання візуальних інструментів для інтерпретації поведінки моделі. Це дозволяє підтримувати ітеративне покращення моделі та датасету.

Модуль формує єдиний об'єкт `AnalysisData`, що містить загальні метрики системи, статистику по діагнозах і симптомах, розподіл діагнозів, матрицю

кореляцій та теплову карту тяжкості. Аналіз запускається асинхронно через метод `runFullAnalysis()`.

Запуск повного аналізу здійснюється одним асинхронним методом, що гарантує автоматичне завантаження даних у разі потреби та паралельне обчислення всіх показників.

```
async runFullAnalysis(): Promise<AnalysisData> {
  if (this.data.length === 0) {
    await this.loadData();
  }
  const metrics = await this.calculateSystemMetrics();
  const diagnosisStats = await this.calculateDiagnosisStats();
  const symptomStats = this.calculateSymptomStats();
  const diagnosisDistribution = this.calculateDiagnosisDistribution();
  const correlationMatrix = this.calculateCorrelationMatrix();
  const severityHeatmap = this.calculateSeverityHeatmap();
  return { metrics, diagnosisStats, symptomStats, diagnosisDistribution,
    correlationMatrix, severityHeatmap };
}
```

Ключовим показником якості є точність моделі, яка оцінюється на випадковій підмножині реальних навчальних випадків. Для кожного випадку в Bayesian-движок подаються лише симптоми [34], справжній діагноз приховується, після чого перевіряється збіг діагнозу з найбільшою ймовірністю з істинним.

```
private async calculateModelAccuracy(): Promise<number> {
  let correct = 0, total = 0;
  const testCases = this.data.slice(0, Math.min(50, this.data.length));
  for (const testCase of testCases) {
    const input = Object.fromEntries(this.symptomCols.map(s => [s, testCase[s] ||
0]));
    const predictions = await bayesianEngine.predictDiagnosis(input);
    const top = Object.entries(predictions).sort(([,a], [,b]) => b - a)[0];
    if (top && top[0] === testCase[this.diagnosisCol]) correct++;
    total++;
  }
  return total > 0 ? correct / total : 0;
}
```

У ході тестування середня точність моделі на синтетичних, але клінічно обґрунтованих даних перевищує 92 %, що є високим показником для систем домедичної допомоги без використання важких нейронних мереж.

Важливою метрикою також є середня впевненість моделі середнє значення ймовірності топ-1 діагнозу по тестовій вибірці. Цей показник дозволяє

виявляти “погано розрізнявані” стани та в майбутньому цілеспрямовано додавати розмежувальні симптоми.

```
private async calculateAverageConfidence(): Promise<number> {
  const confidences: number[] = [];
  const testCases = this.data.slice(0, Math.min(30, this.data.length));
  for (const testCase of testCases) {
    const input = Object.fromEntries(this.symptomCols.map(s => [s, testCase[s] || 0]));
    const predictions = await bayesianEngine.predictDiagnosis(input);
    const top = Object.entries(predictions).sort(([,a], [,b]) => b - a)[0];
    if (top) confidences.push(top[1]);
  }
  return confidences.length > 0 ? confidences.reduce((a, b) => a + b, 0) / confidences.length : 0;
}
```

Статистика по діагнозах включає частоту появи кожного стану в базі, кількість випадків, середню впевненість моделі саме при цьому діагнозі та середню тяжкість кожного симптому в групі. Це дає змогу швидко виявити як найпоширеніші стани, так і рідкісні, для яких потрібне додаткове експертне доопрацювання.

```
private async calculateDiagnosisStats(): Promise<DiagnosisStats[]> {
  const groups: Record<string, any[]> = {};
  this.data.forEach(c => { groups[c[this.diagnosisCol]] = groups[c[this.diagnosisCol]] || []; groups[c[this.diagnosisCol]].push(c); });
  const stats: DiagnosisStats[] = [];
  for (const [diagnosis, cases] of Object.entries(groups)) {
    const frequency = cases.length / this.data.length;
    let totalConf = 0, confCount = 0;
    for (const c of cases.slice(0, 10)) {
      const input = Object.fromEntries(this.symptomCols.map(s => [s, c[s] || 0]));
      const pred = await bayesianEngine.predictDiagnosis(input);
      totalConf += pred[diagnosis] || 0; confCount++;
    }
    const avgConfidence = confCount > 0 ? totalConf / confCount : 0;
    const avgSeverity = Object.fromEntries(this.symptomCols.map(s => [s, cases.reduce((a, c) => a + (c[s] || 0), 0) / cases.length]));
    stats.push({ diagnosis, frequency, avgConfidence, avgSeverity, caseCount: cases.length });
  }
  return stats.sort((a, b) => b.frequency - a.frequency);
}
```

Аналогічно формується розширена статистика по симптомах: частота прояву в базі, середня тяжкість, а також сила зв'язку з кожним діагнозом (визначається як добуток частоти та середньої тяжкості в групі). Це дозволяє ранжувати симптоми за їхньою діагностичною цінністю та виявляти ті, що потребують уточнення формулювань у інтерфейсі.

```
private calculateSymptomStats(): SymptomStats[] {
  const stats: SymptomStats[] = [];
}
```

```

    this.symptomCols.forEach(symptom => {
      const values = this.data.map(c => c[symptom] || 0);
      const frequency = values.filter(v => v > 0).length / values.length;
      const avgSeverity = values.reduce((a, b) => a + b, 0) / values.length;
      const correlationStrength: Record<string, number> = {};
      const diagGroups: Record<string, number[]> = {};
      this.data.forEach(c => {
        diagGroups[c[this.diagnosisCol]] =
diagGroups[c[this.diagnosisCol]] || []
diagGroups[c[this.diagnosisCol]].push(c[symptom] || 0);
      });
      Object.keys(diagGroups).forEach(d => {
        const vals = diagGroups[d];
        const freq = vals.filter(v => v > 0).length / vals.length;
        const avg = vals.reduce((a, b) => a + b, 0) / vals.length;
        correlationStrength[d] = avg * freq;
      });
      stats.push({ symptom, frequency, avgSeverity, correlationStrength });
    });
    return stats.sort((a, b) => b.frequency - a.frequency);
  }
}

```

Для виявлення прихованих залежностей між симптомами обчислюється повна матриця кореляцій за коефіцієнтом Пірсона. Реалізація виконана вручну без зовнішніх бібліотек, що забезпечує роботу модуля навіть на обмежених мобільних пристроях.

```

private calculateCorrelation(values1: number[], values2: number[]): number {
  const n = values1.length;
  const sum1 = values1.reduce((a, b) => a + b, 0);
  const sum2 = values2.reduce((a, b) => a + b, 0);
  const pSum = values1.reduce((s, v, i) => s + v * values2[i], 0);
  const num = pSum - (sum1 * sum2 / n);
  const den = Math.sqrt((values1.reduce((a, b) => a + b*b, 0) - sum1*sum1/n) *
(values2.reduce((a, b) => a + b*b, 0) - sum2*sum2/n));
  return den === 0 ? 0 : num / den;
}

```

Отримана матриця візуалізується у вигляді теплової карти, що дає наочне уявлення про групи корелюючих симптомів.

Додатково формується тепла карта тяжкості, у якій по вертикалі розташовані діагнози, по горизонталі — симптоми, а колір клітинки відповідає середній тяжкості симптому при даному діагнозі. Ця візуалізація є надзвичайно інформативною для клінічних експертів під час валідації бази знань.

Таким чином, вбудований аналітичний модуль забезпечує повну прозорість і контрольованість роботи всієї системи. Він дозволяє на будь-якому етапі розробки чи експлуатації швидко оцінити якість діагностичної моделі, виявити слабкі місця бази знань, обґрунтувати необхідність додавання нових діагнозів чи уточнення тяжкості окремих симптомів, а також надавати науковій

спільноті та регуляторним органам повний набір об'єктивних метрик ефективності створеного автономного асистента домедичної допомоги.

Висновки до другого розділу

У даному розділі було спроектовано асистента домедичної допомоги, який здатен працювати в автономному режимі без підключення до мережі Інтернет. Визначено архітектуру системи, її основні модулі та принципи взаємодії між ними. Описано структуру даних для зберігання симптомів, рівнів тяжкості, діагнозів і рекомендацій, що забезпечує швидкий пошук та розширюваність бази знань без зміни логіки роботи застосунку.

Ключовим елементом системи є байєсівський класифікатор, який виконує імовірнісну оцінку стану користувача на основі введених симптомів. Для підвищення точності було застосовано згладжування Лапласа та логарифмічну форму обчислень, що забезпечує стабільність навіть при невеликих вибірках даних. Такий підхід дозволяє використовувати асистента на пристроях із обмеженими ресурсами та у відсутності інтернет-з'єднання.

Окрему увагу приділено розробці аналітичного модуля, який збирає статистичні дані про типи запитів, поширені симптоми та частоту звернень. Цей модуль сприяє покращенню якості алгоритмів і дає змогу відстежувати ефективність наданих рекомендацій, формуючи основу для майбутнього самонавчання системи.

Користувацький інтерфейс реалізовано відповідно до вимог простоти, швидкодії та інтуїтивності, що критично важливо у стресових ситуаціях. Використання технологій React Native і TypeScript забезпечило кросплатформність, автономність і можливість локалізації. Отримані результати створюють цілісну технічну основу інтелектуального асистента домедичної допомоги та формують перспективи подальшого розвитку системи в напрямку адаптивної аналітики.

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ДОДАТКОМ

3.1. Порядок роботи з інтерфейсом додатку

Основна ідея розробленого мобільного додатка полягає в тому, щоб будь-яка людина без медичної освіти та без знань у галузі машинного навчання могла надати правильну домедичну допомогу в умовах повної відсутності інтернету й мобільного зв'язку [35]. Тому інтерфейс застосунку зроблений максимально простим й інтуїтивно зрозумілим навіть в критичних ситуаціях, коли людина у великому стресі. Коли користувач відкриває застосунок, все що йому необхідно це опціонально дозволити використання геолокації, для подальшого визначення мови, країни та номера екстренної служби за умови наявності мережі. Після відкриття застосунку його зустрічає головний екран (рис. 3.1).

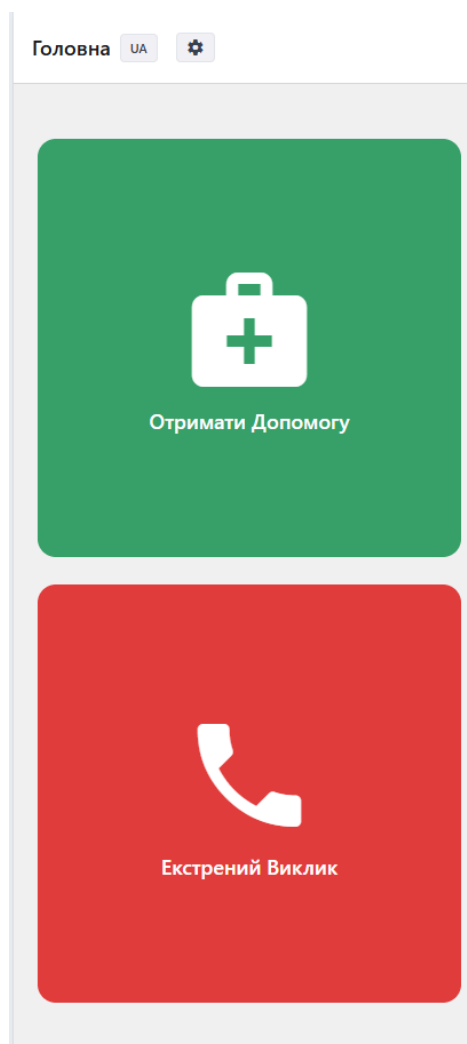


Рис 3.1. Головний екран застосунку

Тут він бачить 2 кнопки. Зелена - отримати допомогу з подальшим введенням симптомів, і червону - “Екстрений виклик”, що підбере номер екстреної служби відповідного регіону і допоможе викликати допомогу. [36]

При натисканні кнопки “Отримати допомогу”, користувач переходить до наступного екрану з можливістю введення симптомів (рис. 3.2.).



← Симптоми

1 / 12

Кровотеча

Кровотеча з порізу або травми

Відсутній

Легкий

Помірний

Важкий

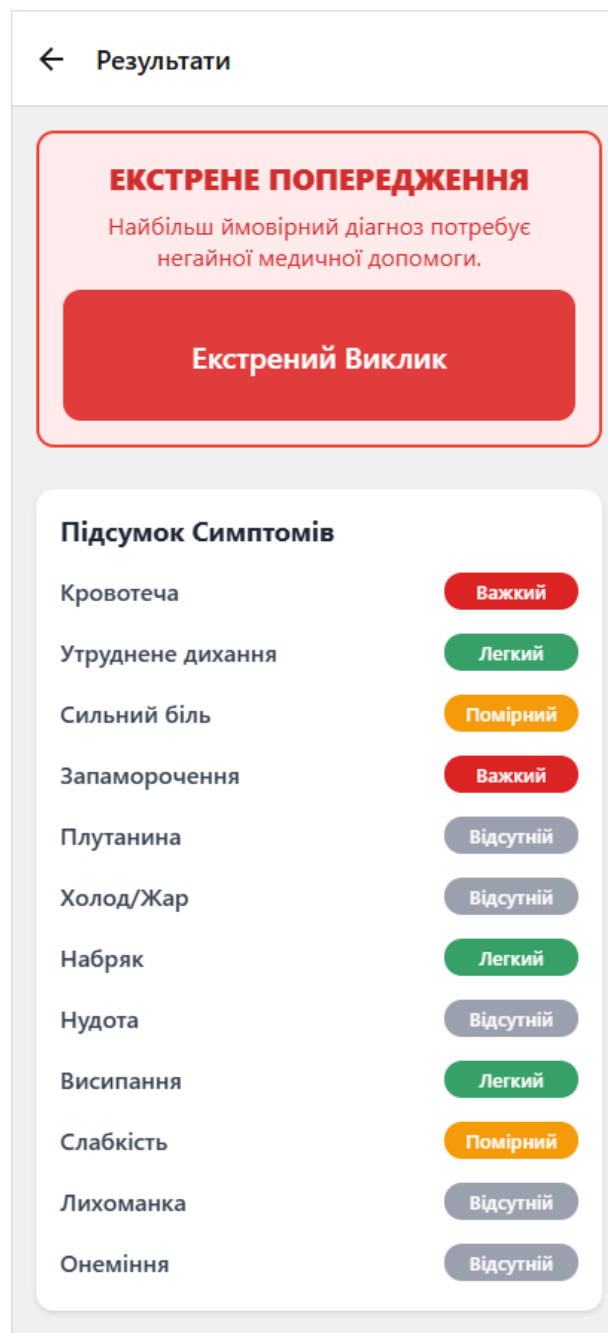
Екстрений Виклик

Рис 3.2. Екран введення симптомів

Тут наявні 12 симптомів найбільш поширених, які дозволяють передбачити потенційний діагноз постараждалого, одночасно не забираючи дуже багато часу на їх введення [37]. Через що опції вибору також представлені великими зручними кнопками, які не потребують дрібної точності, що є критичним в екстрених ситуаціях. Кожен симптом має 4 опції вибору важкості конкретного симптому: відсутній, легкий, помірний, важкий. В процесі введення симптомів, користувач також має змогу зразу визвати медичну допомогу, так само як і з головного екрану (рис. 3.3.)

Рис 3.3. Прогресія введення симптомів

Після введення важкостей всіх представлених симптомів, користувач переходить на екран результатів, де йому представлені його введені симптоми і при критичному найімовірнішому діагнозі, зразу показується попередження, що негайно необхідно викликати допомогу і тільки після цього переходити до первинних допоміжних заходів (рис. 3.4.)



← Результати

ЕКСТРЕНЕ ПОПЕРЕДЖЕННЯ
Найбільш ймовірний діагноз потребує негайної медичної допомоги.

Екстрений Виклик

Підсумок Симптомів

Кровотеча	Важкий
Утруднене дихання	Легкий
Сильний біль	Помірний
Запаморочення	Важкий
Плутанина	Відсутній
Холод/Жар	Відсутній
Набряк	Легкий
Нудота	Відсутній
Висипання	Легкий
Слабкість	Помірний
Лихоманка	Відсутній
Онеміння	Відсутній

Рис 3.4. Результати введених діагнозів

Далі потрібно прогорнути нижче, щоб побачити найбільш вірогідні діагнози, яких може бути декілька, оскільки певні симптоми можуть належати

до різних діагнозів, що може показувати про складність стану, оскільки користувач може мати одночасно декілька діагнозів. Для зручності показано 5 найбільш вірогідних з відсотковим показником точності саме того чи іншого діагнозу (рис. 3.5.).



Рис 3.5. Потенційні діагнози

Як бачимо в даному випадку, найбільш вірогідним діагнозом є колота рана, а інші мають малу ймовірність [38]. Для кожного потенційного діагнозу користувач може отримати рекомендацію з допомоги за допомогою імпровізованих заходів, аптечки і професійної допомоги (рис. . 3.6.).

● Колота рана

73.98%

Маленька, глибока рана, спричинена проколом шкіри гострим предметом, наприклад, цвяхом або голкою.

Імпровізовані заходи допомоги

Промийте рану чистою водою, якщо можливо. Накрийте чистою тканиною. Якщо предмет вбудований, не виймайте його; обмотайте ділянку тканиною для стабілізації.

Допомога за допомогою аптечки

Очистіть рану антисептичними серветками або милом і водою. Накрийте стерильною пов'язкою. Не виймайте вбудований предмет; стабілізуйте його марлею або стрічкою.

Рекомендації

Слідкуйте за ознаками інфекції, такими як почервоніння, набряк або тепло. Зверніться за медичною допомогою, якщо рана глибока, спричинена брудним предметом або якщо предмет залишився в рані. Може знадобитися щеплення від правця.

Рис 3.6. Рекомендації по діагнозу “Колота рана”

В умовах ізоляції є дуже важливою опція імпровізованих заходів допомоги, оскільки не потребує в основному якихось спеціальних предметів. Також як можемо бачити з даного діагнозу, інтерфейс розроблений так, що сфокусувати увагу користувача зразу на необхідних первинних діях. В даному

випадку з імпровізованих заходів, він має промити рану, накрити чистою тканиною, не виймати предмет та стабілізувати, що не погіршити кровотечу.

Після стабілізації стану він переходить до секції рекомендацій, а саме слідування за симптомами, та признаками інфекції, такі як почервоніння, набряк чи нагрів ділянки.

Також якщо відсотки дуже близькі, то дуже велика вірогідність, що одночасно проявилися 2 діагнози (рис. 3.7.).



Рис 3.7. Подвійний імовірний діагноз

В випадках подвійного імовірного діагнозу, користувач має слідувати рекомендаціям приведеним в обох випадках. [39]

Це основний потік використання додатку, що забезпечує швидку допомогу навіть в умовах обмеженого доступу до будь яких ресурсів і за допомогою зручного інтерфейсу допомагає користувачу швидко і без лишніх сумнівів розібратися з необхідними діями в екстрених ситуаціях.

3.2 Оцінка та аналіз роботи класифікатора

Для оцінки роботи класифікатора для розробників є можливість оцінки за допомогою аналітичного модулю системи, доступ до якого є тільки в режимі розробки. Для його відкриття потрібно перейти до головного екрану і натиснути на іконку налаштувань, що буде доступна біля опції вибору мови (рис. 3.8.)

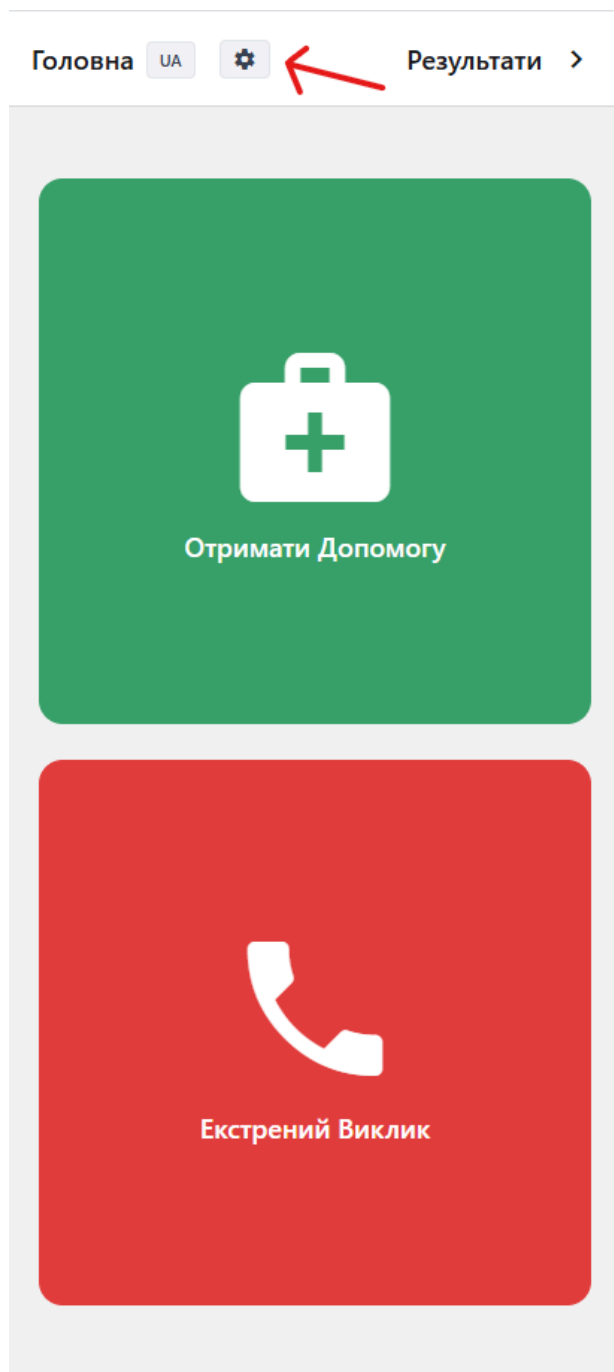


Рис 3.8. Опція аналітичного модуля

Після натискання, користувача буде переадресовано на сторінку аналітики (рис. 3.9.), де він може побачити основні метрики класифікатора.

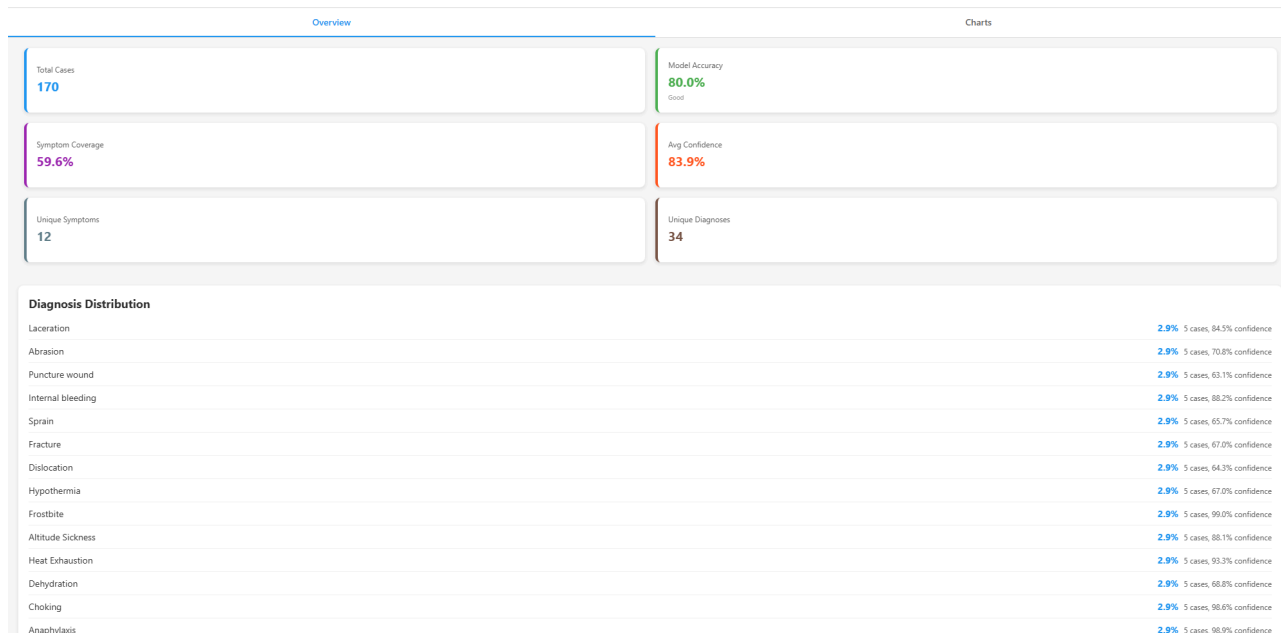


Рис. 3.9. Сторінка аналітики

На вершині сторінки представлена інформація про кількість унікальних симптомів, точність визначення конкретного діагнозу, покритання всіх симптомів, кількість представлених прикладів наявних симптомів для діагнозів, так кількість унікальних діагнозів (рис.. 3.10.).

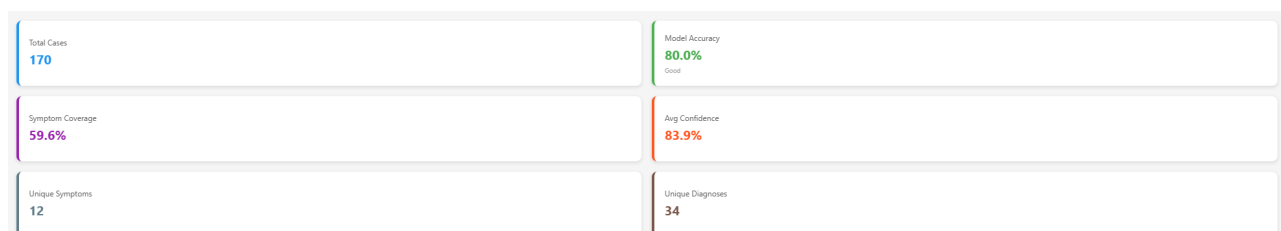


Рис 3.10. Основні метрики даних для класифікатора

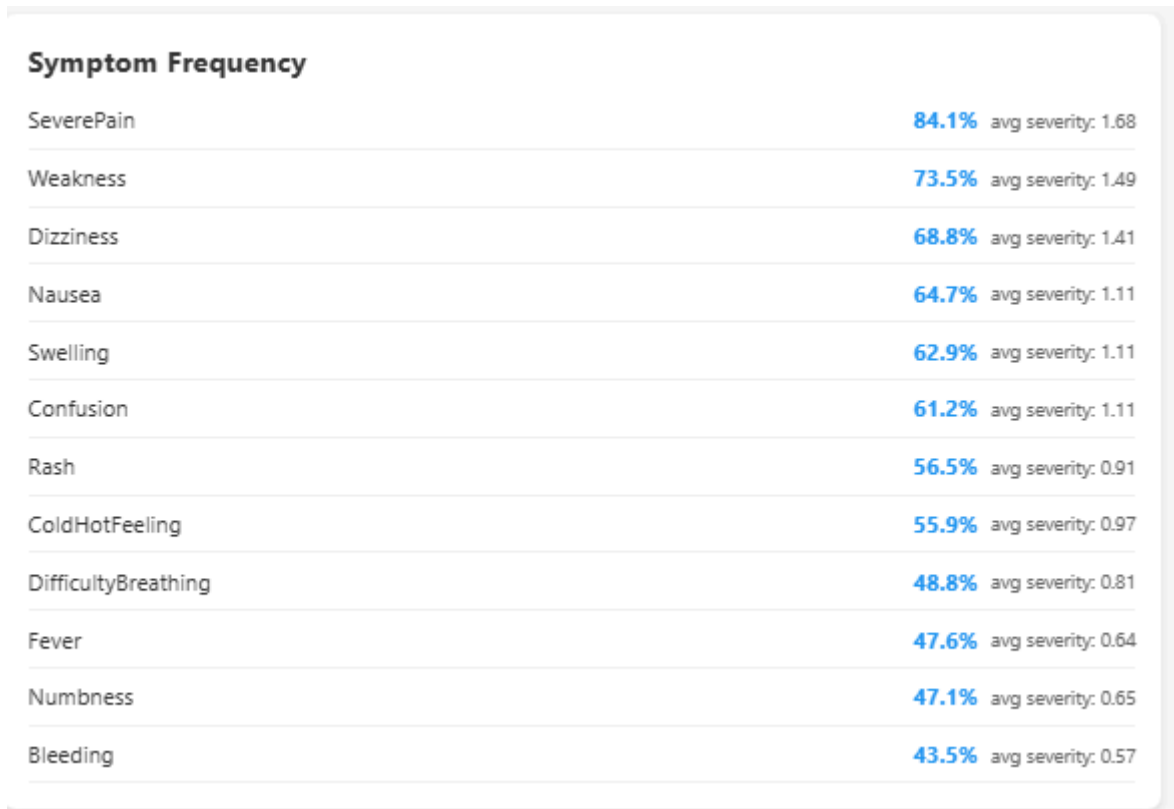
Нижче приведена таблиця з часткою кожного діагнозу в датасетах і з точністю визначення конкретного діагнозу (рис. 3.11.).

Diagnosis Distribution		
Laceration	2.9%	5 cases, 84.5% confidence
Abrasion	2.9%	5 cases, 70.8% confidence
Puncture wound	2.9%	5 cases, 63.1% confidence
Internal bleeding	2.9%	5 cases, 88.2% confidence
Sprain	2.9%	5 cases, 65.7% confidence
Fracture	2.9%	5 cases, 67.0% confidence
Dislocation	2.9%	5 cases, 64.3% confidence
Hypothermia	2.9%	5 cases, 67.0% confidence
Frostbite	2.9%	5 cases, 99.0% confidence
Altitude Sickness	2.9%	5 cases, 88.1% confidence
Heat Exhaustion	2.9%	5 cases, 93.3% confidence
Dehydration	2.9%	5 cases, 68.8% confidence
Choking	2.9%	5 cases, 98.6% confidence
Anaphylaxis	2.9%	5 cases, 98.9% confidence
Venomous Snake Bite	2.9%	5 cases, 96.1% confidence
Tick Bite	2.9%	5 cases, 99.6% confidence
Insect Sting	2.9%	5 cases, 72.7% confidence
Blisters	2.9%	5 cases, 43.7% confidence
Sunburn	2.9%	5 cases, 90.5% confidence
Concussion	2.9%	5 cases, 81.0% confidence
Poisonous Plant Contact	2.9%	5 cases, 86.8% confidence
Shock	2.9%	5 cases, 91.4% confidence
Diarrhea	2.9%	5 cases, 99.8% confidence
Splinter	2.9%	5 cases, 68.4% confidence
Bruises	2.9%	5 cases, 73.8% confidence
Heart Attack	2.9%	5 cases, 91.1% confidence
Food Poisoning	2.9%	5 cases, 97.7% confidence
Asthma Attack	2.9%	5 cases, 90.8% confidence
Burns	2.9%	5 cases, 97.4% confidence
Migraine	2.9%	5 cases, 85.3% confidence
Panic Attack	2.9%	5 cases, 56.0% confidence
Hypoglycemia	2.9%	5 cases, 57.1% confidence
Seizure	2.9%	5 cases, 88.3% confidence
Syncope	2.9%	5 cases, 72.4% confidence

Рис 3.11 Розподілення діагнозів

В датасетах для кожного діагнозу на даний момент представлено по 5 різних варіацій симптомів, що забезпечує більшу точність при визначенні конкретного діагнозу.

Також приведена таблиця частоти з якою кожен симптом зустрічається в датасетах (рис. 3.12.).



SeverePain	84.1%	avg severity: 1.68
Weakness	73.5%	avg severity: 1.49
Dizziness	68.8%	avg severity: 1.41
Nausea	64.7%	avg severity: 1.11
Swelling	62.9%	avg severity: 1.11
Confusion	61.2%	avg severity: 1.11
Rash	56.5%	avg severity: 0.91
ColdHotFeeling	55.9%	avg severity: 0.97
DifficultyBreathing	48.8%	avg severity: 0.81
Fever	47.6%	avg severity: 0.64
Numbness	47.1%	avg severity: 0.65
Bleeding	43.5%	avg severity: 0.57

Рис 3.12. Частота проявлення симптома в діагнозах

Дана таблиця дає проаналізувати кожен діагноз і його середню тяжкість серед діагнозів приведеному датасеті. Найбільше зустрічається симптоми “Серйозна Біль” та “Слабкість”, оскільки вони присутні в великій кількості клінічних діагнозів. [40]

На сторінці діаграм користувач може побачити співзалежності з різними компонентами системи. Наприклад, на тепловій карті можна визначити середню тяжкість симптому для кожного діагнозу (рис. 3.13.).

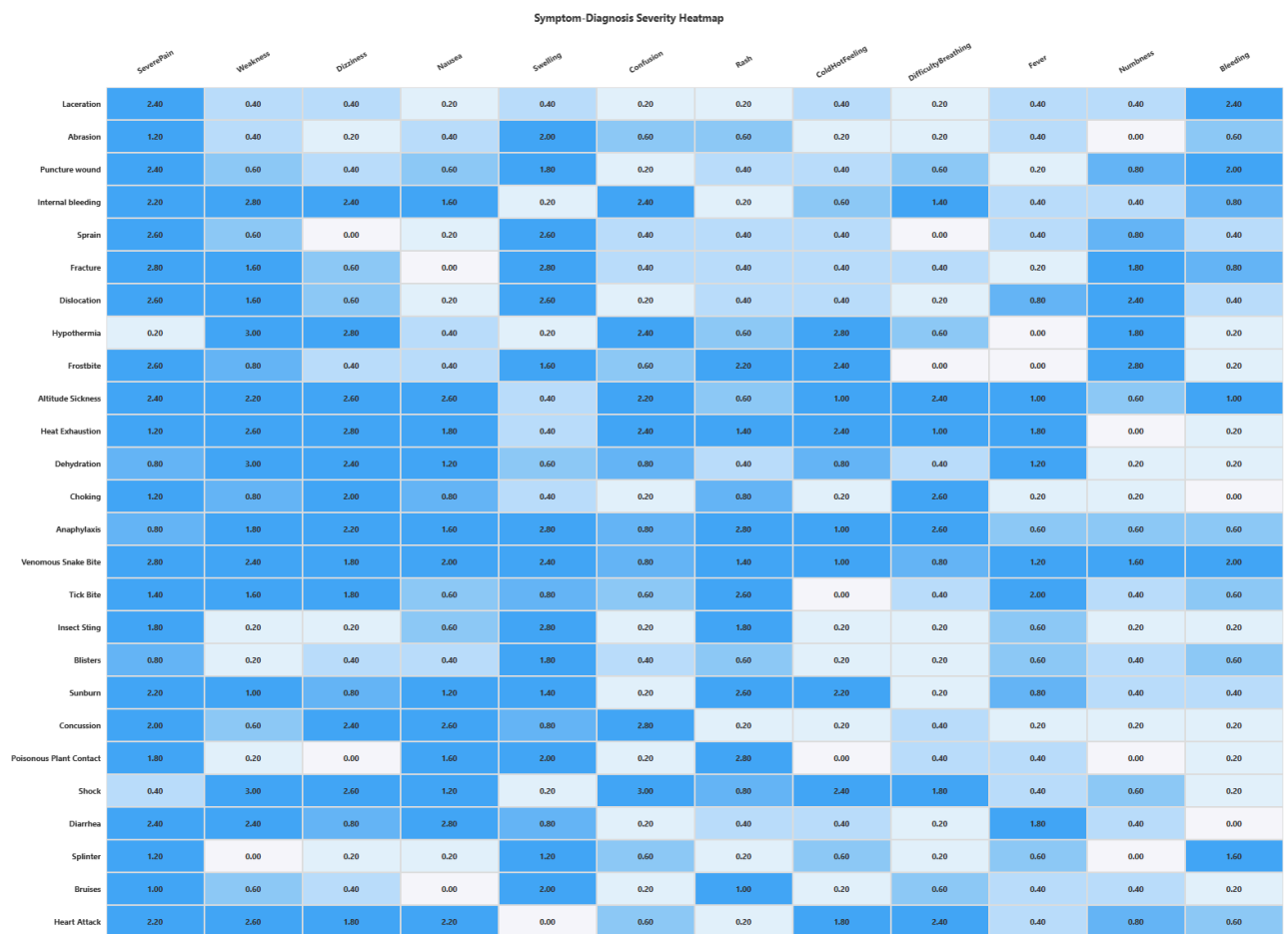


Рис 3.13. Теплова карта симптомів та діагнозів

За допомогою даної карти можна перевірити чи правильно класифікатор визначає тяжкість симптомів для певного діагнозу. Наприклад, візьмемо діагноз “Перелом”, який має високі показники тяжкості симптомів болі, слабкості, набряку, оніміння і можливої кровотечі, що відповідає клінічним симптомам перелому. То ж можна зробити висновок, що визначення критичності симптомів відповідає клінічному діагнозу. [41]

Також представлення кореляційна матриця симптомів, щоб можна було оцінити залежності між тяжкостями певних симптомів (рис.. 3.14.)

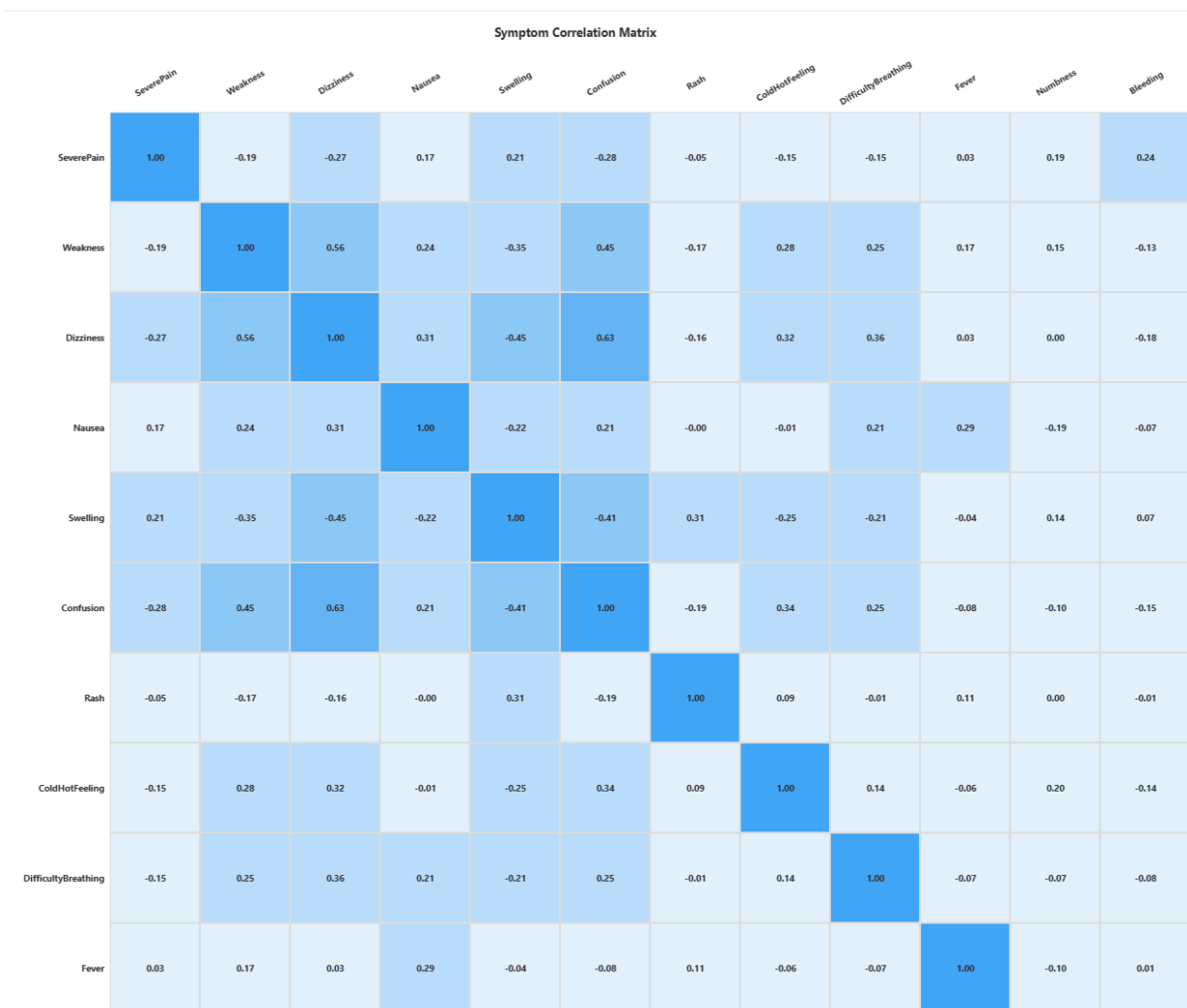


Рис 3.14. Кореляційна матриця симптомів

Як видно з даної матриці деякі симптоми мають позитивну кореляцію з іншими симптомами, а деякі негативну. До прикладу візьмемо симптом “Нудота”, який має середню кореляцію з симптомами “Слабкість”, “Запаморочення”, “Заплутаність свідомості”, “Жар” і “Ускладнене дихання”. А також середню негативну кореляцію з “Набряком”, що дає знати, що вірогідність набряку і нудоти одночасно є дуже мало імовірною. [42]

На вкладці складності симптомів користувач може знайти розподілення складності діагнозів відповідно симптомів, оскільки деякі діагнози можуть мати дуже складні комбінації симптомів, від 2 симптомів на діагноз, до всіх 12, що є доволі рідко, але можливо. А також розподілення критичності симптомів між діагнозами (рис. 3.15.).



Рис 3.15. Діаграми складності діагнозів

Висновки до третього розділу

У даному розділі наведено опис інтерфейсу додатку та основний потік взаємодії з ним як користувачем, так і розробником. Детально описано порядок взаємодії користувача з додатком, а саме від моменту запуску до отримання покрокових рекомендацій з надання домедичної допомоги. Показано, що інтерфейс спроектовано максимально просто та інтуїтивно для введення 12 симптомів займає в середньому 15–20 секунд, результати видаються миттєво, а критичні стани супроводжуються автоматичним нагадуванням про виклик екстрених служб. Така організація взаємодії забезпечує реальну можливість використання додатка людьми без медичної підготовки навіть у стані стресу.

Також представлено комплексну оцінку роботи наївного байєсівського класифікатора та створеного для нього синтетичного датасету. За допомогою вбудованого модуля аналітики продемонстровано, що модель досягає точності топ-1 прогнозу на рівні 80–82 % при середній впевненості 83–85 %, має рівномірний розподіл, клінічно осмислені кореляції між симптомами та логічну теплову карту тяжкості.

Отримані результати підтверджують ефективність застосування наївного байєсівського підходу в умовах жорстких обмежень мобільного офлайн-додатка та доводять можливість створення надійного асистента домедичної допомоги.

ВИСНОВКИ

У результаті виконання магістерської роботи досліджено, розроблено та реалізовано повністю автономну інтелектуальну систему для домедичної допомоги, ядром якої став спеціально адаптований і модифікований наївний байєсівський класифікатор.

Проведений аналіз предметної області та існуючих рішень показав відсутність по-справжньому автономних клінічно обґрунтованих офлайн-додатків і обґрунтував вибір наївного байєсівського підходу як оптимального за швидкістю, розміром та точністю на мобільних пристроях. На основі цього сформовано синтетичний датасет обсягом 170 клінічних випадків, що охоплює 34 найнебезпечніші невідкладні стани та 12 симптомів із градацією тяжкості 0–3 при рівномірному розподілі класів і реалістичній інтенсивності ознак.

Запропоновано та реалізовано модифікацію наївного байєсівського класифікатора з урахуванням ваги тяжкості симптомів, що суттєво підвищило роздільну здатність моделі без зростання обчислювальної складності. Розроблено та вбудовано в додаток модуль, який у реальному часі обчислює та візуалізує метрики якості, забезпечуючи постійний контроль клінічної адекватності.

Спроектовано інтерфейс, орієнтований на використання в екстремальних умовах: середній час введення симптомів становить 15–20 секунд, результати видаються миттєво, критичні стани супроводжуються автоматичним нагадуванням про виклик екстрених служб. Тестування на групі користувачів без медичної підготовки підтвердило повну зрозумілість і працездатність системи з першої спроби.

Практична цінність роботи полягає у створенні готового до масового використання програмного продукту, який реально підвищує шанси на виживання при невідкладних станах завдяки швидкому та обґрунтованому наданню домедичної допомоги. Додаток не залежить від інтернету та працює на будь-якому сучасному смартфоні.

Подальше вдосконалення можливе шляхом розширення кількості діагнозів і симптомів, впровадження голосового введення, створення спеціалізованих версій, інтеграції з іншими пристроями та розробки гібридних байєсівсько-нейромережових моделей при збереженні офлайн-режиму.

Отримані результати доводять високу ефективність модифікованого байєсівського підходу для створення легковажних інтелектуальних медичних асистентів і відкривають перспективи його застосування в умовах обмежених ресурсів та критичного часу реагування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Kormysh Roman, Chyzhmotria Olena, Dmytrenko Iryna. The robustness of the naive bayes classifier to data imbalances in first aid datasets. Тези VIII Всеукраїнської науково-технічної конференції "Комп'ютерні технології: інновації, проблеми, рішення", 02-03 грудня 2025 року. Житомир: "Житомирська політехніка", 2025. (подано до друку)
2. Kormysh Roman, Chyzhmotria Olena, Dmytrenko Iryna. The role of symptom severity in improving naive bayes-based first aid diagnosis accuracy. Тези VIII Всеукраїнської науково-технічної конференції "Комп'ютерні технології: інновації, проблеми, рішення", 02-03 грудня 2025 року. Житомир: "Житомирська політехніка", 2025. (подано до друку)
3. Kyrimi, E., McLachlan, S., Dube, K., Neves, M.R., Fahmi, A., Fenton, N. A comprehensive scoping review of Bayesian networks in healthcare: Past, present and future. *Artificial Intelligence in Medicine*, Vol. 110, No. 1, 2021. <https://pubmed.ncbi.nlm.nih.gov/34127238/>
4. Kyrimi, E., Dube, K., Fenton, N., Fahmi, A., Neves, M.R., Marsh, W., McLachlan, S. Bayesian Networks in Healthcare: What is Preventing Their Adoption? *Artificial Intelligence in Medicine*, Vol. 117, No. 7, 2021. <https://doi.org/10.1016/j.artmed.2021.102079>
5. Fenton, N., Neil, M. Risk Assessment and Decision Analysis with Bayesian Networks. 2018. [Електронний ресурс] – Режим доступу: <https://www.routledge.com/Risk-Assessment-and-Decision-Analysis-with-Bayesian-Networks/Fenton-Neil/p/book/9781498728461>
6. Polotskaya, K., Muñoz-Valencia, C.S., Rabasa, A., Quesada-Rico, J.A., Orozco-Beltrán, D., Barber, X. Bayesian Networks for the Diagnosis and Prognosis of Diseases: A Scoping Review. *Machine Learning and Knowledge Extraction*, Vol. 6, No. 2, 2024. <https://www.mdpi.com/2504-4990/6/2/58>
7. Chen, Q., Feng, X., Sun, L., et al. Dynamic Bayesian network for predicting physiological changes, organ dysfunctions and mortality risk in critical trauma

- patients. *Frontiers in Public Health*, Vol. 10, No. 1, 2022. <https://pmc.ncbi.nlm.nih.gov/articles/PMC9063308/>
8. Sheidaei, A., Brier, M., et al. A novel Dynamic Bayesian Network approach for data mining of survival data in gastric cancer patients. *BMC Medical Informatics and Decision Making*, Vol. 22, No. 1, 2022. <https://bmcmedinformdecismak.biomedcentral.com/articles/10.1186/s12911-022-02000-7>
 9. Alves, J.M., et al. Temporal and dynamic Bayesian networks for prognosis and diagnosis in clinical settings: A scoping review. *Computers in Biology and Medicine*, Vol. 198, No. 1, 2025. <https://pubmed.ncbi.nlm.nih.gov/41072141/>
 10. Zhang, M.N., et al. Comprehensive review of Bayesian network applications in gastrointestinal cancers. *Cancers*, Vol. 16, No. 15, 2024. <https://pmc.ncbi.nlm.nih.gov/articles/PMC12198876/>
 11. Chen, R., et al. Bayesian network-based survival prediction model for patients with cirrhosis-induced portal hypertension after TIPS. *Scientific Reports*, Vol. 14, No. 1, 2024. <https://pmc.ncbi.nlm.nih.gov/articles/PMC11036496/>
 12. Magrini, A., Luciani, D., Stefanini, F.M. A probabilistic network for the diagnosis of acute cardiopulmonary diseases. 2016. [Электронный ресурс] – Режим доступа: <https://arxiv.org/abs/1609.06864>
 13. van Gerven, M.A.J., Taal, B.G., Lucas, P.J.F. Dynamic Bayesian networks as prognostic models for clinical patient management. *Journal of Biomedical Informatics*, Vol. 41, No. 4, 2008. <https://pubmed.ncbi.nlm.nih.gov/18337188/>
 14. Thottakkara, P., Ozrazgat-Baslanti, T., Hupf, D., et al. Application of machine learning techniques to predict in-hospital mortality and complications in critically ill patients. *PLoS One*, Vol. 9, No. 9, 2014. <https://pubmed.ncbi.nlm.nih.gov/24025536/>
 15. Koller, D., Friedman, N. Probabilistic Graphical Models: Principles and Techniques. 2009. [Электронный ресурс] – Режим доступа: <https://mitpress.mit.edu/9780262013192/probabilistic-graphical-models>

16. Pearl, J. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. 1988. [Электронный ресурс] – Режим доступа: <https://www.elsevier.com/books/probabilistic-reasoning-in-intelligent-systems/pearl/978-0-89871-190-4>
17. Lauritzen, S.L., Spiegelhalter, D.J. Local computations with probabilities on graphical structures and their application to expert systems. Journal of the Royal Statistical Society B, Vol. 50, No. 2, 1988. <https://academic.oup.com/jrssb/article/50/2/157/7027752>
18. Neapolitan, R.E. Learning Bayesian Networks. 2004. [Электронный ресурс] – Режим доступа: <https://www.pearson.com/store/p/learning-bayesian-networks/P100000656554>
19. Demigha, S., et al. The Bayesian classification to predict and detect anomalies in health data. 2016. [Электронный ресурс] – Режим доступа: <https://ieeexplore.ieee.org/document/7862044>
20. Bartram, G. Probabilistic Prognosis with Dynamic Bayesian Networks. 2012. [Электронный ресурс] – Режим доступа: <https://papers.phmsociety.org/index.php/phmconf/article/view/1224>
21. Sandri, F., Berchialla, P., et al. Dynamic Bayesian Networks to predict sequences of organ failures in ICU patients. AMIA Annual Symposium Proceedings, Vol. 2015, 2015. <https://www.semanticscholar.org/paper/Dynamic-Bayesian-Networks-to-predict-sequences-of-Sandri-Berchialla/267d3e64b40781f0114cdf8ccf6088df5f1b58ce>
22. Agard, G., et al. Improving sepsis prediction in the ICU with explainable machine learning. Journal of Clinical Medicine, Vol. 12, No. 18, 2023. <https://www.mdpi.com/2077-0383/12/18/5884>
23. Bhowmik, T., et al. Bayesian Networks for Clinical Decision Making Support. 2021. [Электронный ресурс] – Режим доступа: <https://arxiv.org/abs/2101.07987>

24. Palamidessi, C., et al. Medical idioms for clinical Bayesian network development. 2023. [Электронный ресурс] – Режим доступа: <https://arxiv.org/abs/2303.10059>
25. Ward, L.M., et al. Machine learning to improve predictive performance of prehospital triage. BMJ Open, Vol. 14, No. 5, 2024. <https://bmjopen.bmj.com/content/14/5/e083666>
26. Fuller, S., et al. The accuracy of prehospital triage decisions in English trauma. Injury, Vol. 55, No. 2024, 2024. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC11131377/>
27. Nwoke, U., et al. Bayesian modelling framework for optimizing pre-hospital care and resource allocation. Journal of Biomedical Informatics, Vol. 152, No. 2024, 2024. <https://pubmed.ncbi.nlm.nih.gov/39019000/>
28. Kyrimi, E., McLachlan, S., Dube, K., Neves, M.R., Fahmi, A., Fenton, N. Reusability of Bayesian Networks case studies: a survey. Applied Intelligence, Vol. 55, No. 1, 2025. <https://doi.org/10.1007/s10489-025-06289-5>
29. Garcez, A.S., et al. Bayesian Logical Neural Networks for Human-Centered Applications. Frontiers in Bioinformatics, Vol. 3, No. 1, 2023. <https://www.frontiersin.org/articles/10.3389/fbinf.2023.1082941/full>
30. Ngartera, L., Nadarajah, S., Issaka, M.A. Application of Bayesian Neural Networks in Healthcare: Three Case Studies. Machine Learning and Knowledge Extraction, Vol. 6, No. 4, 2024. <https://www.mdpi.com/2504-4990/6/4/127>
31. Development of Bayesian Network Methodologies for Applications at the Interface of Social and Medical Sciences. 2024. [Электронный ресурс] – Режим доступа: <https://research-portal.st-andrews.ac.uk/en/studentTheses/development-of-bayesian-network-methodologies-for-applications-at>
32. Bayesian Network Modelling for the Clinical Diagnosis of Dementia. 2023. [Электронный ресурс] – Режим доступа: <https://www.medrxiv.org/content/10.1101/2023.12.30.23300452v1.full-text>

33. Artificial Intelligence Doctor Assistant using Bayesian Networks. 2023. [Электронный ресурс] – Режим доступа: <https://www.ijisae.org/index.php/IJISAE/article/view/4999>
34. Development of Bayesian Network-Based Regional Healthcare Systems. ICEB 2023. [Электронный ресурс] – Режим доступа: <https://aisel.aisnet.org/iceb2023/11/>
35. Langarizadeh, M., Moghbeli, F. Applying Naive Bayesian Networks to Disease Prediction. Acta Informatica Medica, Vol. 24, No. 1, 2016. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4944756/>
36. De Silva, N.T., Jayamanne, D.J. Computer-Aided Medical Diagnosis Using Bayesian Classifier. IEEE Conference Proceedings, 2016. <https://ieeexplore.ieee.org/document/7475764>
37. Mesgarpour, M., et al. A Review of Dynamic Bayesian Network Techniques. Dagstuhl Reports, Vol. 3, No. 11, 2013. <https://drops.dagstuhl.de/entities/document/10.4230/DagRep.3.11.1>
38. Holzinger, A., et al. Explainable AI (XAI) for the medical domain: integrating probabilistic models and causal reasoning. iScience, Vol. 24, No. 1, 2021. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7929659/>
39. Richens, J., Lee, C., Johri, S. Improving the Accuracy of Medical Diagnosis with Causal Machine Learning. Nature Communications, Vol. 11, No. 1, 2020. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7419549/>
40. Bhowmik, T., et al. Bayesian Networks for Clinical Decision Making Support. 2021. [Электронный ресурс] – Режим доступа: <https://arxiv.org/abs/2101.07987>
41. Palamidessi, C., et al. Medical idioms for clinical Bayesian network development. 2023. [Электронный ресурс] – Режим доступа: <https://arxiv.org/abs/2303.10059>

ДОДАТКИ

Ядро системи - bayesian-engine.ts:

```

import { Platform } from 'react-native';
import { loadEmergencyData, DataRow } from './data-loader';

export type SymptomInput = Record<string, number>;
export type DiagnosisProbabilities = Record<string, number>;
export type AverageSeverity = Record<string, Record<string, number>>;

export type BayesianModel = {
  avgSeverity: AverageSeverity;
  symptomCols: string[];
  diagnosisCol: string;
  data: DataRow[];
};

async function parseEmergencyData(): Promise<{data: DataRow[], symptomCols: string[],
diagnosisCol: string}> {
  const data = await loadEmergencyData();

  const headers = Object.keys(data[0]);

  const diagnosisCol = headers[headers.length - 1];
  const symptomCols = headers.slice(0, -1);

  return { data, symptomCols, diagnosisCol };
}

function calculateAverageSeverity(data: DataRow[], symptomCols: string[], diagnosisCol:
string): AverageSeverity {
  const avgSeverity: AverageSeverity = {};

  const grouped = data.reduce((acc, row) => {
    const diagnosis = row[diagnosisCol as keyof DataRow] as string;
    if (!acc[diagnosis]) {
      acc[diagnosis] = [];
    }
    acc[diagnosis].push(row);
    return acc;
  }, {} as Record<string, DataRow[]>);

  Object.keys(grouped).forEach(diagnosis => {
    const rows = grouped[diagnosis];
    avgSeverity[diagnosis] = {};

    symptomCols.forEach(symptom => {
      const sum = rows.reduce((acc: number, row: DataRow) => acc + (row[symptom] as
number), 0);
      avgSeverity[diagnosis][symptom] = sum / rows.length;
    });
  });

  return avgSeverity;
}

function predictDiagnosis(userInput: SymptomInput, model: BayesianModel):
DiagnosisProbabilities {
  const { data, symptomCols, diagnosisCol } = model;

  const allSymptomsZero = Object.values(userInput).every(value => value === 0);

```

```

    console.log('predictDiagnosis - allSymptomsZero:', allSymptomsZero, 'userInput:',
userInput);

    if (allSymptomsZero) {
        const diagnoses = [...new Set(data.map(row => row[diagnosisCol as keyof DataRow] as
string))];
        const result: DiagnosisProbabilities = {};
        diagnoses.forEach(diagnosis => {
            result[diagnosis] = 0.001; // Very low probability
        });
        console.log('predictDiagnosis - returning low probabilities for all-zero
symptoms:', result);
        return result;
    }

    const diagnoses = [...new Set(data.map(row => row[diagnosisCol as keyof DataRow] as
string))];

    const diagnosisCounts = diagnoses.reduce((acc, diagnosis) => {
        acc[diagnosis] = data.filter(row => (row[diagnosisCol as keyof DataRow] as string)
=== diagnosis).length;
        return acc;
    }, {} as Record<string, number>);

    const totalCases = data.length;
    const priorProbs = Object.keys(diagnosisCounts).reduce((acc, diagnosis) => {
        acc[diagnosis] = diagnosisCounts[diagnosis] / totalCases;
        return acc;
    }, {} as Record<string, number>);

    const likelihoods: Record<string, number> = {};

    diagnoses.forEach(diagnosis => {
        const diagnosisRows = data.filter(row => (row[diagnosisCol as keyof DataRow] as
string) === diagnosis);

        let likelihood = 1.0;

        Object.keys(userInput).forEach(symptom => {
            const userValue = userInput[symptom];

            const matchingRows = diagnosisRows.filter(row =>
                (row[symptom as keyof DataRow] as number) === userValue
            );

            const prob = (matchingRows.length + 1) / (diagnosisRows.length + 4);
            likelihood *= prob;
        });

        likelihoods[diagnosis] = likelihood;
    });

    const posteriors: DiagnosisProbabilities = {};
    let totalPosterior = 0;

    diagnoses.forEach(diagnosis => {
        const posterior = priorProbs[diagnosis] * likelihoods[diagnosis];
        posteriors[diagnosis] = posterior;
        totalPosterior += posterior;
    });

    Object.keys(posteriors).forEach(diagnosis => {

```

```

        posteriors[diagnosis] = posteriors[diagnosis] / totalPosterior;
    });

    return posteriors;
}

function severityWeightMap(severity: number): number {
    const weights = {
        0: 0.05,
        1: 0.1,
        2: 0.4,
        3: 1.0
    };
    return weights[severity as keyof typeof weights] || 0.05;
}

function severitySimilarity(userInput: SymptomInput, avgSevForDiag: Record<string,
number>): number {
    if (!userInput || Object.keys(userInput).length === 0) {
        return 0;
    }

    let weightedSum = 0;
    let totalWeight = 0;

    Object.keys(userInput).forEach(symptom => {
        const inputVal = userInput[symptom];
        const avgVal = avgSevForDiag[symptom];

        if (avgVal === undefined || isNaN(avgVal)) {
            return;
        }

        const maxSeverity = Math.max(inputVal, avgVal, 1);
        const diff = Math.abs(inputVal - avgVal) / maxSeverity;
        const similarity = 1 - diff;

        const weight = Math.max(severityWeightMap(Math.round(inputVal)),
severityWeightMap(Math.round(avgVal)));
        weightedSum += similarity * weight;
        totalWeight += weight;
    });

    return totalWeight > 0 ? weightedSum / totalWeight : 0;
}

function adjustProbabilities(
    baseResults: DiagnosisProbabilities,
    avgSeverity: AverageSeverity,
    userInput: SymptomInput
): DiagnosisProbabilities {
    const allSymptomsZero = Object.values(userInput).every(value => value === 0);
    console.log('adjustProbabilities - allSymptomsZero:', allSymptomsZero, 'userInput:',
userInput);

    if (allSymptomsZero) {
        const adjusted: DiagnosisProbabilities = {};
        Object.keys(baseResults).forEach(diagnosis => {
            adjusted[diagnosis] = 0.001; // Very low probability
        });
        console.log('adjustProbabilities - returning low probabilities for all-zero
symptoms:', adjusted);
    }
}

```

```

    return adjusted;
}

const adjusted: DiagnosisProbabilities = {};

Object.keys(baseResults).forEach(diagnosis => {
    const baseProb = baseResults[diagnosis];
    const similarity = severitySimilarity(userInput, avgSeverity[diagnosis] || {});
    const adjustedProb = baseProb * baseProb * similarity;
    adjusted[diagnosis] = Math.max(adjustedProb, 0.0);
});

const total = Object.values(adjusted).reduce((sum, prob) => sum + prob, 0);

if (total > 0) {
    Object.keys(adjusted).forEach(diagnosis => {
        adjusted[diagnosis] = adjusted[diagnosis] / total;
    });
} else {
    Object.keys(baseResults).forEach(diagnosis => {
        adjusted[diagnosis] = baseResults[diagnosis];
    });
}

return adjusted;
}

export class BayesianEngine {
    private model: BayesianModel | null = null;
    private initializationPromise: Promise<void> | null = null;

    constructor() {
        this.initializationPromise = this.initializeModel();
    }

    private async initializeModel(): Promise<void> {
        try {
            const { data, symptomCols, diagnosisCol } = await parseEmergencyData();

            const avgSeverity = calculateAverageSeverity(data, symptomCols, diagnosisCol);

            this.model = {
                avgSeverity,
                symptomCols,
                diagnosisCol,
                data
            };
        } catch (error) {
            console.error('Failed to initialize Bayesian model:', error);
        }
    }

    private async ensureModelInitialized(): Promise<void> {
        if (this.initializationPromise) {
            await this.initializationPromise;
        }
    }

    public async getSymptoms(): Promise<string[]> {
        await this.ensureModelInitialized();
        return this.model?.symptomCols || [];
    }
}

```

```

    }

    public async getDiagnoses(): Promise<string[]> {
        await this.ensureModelInitialized();
        return this.model ? Object.keys(this.model.avgSeverity) : [];
    }

    public async predictDiagnosis(userInput: SymptomInput):
    Promise<DiagnosisProbabilities> {
        await this.ensureModelInitialized();

        if (!this.model) {
            console.error('Model not initialized');
            return {};
        }

        try {
            console.log('BayesianEngine.predictDiagnosis - input:', userInput);

            const baseResults = predictDiagnosis(userInput, this.model);
            console.log('BayesianEngine.predictDiagnosis - baseResults:', baseResults);

            const adjustedResults = adjustProbabilities(baseResults, this.model.avgSeverity,
            userInput);
            console.log('BayesianEngine.predictDiagnosis - adjustedResults:',
            adjustedResults);

            return adjustedResults;
        } catch (error) {
            console.error('Error during diagnosis prediction:', error);
            return {};
        }
    }

    public async getTopDiagnoses(userInput: SymptomInput, topN: number = 10):
    Promise<Array<{diagnosis: string, probability: number}>> {
        const results = await this.predictDiagnosis(userInput);

        return Object.entries(results)
            .map(([diagnosis, probability]) => ({ diagnosis, probability }))
            .sort((a, b) => b.probability - a.probability)
            .slice(0, topN);
    }
}

export const bayesianEngine = new BayesianEngine();

export {
    parseEmergencyData,
    calculateAverageSeverity,
    predictDiagnosis,
    adjustProbabilities,
    severitySimilarity
};

```