Хмельницький національний університет
Факультет програмування та комп'ютерних і телекомунікаційних систем
Кафедра комп'ютерних наук та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему *Експертна система визначення рівня епідеміологічної небезпеки*

Галузь знань *12 – Інформаційні технології*
Шифр і назва галузі знань

Спеціальність *122 – Комп'ютерні науки*
Шифр і назва спеціальності

Освітня програма *Комп'ютерні науки*
Назва освітньої програми

Виконав: *студент 4 курсу, група КН-17-1* — Підпис — *О.М. Овчарук*
Курс, група виконавця; Ініціали, прізвище

Керівник: *к.т.н., доцент кафедри КНІТ* — Підпис — *О.В. Мазурець*
Науковий ступінь, посада; Ініціали, прізвище

Нормоконтроль: *к.т.н., доцент кафедри КНІТ* — Підпис — *Р.О. Багрій*
Науковий ступінь, посада; Ініціали, прізвище

До захисту допускаю:
Зав. кафедри КНІТ, д.т.н., професор — Підпис — *О.В. Бармак*
Ініціали, прізвище

08 червня 2021 р.

Хмельницький 2021

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет *програмування та комп'ютерних і телекомунікаційних систем*
Кафедра *комп'ютерних наук та інформаційних технологій*
Освітній ступінь *бакалавр*
Галузь знань *12 – Інформаційні технології*
Спеціальність *122 – Комп'ютерні науки*

ЗАТВЕРДЖУЮ
Завідувач кафедри *комп'ютерних наук та інформаційних технологій*

(підпис)
д.т.н., професор О.В. Бармак
«*08*» *лютого* *2021* року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

1. Тема кваліфікаційної роботи бакалавра: «*Експертна система визначення рівня епідеміологічної небезпеки*»

2. Завдання видано студенту *Овчаруку Олександру Миколайовичу*
(прізвище, ім'я, по батькові)

3. Керівник роботи *доцент кафедри КНІТ Мазурець Олександр Вікторович*
(посада, прізвище, ім'я, по батькові)

4. Затверджено наказом університету від «*05*» *02* *2021* р. № *11*

5. Зміст пояснювальної записки (перелік задач) та вихідні дані:

Метою роботи є розробка експертної системи визначення рівня епідеміологічної небезпеки. Вимогою до системи є можливість створювати навчальні моделі з різною кількістю вхідних параметрів та епох, а при створенні вхідних параметрів надавати можливість вказувати їхні одиниці виміру та тип. Також слід забезпечити можливість формування експертного висновку з рекомендацією найбільш вірогідного результату, що особливо актуально для визначення рівня епідеміологічної ситуації у локальних межах в умовах адаптивного карантину.

Виконавець:	*студент 4 курсу, група КН-17-1* (Курс, група виконавця)	(Підпис)	*О.М. Овчарук* (Ініціали, прізвище)
Керівник:	*к.т.н., доцент кафедри КНІТ* (Науковий ступінь, посада)	(Підпис)	*О.В. Мазурець* (Ініціали, прізвище)

Анотація

Тема кваліфікаційної роботи бакалавра: «*Експертна система визначення рівня епідеміологічної небезпеки*»

Виконавець кваліфікаційної роботи бакалавра: *студент групи КН-17-1 Овчарук Олександр Миколайович*

Керівник кваліфікаційної роботи бакалавра: *к.т.н., доцент кафедри КНІТ Мазурець Олександр Вікторович*

Кваліфікаційна робота бакалавра містить:

Пояснювальна записка				Кількість додатків
Сторінок	Рисунків	Таблиць	Джерел інформації	
55	26	12	34	4

Метою кваліфікаційної роботи бакалавра є розробка експертної системи визначення рівня епідеміологічної небезпеки. Для розробки інформаційної системи на платформі .NET було використано мову програмування C#, а також систему керування базами даних MS SQL Server.

Розроблена система призначена для визначення рівня епідеміологічної ситуації у локальних межах в умовах адаптивного карантину із використанням нейромережевих технологій. Експертна система визначення рівня епідеміологічної небезпеки дозволяє створювати навчальні моделі з різною кількістю вхідних параметрів та епох, при створенні вхідних параметрів надаючи можливість вказувати їхні одиниці виміру та тип, й здійснювати формування експертного висновку з рекомендацією найбільш вірогідного результату

Ключові слова: епідеміологічна ситуація, карантин, нейронні мережі, експертна система.

Виконавець: *студент 4 курсу, група КН-17-1* (Курс, група виконавця) _____ (Підпис) *О.М. Овчарук* (Ініціали, прізвище)

Зміст

Перелік скорочень

Скорочення, термін, позначення	Пояснення
БД	База даних
ІС	Інформаційна система
ІТ	Інформаційні технології
КРБ	Кваліфікаційна робота бакалавра
КН	Комп’ютерні науки
КНІТ	Комп’ютерні науки та інформаційні технології
НМ	Нейронна мережа
ПЗ	Пояснювальна записка
ПП	Програмний продукт
СКБД	Система керування базами даних
ХНУ	Хмельницький національний університет.
CLR	Common Language Runtime
CLR	Common Language Runtime
FCL	Framework Class Library
MS	Microsoft

Вступ

З початку XX століття в світі починають активно розвиваються методи прогнозування інфекційної захворюваності. Тема дослідження темпів поширення інфекційних захворювань сьогодні є особливо актуальною, оскільки в світі вирує пандемія коронавірусу.

Коронавірус є гострим вірусним захворювання, яке характеризується переважним ураженням дихальної системи та шлунково-кишкового тракту. Особливістю даної інфекціє є достатньо довгий інкубаційний період (від 2 до 14 днів) протягом якого інфікована людина не помічаючи ніяких симптомів буде заражувати оточуючих.

Основними засобами для запобігання поширенню коронавірусної інфекції є проведення вакцинації населення та впровадження карантинних обмежень [1]. Для боротьби з коронавірусом на території України було впроваджено адаптивний карантин, який дозволяє впроваджувати один з видів карантинних обмежень в залежності від стану епідеміологічної ситуації в регіоні.

На основі висновку головного санепідомолога області відбувається визначення стану епідеміологічної ситуації в регіоні. При формуванні ним експертного рішення враховується велика кількість параметрів, зібраних за певний період часу. Особливо складним питанням є те, що для визначення рівня епідеміологічної небезпеки слід враховувати не тільки поточний стан справ, а й ситуацію, що йому передувала. Враховуючи велику кількість вхідних даних та їхню властивість до постійної зміни, час протягом якого вони будуть аналізуватися експертом буде доволі суттєвим.

Отже, на сьогоднішній момент розробка комп’ютерного додатку, що реалізовуватиме можливості аналізу епідеміологічної ситуації в регіоні та буде брати до уваги стан не тільки поточної епідеміологічної ситуації, а також і тої, що їй передувала, є актуальною.

Інфекційні захворювання і їх здатність до масового поширення є причиною виникнення епідемій, а основним засобом запобігання поширенню

інфекції є впровадження карантинних обмежень. Якщо на території діє адаптивний карантин, то це дозволяє впроваджувати один з видів карантинних обмежень, в залежності від епідеміологічної ситуації в регіоні, а визначення стану епідеміологічної ситуації в регіоні відбувається по висновку головного санепідомолога. При формуванні експертного рішення враховується велика кількість параметрів, зібраних за великий період часу, зокрема наявність ліжок з підведеним киснем, кількість вільних та занятих лікарняних ліжок, прогрес захворюваності тощо. Тому умовах значної кількості вхідних параметрів При формуванні експертного рішення внаслідок їх постійного оновлення та значних термінів, потрібних для аналізу, актуальною є розробка експертної системи визначення рівня епідеміологічної небезпеки.

Розділ 1
Характеристика предметної області та постановка задачі

1.1 Аналіз предметної області

З давніх часів епідемії створювали смертельну небезпеку для людства. Перші історично зафіксовані випадки епідемій датуються 430 роком до н.е., тоді в ході Пелопоннеської війни Афінська армія через епідемію черевного тифу втратила чверть особового складу. Приблизно у той же період Гіппократ пише свій трактат «Епідемії» у якому проводить дослідження різних епідемічних захворювань які виникали в містах Греції [2]. Сьогодні поняття епідемії трактується як масове поширення інфекційної хвороби серед населення відповідної території за короткий проміжок часу [3].

Саме інфекційні захворювання та їх природня здатність до масового поширення і є причиною виникнення епідемії. Інфекційні хвороби викликають розлади здоров'я людей, які спричинені біологічними збудниками (вірусами, бактеріями, грибками, кліщами та іншими патогенними паразитами) та продуктами їх життєдіяльності (токсинами), які здатні до масового поширення і легко передаються від заражених осіб здоровим [4]. Класифікація інфекційних захворювань ґрунтується на механізмі передачі інфекції і специфіці первинної локалізації збудника в організмі, який викликає основні клінічні прояви хвороби. Згідно з цією класифікацією, інфекційні хвороби поділяються на чотири групи.

Кишкові інфекційні хвороби. Зараження людини відбувається перорально через продукти харчування (воду та їжу). До цієї групи належать: черевний тиф, дизентерія, холера, вірусний гепатит А і Е.

Кров’яні (трансмісивні) інфекційні хвороби. Зараження людини відбувається, коли збудник потрапляє безпосередньо в кров (лімфу) від живих кровосисних переносників. До цієї групи належать малярія, чума, геморагічна гарячка та інші.

Інфекційні хвороби зовнішніх покривів. Зараження людини відбувається при безпосередньому контакті або через фактори зовнішнього середовища: одяг,

посуд, продукти харчування та ін. До цієї групи захворювань належать: сказ, венеричні хвороби, правець та інші.

Інфекційні хвороби дихальних шляхів. Збудник переважно локалізується в слизовій оболонці дихальних шляхів, виділяється у зовнішнє середовище під час кашлю, чхання, розмови. Зараження людини відбувається під час потрапляння збудника інфекції на слизову оболонку. До цієї групи належать гострі респіраторні захворювання, кір, грип, коронавірус та інші [5].

На сьогоднішній день в світі вирує пандемія коронавірусу. Коронавірус (COVID-19) є гострим вірусним захворювання, яке характеризується переважним ураженням дихальної системи та шлунково-кишкового тракту. Особливістю даної інфекціє є достатньо довгий інкубаційний період (від 2 до 14 днів) протягом якого інфікована людина не помічаючи ніяких симптомів буде заражувати оточуючих. Для боротьби з COVID-19 активно проводиться вакцинація населення та тестування ззовні здорових осіб на виявлення інфекції [1]. Основним засобом запобігання поширенню інфекції є впровадження карантинних обмежень. На території України діє адаптивний карантин, який дозволяє впроваджувати один з видів карантинних обмежень в залежності від стану епідеміологічної ситуації в регіоні. Всього існує чотири карантинні зони, обмеження яких запроваджуються в залежності від рівня завантаженості лікарень.

«Зелена зона» вводиться, якщо відсутній сплеск захворювань, завантаженість ліжок у лікарні на рівні 5%. При впровадженні карантинної зони забороняється:

– перебування у громадських будівлях без маски або респіратора;

– проведення масових заходів за участі більше 50 осіб;

– наповнення залів в кінотеатрах та закладах культури більше чим на 50%;

– перевезення стоячих пасажирів;

– відвідування закладів дошкільної, шкільної, позашкільної та спеціалізованої освіти, якщо на самоізоляції перебуває більше 50% дітей та персоналу закладу.

«Жовта зона» вводиться, якщо 5 днів поспіль змінюються показники захворювань, завантаженість лікарняних ліжок на рівні від 5% до 50%. При впровадженні карантинної зони зберігаються обмеження зеленої зони, а також забороняється:

– проведення масових заходів за участі більше 30 осіб;

– відвідування установ соціального захисту, в яких перебувають люди похилого віку, крім випадків надавання екстреної допомоги.

«Помаранчева зона» впроваджується, якщо завантаження лікарняних ліжок становить від 50% до 75%. При впровадженні карантинної зони зберігаються обмеження жовтої зони, а також забороняється:

– проведення масових заходів за участі більше 20 осіб;

– робота закладів розміщення (хостели, туристичні бази тощо), окрім готелів;

– робота спортзалів, фітнес-центрів та закладів культури.

«Червона зона» вводиться, якщо протягом 14 днів кількість виявлених випадків захворювання більше 320 на 100 тис. населення, завантаженість лікарняних ліжок понад 75%. При впровадженні карантинної зони зберігаються обмеження помаранчевої зони, а також забороняється:

– робота громадського транспорту;

– відвідування закладів освіти;

– відвідування закладів культури та кінотеатрів;

– робота ТРЦ, кафе та ресторанів [6].

Визначення стану епідеміологічної ситуації в регіоні відбувається на основі висновку головного санепідомолога області. При формуванні експертного рішення враховується велика кількість параметрів, зібраних за певний період часу, таких як: кількість вільних та занятих лікарняних ліжок, наявність ліжок з підведеним киснем, прогрес захворюваності та інші. Враховуючи велику

кількість вхідних даних та їхню постійну зміну, час протягом якого вони будуть аналізуватися експертом буде доволі суттєвим. Отже, в умовах значної кількості вхідних параметрів їх постійного оновлення та значних термінів їхнього аналізу, актуальною є розробка експертної системи.

1.2 Аналіз існуючого програмного забезпечення предметної області

1.2.1 Програмне забезпечення для відстеження епідеміологічної ситуації

Дослідження темпів поширення коронавірусу проводиться з використанням інформаційних технологій. Дані про кількість хворих публікують кожного дня на багатьох вебсайтах, сторінках в соціальних мережах та інших публічних платформах.

Однією з таких платформ є «Система моніторингу поширення епідемії коронавірусу» розроблена за підтримки РНБО України [7]. За допомогою даної системи можна переглянути інформацію про кількість заражених, померлих та вилікуваних людей. Дана система моніторингу наглядно зображує на карті кількість людей які хворіють у конкретному регіоні або країні (рисунок 1.1).

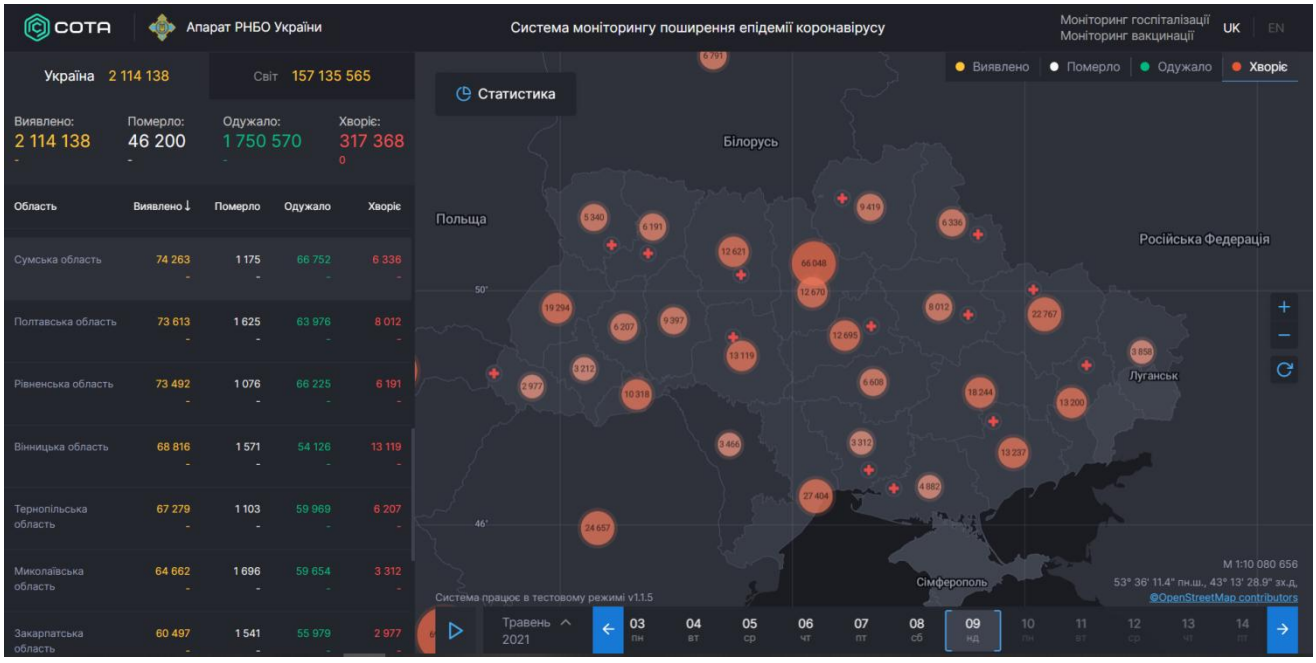


Рисунок 1.1 – Система моніторингу поширення епідемії коронавірусу [7]

Також в даної системи є можливість продемонструвати темпи поширення інфекції у світі з позначенням на карті кількості виявлених інфікованих протягом кожного дня з початку поширення хвороби. Крім того у даної платформи є функціонал для побудови графіків поширення коронавірусної інфекції за вказаний період часу в вибраному регіоні (рисунок 1.2).

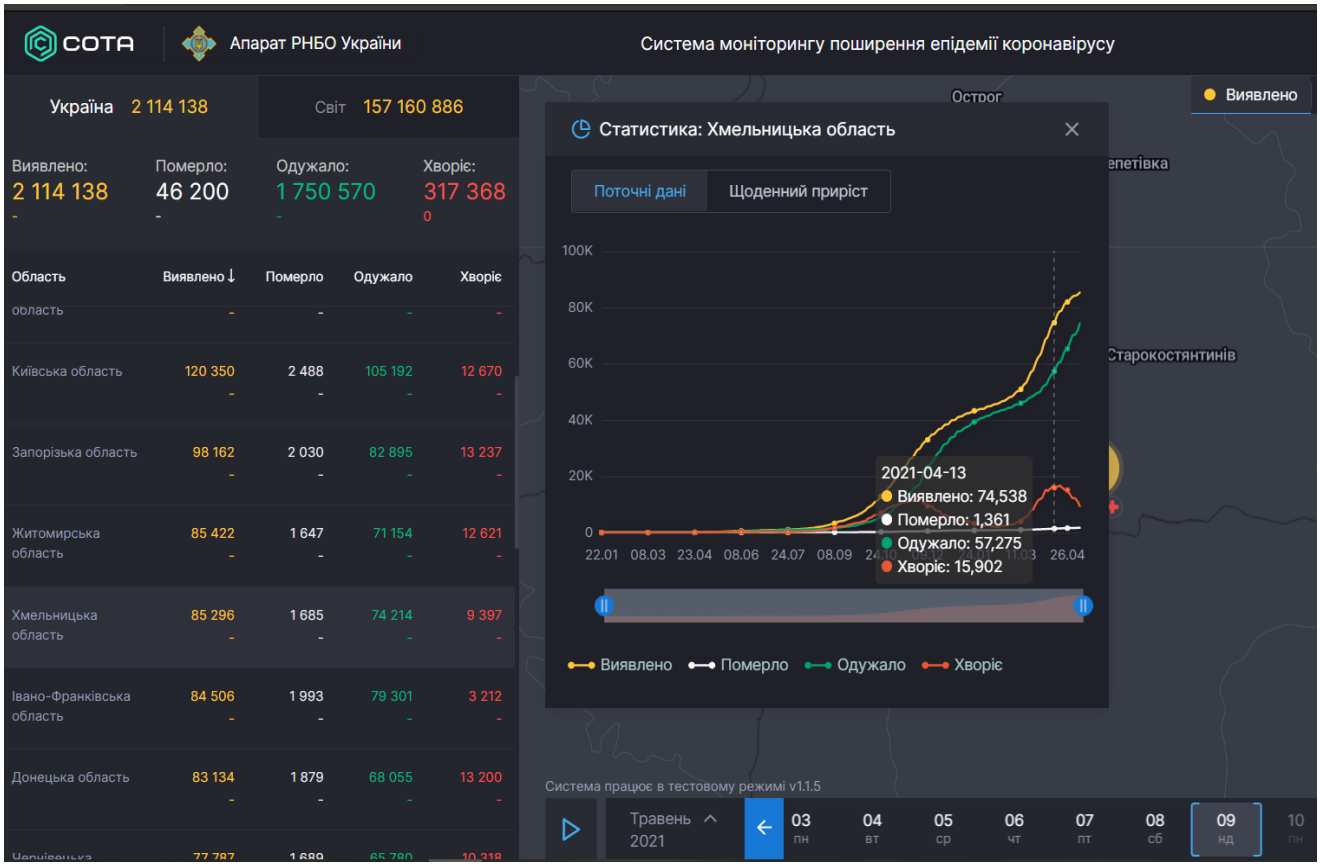


Рисунок 1.2 – Графік перебігу коронавірусу в Хмельницькій області [7]

До переваг «Системи моніторингу поширення епідемії коронавірусу» можна віднести перегляд детальної інформації стосовно кількості заражених, хворих, померлих та вилікуваних людей, а також надання можливості представлення цієї інформації у вигляді графіків. Недоліками цієї системи є відсутність прогнозування перебігу хвороби, а також неможливість розділення регіонів на карантинні зони, що є значним упущенням в умовах адаптивного карантину.

Також порталом з потужнім функціоналом є «Система охорони здоров'я України» [8]. На даному порталі публікуються звіти про стан медичних закладів. За його допомогою можна переглядати детальну інформацію про лікарняні ліжка доступні у медичних закладах, а саме: кількість завантажених ліжок, наявність місць обладнаних апаратом штучної вентиляції легень та кількість ліжок з підведеним киснем. Також «Система охорони здоров'я України» надає можливість ознайомитися з інформацією про число пацієнтів госпіталізованих медичним закладом, окрім того відмічаються дані про частку місць виділених для хворих на COVID-19 та зазначається відсоток їхньої завантаженості (рисунок 1.3).

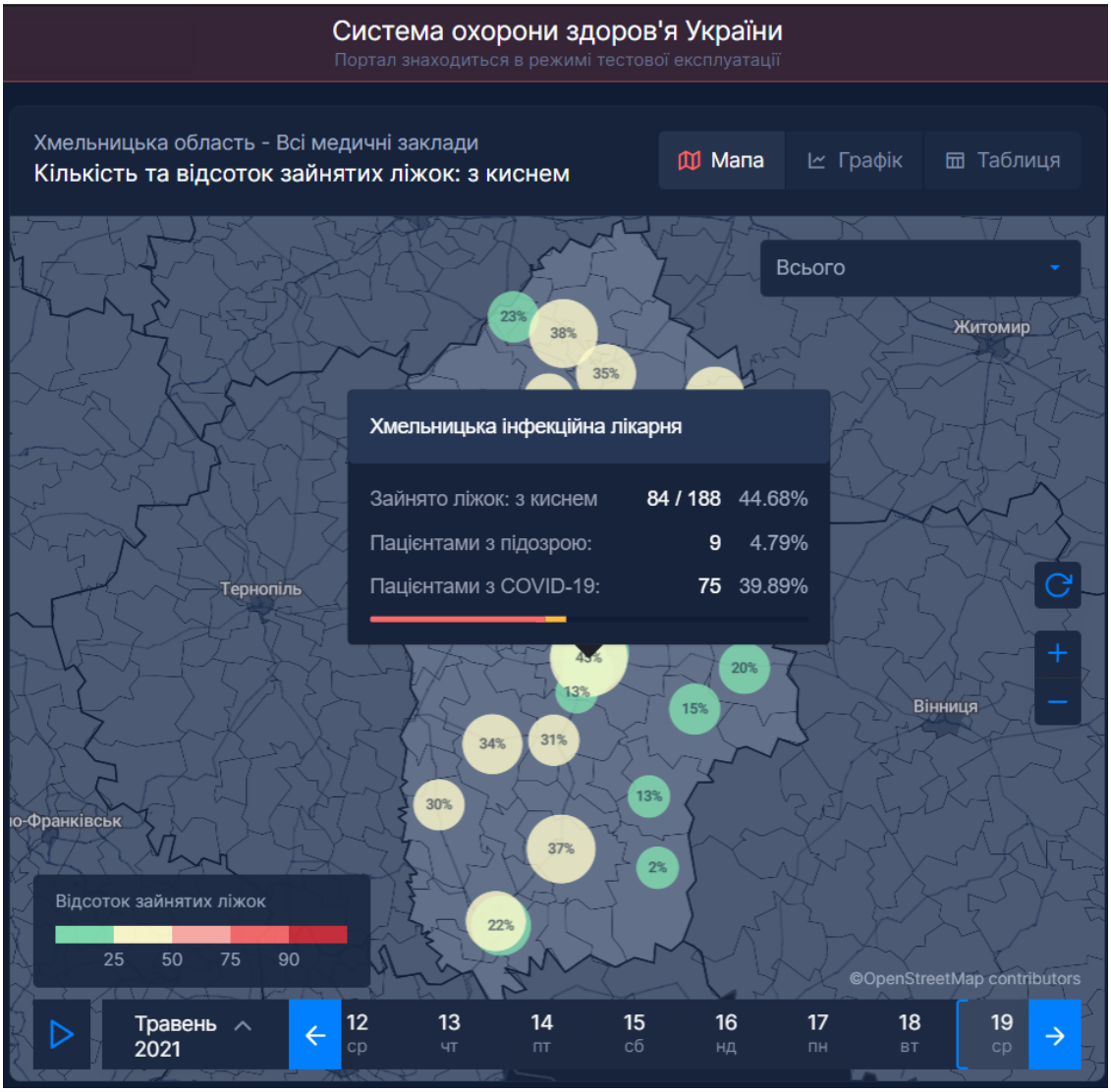


Рисунок 1.3 – «Система охорони здоров'я України» перегляд інформації про кількість госпіталізованих [8]

Окрім цього на порталі розміщується інформація про темпи проведення вакцинації в країні, а саме: кількість вакцинованого населення, його вікова група та назва використаної вакцини. Наведену на порталі інформацію можна переглядати як за конкретний день так і за певний проміжок часу з демонстрацією її на відповідному графіку (рисунок 1.4).

Рисунок 1.4 – Темпи вакцинації насення [8]

Безперечними перевагами порталу «Система охорони здоров’я України» є можливість перегляду інформації про наявність вакцин та відсоток завантаженість лікарняних ліжок у медичних закладах, що особливо актуально в умовах пандемії коронавірусу. До недоліків даної система можна віднести відсутність інформації, щодо поширення коронавірусу у сусідніх країнах, оскільки збільшення темпів поширення хвороби в них може призвести до погіршення епідеміологічної ситуації в Україні.

Ще одним прикладом інформаційної платформи, яка виконує збір даних пов’язаних з поширенням коронавірусної інфекції, є вебсайт «Центра системних наук та інженерії університету Джонса Хопкінса» [9]. За допомогою даного вебсайту можна переглянути число хворих, заражених та померлих людей.

Інформацію стосовно захворюваності можна переглянути, як для всього світу так і для конкретної країни (рисунок 1.5).

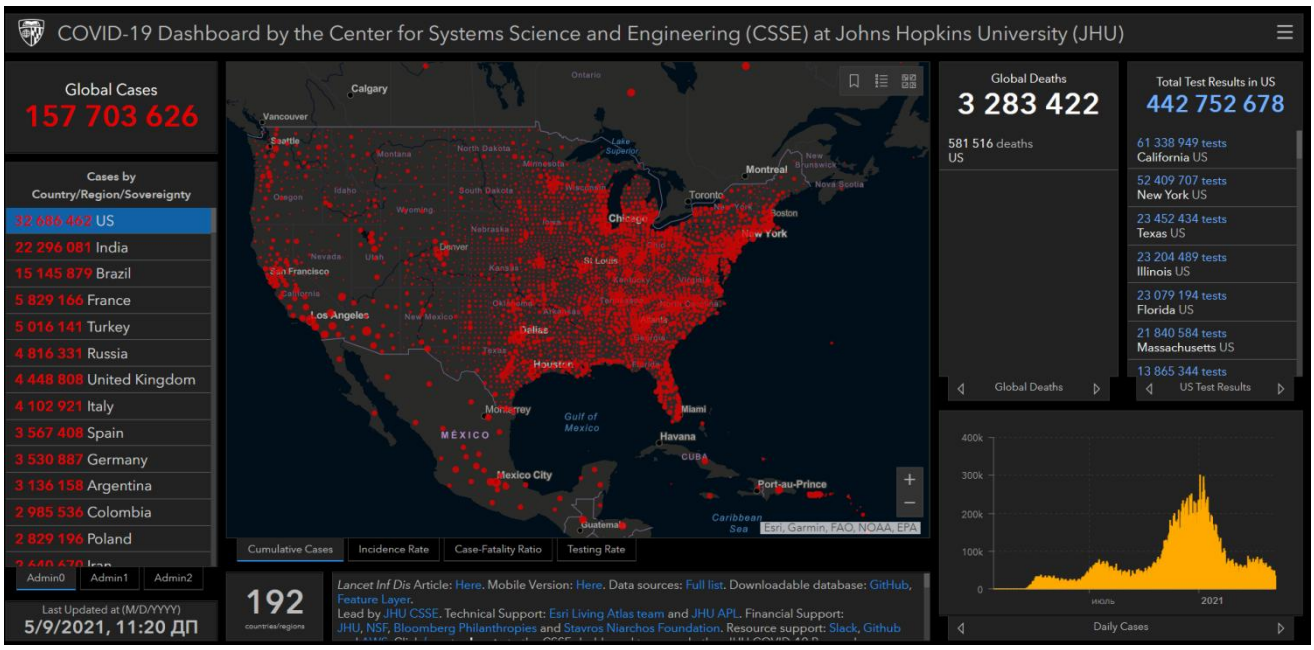


Рисунок 1.5 – Кількість випадків зараження на території США [9]

До переваг сайту «Центра системних наук та інженерії університету Джонса Хопкінса» можна віднести доступність його API, що дозволяє використовувати інформацію відображену на сайті для дослідження темпів поширення COVID-19. Недоліками даного сайту є неможливість перегляду інформації про число захворілих за певний проміжок часу, також даний сайт не надає детальної інформації про кількості захворілих у регіоні.

Розглянувши існуючі програмні продукти потрібно відмітити, що це в основному вебсайти котрі виводять інформацію про поширення коронавірусної інфекції. Проте вони не можуть пропонувати певний план дій на основі поточної епідеміологічної ситуації в регіоні, що значно обмежує можливості їх використання. .Отже, на сьогоднішній момент розробка комп’ютерного додатку, що реалізовуватиме можливості аналізу епідеміологічної ситуації в регіоні, є актуальною. Особливо складним питанням є те, що для визначення рівня епідеміологічної небезпеки слід враховувати не тільки поточний стан справ, а й

ситуацію, що йому передувала. Також є доцільним при прийнятті рішень керуватись даними розвитку подій за аналогічних ситуацій.

1.2.2 Використання нейронних мереж у експертних системах

Методи прогнозування інфекційної захворюваності активно розвиваються з початку XX століття. Зважаючи на вирування в світі пандемії COVID-19 дана тема стала особливо актуальною. Епідеміологічні прогнози проводитися для різних часових відрізків і в залежності від цього використовуються для різних цілей. Так, короткостроковий прогноз, що проводиться на декілька тижнів вперед застосовується в оперативному управлінні та виявленні епідемічних спалахів захворюваності. Найбільш корисним можна вважати середньостроковий прогноз терміном від двох місяців до пів року [10], який використовується в тактичному управлінні. Точність такого прогнозу менша ні короткострокового, але він надає більше часу для проведення превентивних заходів для підготовки до надзвичайних ситуацій. При прийнятті стратегічних рішень активно використовуються довгострокові прогнози, що проводяться на декілька років вперед. Досягнення високої якості таких прогнозів в більшості випадків неможливо, проте вони потрібні, наприклад, при оцінці необхідних обсягів виробництва лікарських препаратів та вакцин, плануванні оснащення медичних установ та підготовки персоналу.

Так, в залежності від термінів прогнозу і доступної для аналізу статистики доцільно використовувати різні підходи прогнозування. Основу для аналізу складають тимчасові ряди захворюваності, які можуть доповнюватися даними різної природи – наприклад, характеристиками погодних умов [11]. Частота збору даних обумовлюється видом інфекції, поточною епідеміологічною ситуацією та організаційними можливостями.

Одним із способів прогнозування темпів поширення інфекційних захворювань є використання методів, що спираються на машинне навчання, а саме штучних нейронних мереж. Методологія штучних нейронних мереж

широко відома і добре підходить для розв’язання задач у яких важко проводити аналітичне дослідження. Штучні нейронні мережі є направленим графом, вершини якого моделюють функції біологічних нейронів. Вершини приймають вхідні сигнали і при досить великому значенні їх зваженої суми перетворять їх в вихідний сигнал. Навчання штучних нейронних мереж полягає в обчисленні коефіцієнтів зв'язків між вершинами, що визначають силу вхідних сигналів.

Для прогнозування захворюваності можливо використання безліч варіантів штучних нейронних мереж. Вони можуть відрізнятися за: архітектурою, кількістю нейронів, функцією активації та способом навчання. Формального підходу до вибору характеристик штучних нейронних мереж не існує, в переважній більшості для навчання мережі використовується алгоритм зворотного поширення помилки [12].

Метою навчання штучної нейронної мережі є визначення явних і неявних залежностей між вхідними та вихідними даними. Показники захворюваності, як правило мають значні неточності, тому для навчання нейронної мережі потрібно використати значну кількість вхідних даних. Але через недостатню кількість статистичних даних використання штучних нейронних мереж для довгострокового прогнозування мало ефективне, при цьому під час проведення прогнозувань на короткі терміни вони показують достатньо точні результати.

Таким чином, для автоматизованого визначення рівня епідеміологічної ситуації є доцільним використання нейромережевих технологій.

1.3 Аналіз сучасних засобів створення програмного забезпечення

На сьогоднішній день існує ряд найбільш перспективних платформ для створення комп’ютерних додатків, а саме: .NET та Java.

Зокрема, .NET є програмною платформою, розробленою компанією Microsoft в 2002 році. Основною її ціллю є підтримка розробки сильно розділених компонентів програми. Вона забезпечує спільне використання різних мов програмування, а також загальну модель програмування. Платформа

складається з двох основних частин – загального середовища виконання та бібліотеки класів [13].

Загальне середовище виконання Common Language Runtime є основою платформи .NET Framework. CLR керує кодом під час виконання та надає доступ до служб управління пам'яттю та управління потоками. При цьому середовищем накладаються умови суворої типізації та інші види перевірки точності коду, що забезпечують його безпечність та надійність. Фактично основним завданням загального середовища виконання є управління кодом. Код, що надходить до середовища виконання, називають керованим кодом. Загальне середовище виконання дозволяє проводити розробку програми використовуючи всі переваги середовища виконання, бібліотеки класів і компонентів, написаних іншими розробниками на інших мовах [14].

Бібліотека класів є комплексною об'єктно-орієнтованої колекцією повторно використовуваних типів, які застосовуються для розробки додатків – починаючи з звичайних додатків, що запускаються за допомогою командного рядка, і додатків з графічним інтерфейсом (GUI) і закінчуючи додатками, що використовують останні технологічні можливості ASP.NET, такі як веб-форми і веб-служби XML. Окрім цього бібліотека класів надає типи, від яких керований код користувача може унаслідувати функції. Це не тільки спрощує роботу з типами даних на платформі .NET, але і скорочує час вивчення нових її засобів. Крім того, компоненти незалежних виробників можна легко поєднувати з класами платформи .NET [15].

Особливості платформи .NET є:

– велика бібліотека класів;

– використання об’єктно-орієнтованої моделі розробки програмного забезпечення;

– єдине середовище виконання для різних мов програмування;

– легкість інтегрування різномовних модулів;

– кросплатформне використання [16].

Платформа Java є програмним забезпеченням, що представляє собою середовище для роботи програм, які написані на Java. Вона складається з Java API і віртуальної машини Java (JVM) [17]. Java API є сукупністю бібліотек скомпільованого коду, які можна використовувати для вирішення типових і не тільки завдань.

Віртуальна машина Java є основною частиною виконуючого середовища Java Runtime Environment. Вона виконує байт-код попередньо створений із вхідного тексту програми. Віртуальна машина Java може працювати і з іншими мовами програмування, наприклад Ada. Основними функція JVM є можливість компіляції та запуску програми на любих пристроях та операційних системах, а також виконання управління пам’яттю у запущеному додатку) [18].

Особливості платформи Java є:

– надійність;

– можливість роботи на пристроях зрізною архітектурою;

– безпечність [19].

В процесі проведеного аналізу платформ розробки програмного забезпечення, були виділенні особливості кожної із них. Враховуючи отриману інформацію та специфіку інформаційної системи, найкращим рішенням буде використання платформи .NET.

1.4 Постановка задачі та вимоги до розробки інформаційної системи

Метою кваліфікаційної роботи бакалавра є розробка експертної системи визначення рівня епідеміологічної небезпеки на платформі .NET із використанням нейромережевих технологій. Даний програмний продукт повинний виконувати наступні функції:

– параметризована побудова вхідних моделей за розмірністю кількості параметрів епохи та кількості епох;

– параметризована побудова вихідних моделей за розмірністю кількості можливих висновків експертної системи;

– генерація архітектури нейронної мережі згідно параметрів вхідних та вихідних моделей;

– одержання від користувача атрибутів для кожного з параметрів вхідної моделі (назви параметрів, одиниці виміру параметрів, типи даних параметрів) та вихідної моделі (назви можливих висновків);

– використання бази даних для збереження та відтворення параметрів роботи експертної системи у вигляді параметрів вхідної моделі, параметрів вихідної моделі та їх атрибутів;

– одержання від користувача значень параметрів вхідної та вихідної моделі (значень назви параметрів, одиниць виміру параметрів, типів даних параметрів та назв епох);

– одержання від користувача значень параметрів тестового випадку (значень назви параметрів, одиниць виміру параметрів, типів даних параметрів та назв епох);

– формування експертного висновку нейромережею, який містить прогнозований варіант для тестового випадку та оцінки кожного з можливих варіантів.

Для вирішення поставленої задачі, вхідними даними експертної системи визначення рівня епідеміологічної небезпеки мають бути:

– назви параметрів;

– одниниці виміру параметрів;

– тип даних параметрів;

– назви епох;

– назви можливих висновків.

Вихідними даними експертної системи мають бути:

– прогнозований варіант;

– оцінка кожного з можливих варіантів.

Вимогою до системи є можливість створювати навчальні моделі з різною кількістю вхідних параметрів та епох, а при створенні вхідних параметрів надавати можливість вказувати їхні одиниці виміру та тип; за цими параметрами

необхідно надати користувачеві можливість створювати навчальні вибірки. Також слід забезпечити можливість формування експертного висновку з рекомендацією найбільш вірогідного результату, що особливо актуально для визначення рівня епідеміологічної ситуації у локальних межах в умовах адаптивного карантину. Такий підхід дозволяє врахувати те, що для визначення рівня епідеміологічної небезпеки слід брати до уваги не тільки поточний стан справ, а й ситуацію, що йому передувала, а також є доцільним при прийнятті рішень керуватись даними розвитку подій за аналогічних ситуацій.

Розділ 2

Проектування інформаційної системи

2.1 Моделі, методи, інформаційна технологія системи

Інформаційна технологія визначення рівня епідеміологічної небезпеки зображена на рисунку 2.1.

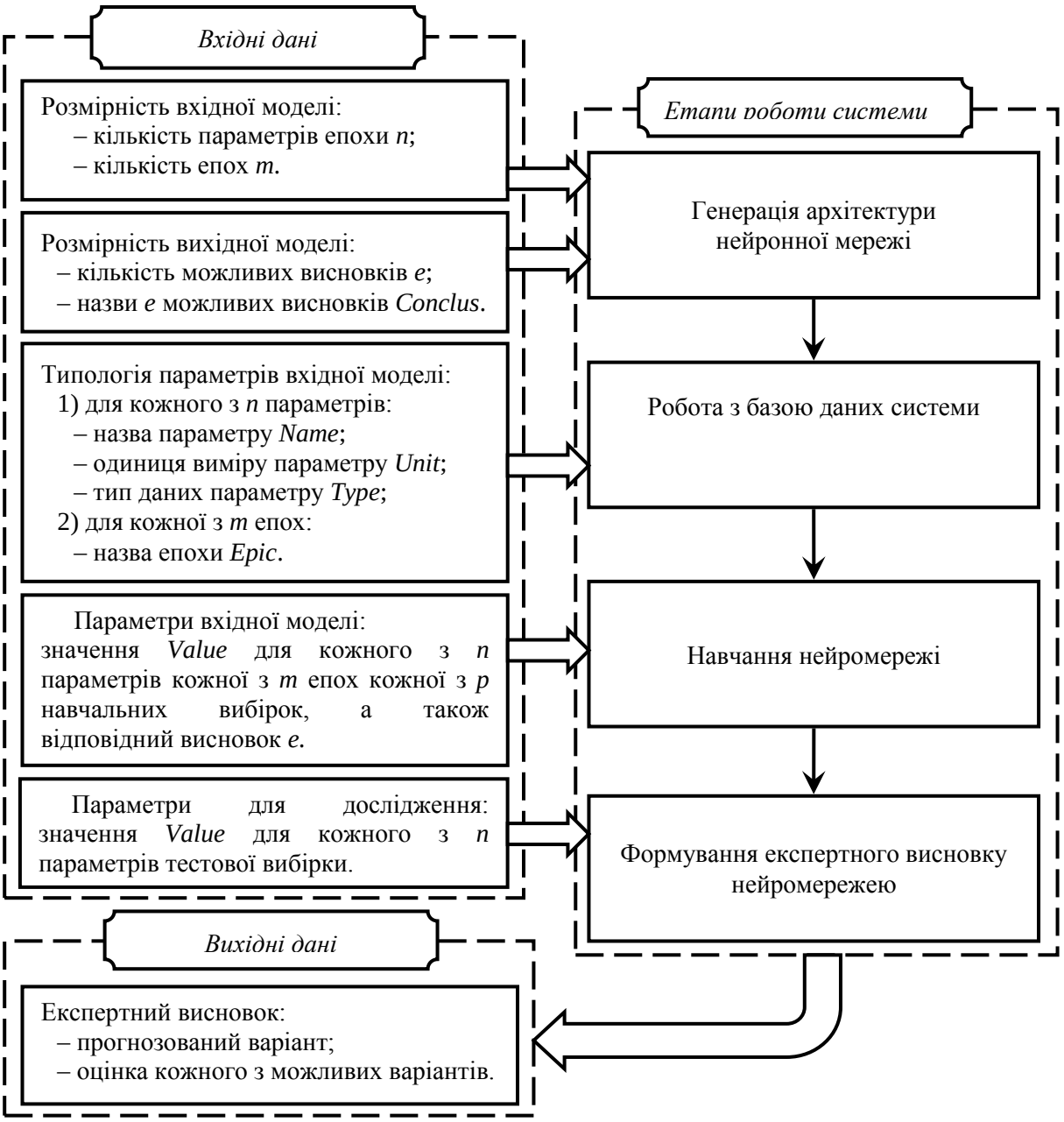


Рисунок 3.1 – Схема інформаційної технології

Розмірності вхідних та вихідних моделей використовуються нейронною мережею для генерації своєї архітектури. При створені вхідної моделі потрібно формувати списки епох та списки їхніх параметрів, для чого кожному параметру потрібно зазначити його атрибути (назви параметрів, одиниці виміру параметрів, типи даних параметрів). Вибір типу даних параметру буде випливати на одинці у яких потрібно буде його вводити. Формування вихідної моделі відбувається на основі кількості можливих висновків експертної системи. Після закінчення формування списків вхідної та вихідної моделі закінчується генерація архітектури нейронної мережі.

На етапі роботи з базою даних системи зазначається типологія параметрів вхідної моделі. Де кожному з списку параметрів має бути присвоєна назва, одиниці виміру та тип. Для елементів списку епох зазначаються їхні назви. У результаті чого відбувається заповнення БД відповідною інформацією.

Після формування архітектури вхідних та вихідних моделей відбувається генерування та заповнення вхідними значеннями навчальних вибірок. Розмірності навчальних вибірок визначаються відповідно до кількості епох та кількості їхніх параметрів. Значення що вводитимуться в навчальну вибірку відповідатимуть значенню кожного з параметрів вибраного з кожної епохи. Також при формуванні навчальної вибірки вказується відповідний експертний висновок. У результаті сформуються навчальні вибірки заповненні конкретними значеннями із вказаними експертними рішеннями. Список створених навчальних вибірок подається на входи нейронної мережі після чого проводиться її навчання.

Після закінчення навчання експертної системи необхідно заповнити тестову вибірку відповідними значеннями, архітектура тестової вибірки відповідає параметрам вхідної моделі. Після введення значень тестового випадку проводиться його розпізнавання у результаті чого формується відповідний

експертний висновок у якому зазначається прогнозований варіант та оцінка інших можливих варіантів.

Реалізація розробленої інформаційної технології дозволить проводити визначення рівня епідеміологічної небезпеки в умовах адаптивного карантину.

2.2 Інформаційна структура системи

2.2.1 Взаємозв'язок компонентів експертної системи

Взаємозв'язок компонентів експертної системи визначення рівня епідеміологічної небезпеки продемонстровано на рисунку 2.2.

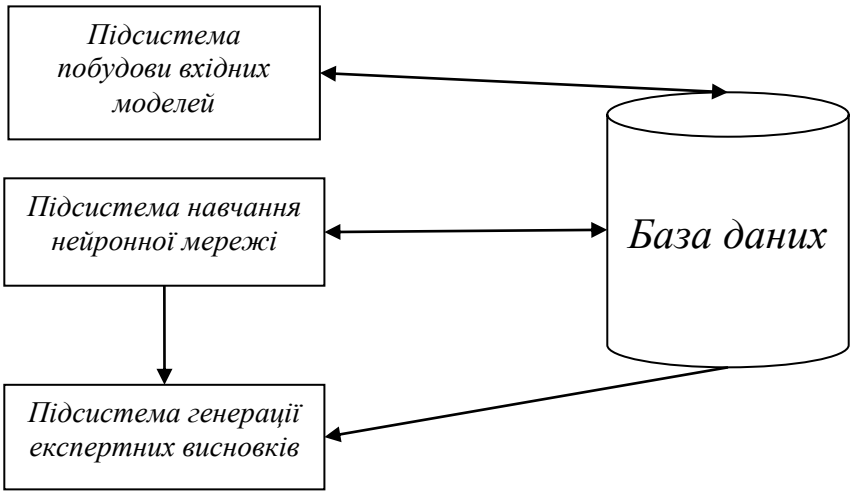


Рисунок 2.2 – Структура компонентів інформаційної системи визначення рівня епідеміологічної небезпеки

«Підсистема побудови вхідних моделей» призначена для формування архітектури вхідних даних нейронної мережі. За її допомогою відбувається одержання від користувача атрибутів для кожного з параметрів вхідної моделі (назви параметрів, одиниці виміру параметрів, типи даних параметрів) та вихідної моделі (назви можливих висновків), а також передача отриманих даних в БД.

«Підсистема навчання нейронної мережі» виконує формування навчальних вибірок, архітектура яких створена за допомогою «Підсистеми побудови вхідних моделей». За її допомогою відбувається одержання від користувача значень параметрів вхідної та вихідної моделі. Дана підсистема отримує з БД атрибути кожного параметру вхідної та вихідної моделі, а назад передає їхні значення.

«Підсистема генерації експертних висновків» призначена для отримання від користувача значень параметрів тестового випадку та формування експертного висновку нейромережею, який містить прогнозований варіант для тестового випадку та оцінку кожного з можливих варіантів. Дані навченої нейронної мережі отримуються з «Підсистеми навчання нейронної мережі», а параметри вхідної моделі з БД.

«База даних» використовується для збереження та відтворення параметрів роботи експертної системи у вигляді параметрів вхідної моделі, параметрів вихідної моделі та їх атрибутів.

Розроблена структура компонентів інформаційної системи визначення рівня епідеміологічної небезпеки дозволяє забезпечити логічний взаємозв’язок елементів експертної системи.

2.2.2 Структура бази даних інформаційної системи

У відповідності з поставленими завданнями для забезпечення збереження даних експертної системи було спроектовано та розроблено відповідну базу даних, даталогічна модель якої зображена на рисунку 2.3.

Враховуючи розроблену структурою бази даних були створенні та заповненні початковими даними таблиці: ParameterValues, Epoch, Results, TrainingSamples, DataType та Parameters. Таблиця ParameterValues призначена для зберігання значень параметрів вхідної моделі (таблиця 2.1).

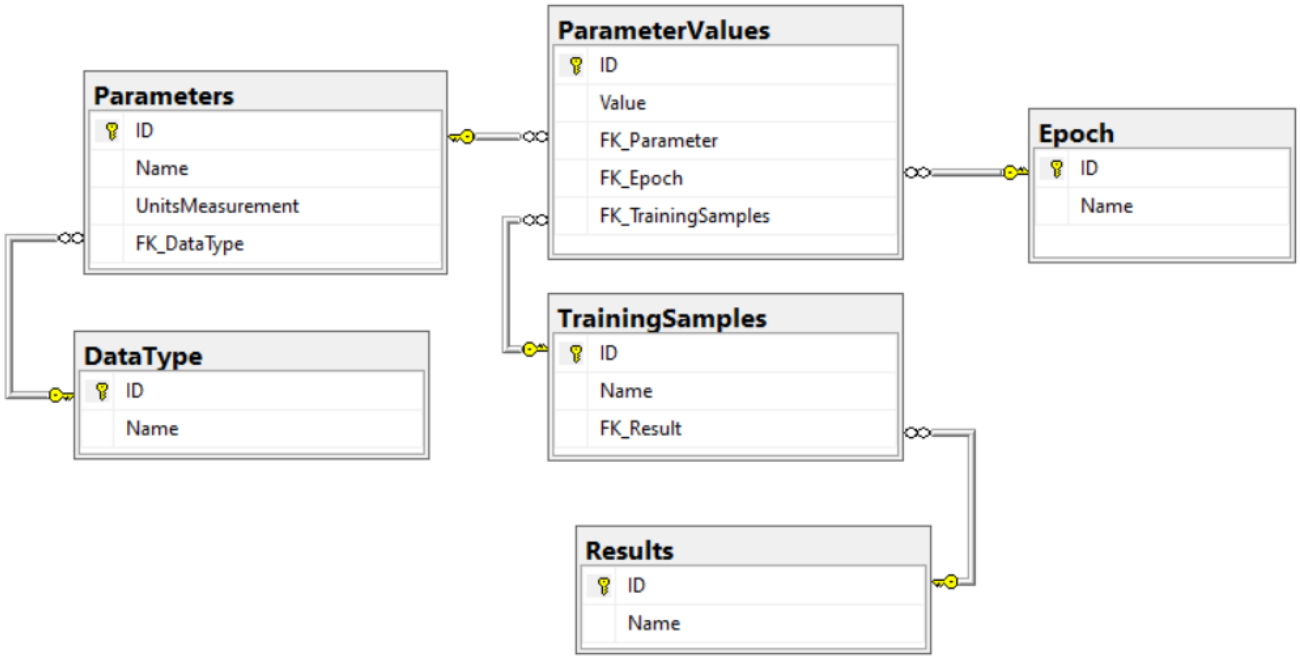


Рисунок 2.3 – Даталогічна модель БД експертної системи визначення рівня епідеміологічної небезпеки

Таблиця 2.1 – Атрибути таблиці ParameterValues

№ п/п	Назва атрибуту	Тип даних	Опис
1.	ID	int	Первинний ключ, числовий ідентифікатор для однозначного визначення запису таблиці
2.	Value	float	Значення параметру
3.	FK_Parameter	int	Вторинний ключ, посилання на запис таблиці «Parameters» призначений для співставлення з відповідним епідеміологічним параметром
4.	FK_Epoch	int	Вторинний ключ, посилання на запис таблиці «Epoch» призначений для співставлення з відповідною епохою
5.	FK_TrainingSamples	int	Вторинний ключ, посилання на запис таблиці «TrainingSamples» призначений для співставлення з відповідною навчальною вибіркою

Таблиця Epoch призначена для зберігання інформації про епохи (таблиця 2.2).

Таблиця 2.2 – Атрибути таблиці Epoch

№ п/п	Назва атрибуту	Тип даних	Опис
1.	ID	int	Первинний ключ, числовий ідентифікатор для однозначного визначення запису таблиці
2.	Name	nvarchar(250)	Назва епохи

Таблиця Results призначена для зберігання даних прогнозованих висновків (таблиця 2.3).

Таблиця 2.3 – Атрибути таблиці Results

№ п/п	Назва атрибуту	Тип даних	Опис
1.	ID	int	Первинний ключ, числовий ідентифікатор для однозначного визначення запису таблиці
2.	Name	nvarchar(250)	Назва висновку

Таблиця TrainingSamples призначена для зберігання даних навчальних вибірок (таблиця 2.4).

Таблиця 2.4 – Атрибути таблиці TrainingSamples

№ п/п	Назва атрибуту	Тип даних	Опис
1.	ID	int	Первинний ключ, числовий ідентифікатор для однозначного визначення запису таблиці
2.	Name	nvarchar(250)	Назва навчальної вибірки
3.	FK_Result	int	Вторинний ключ, посилання на запис таблиці «Results» призначений для співставлення з відповідним можливим висновком

Таблиця DataType призначена для зберігання типів даних параметрів (таблиця 2.5).

Таблиця 2.5 – Атрибути таблиці DataType

№ п/п	Назва атрибуту	Тип даних	Опис
1.	ID	int	Первинний ключ, числовий ідентифікатор для однозначного визначення запису таблиці
2.	Name	nvarchar(250)	Назва типу даних

Таблиця Parameters призначена для зберігання параметрів вхідної моделі (таблиця 2.6).

Таблиця 2.6 – Атрибути таблиці Parameters

№ п/п	Назва атрибуту	Тип даних	Опис
1.	ID	int	Первинний ключ, числовий ідентифікатор для однозначного визначення запису таблиці
2.	Name	nvarchar(250)	Назва параметру
3.	UnitsMeasurement	nvarchar(250)	Одиниця виміру параметру
4.	FK_DataType	int	Вторинний ключ, посилання на запис таблиці «DataType» призначений для співставлення з відповідною типом даних параметру

Розроблена структура бази даних дозволяє виконувати збереження та відтворення параметрів роботи експертної системи необхідних при визначені рівня епідеміологічної небезпеки.

2.3 Вибір засобів розробки інформаційної системи

Для розробки експертної системи визначення рівня епідеміологічної небезпеки на платформі .NET необхідно обрати мову програмування, СКБД та

платформу для розробки графічного користувацького інтерфейсу за допомогою яких проводитиметься розробка проекту.

2.3.1 Вибір мови програмування

Для створення експертної системи визначення рівня епідеміологічної небезпеки було проведено вибір мови програмування, в ході якого розглядалися C# та F#.

C # є сучасною об'єктно-орієнтованою мову програмування. За допомогою C# можна виконувати розробку різних видів програм починаючи від невеликих консольних додатків закінчуючи багатофункціональними програмними застосунками та іграми [20].

C# є відносно молодою мовою програмування, перша версія якої вийшла в 2002 році разом з платформою .NET, яка є основною для неї. На сьогоднішній день актуальною є версія C# 9.0 та .NET 5, у яких відбулися значні зміни відносно попередніх версій. C# продовжує активно розвиватися, і з кожною новою версією з'являється все більше нового функціоналу та оновлень, які багато в чому покращують структуру коду. Попередниками C# були Java та C++, з яких вона отримала частину функціональних можливостей [21].

Об'єктно-орієнтований підхід дозволяє реалізовувати завдання з побудови великих, гнучких, масштабованих і в той же час розширюваних додатків. Даний підхід підтримує механізм обробки подій, які змінюють атрибути об'єктів і моделюють їх взаємодію в предметній області. Механізм наслідування атрибутів і методів дозволяє будувати похідні поняття на основі базових і таким чином створювати модель як завгодно складної предметної області з заданими властивостями [22].

Перевагами мови програмування C# є:

- підтримання великої кількості бібліотек та фреймворків;
- можливість розробки різних типів додатків;
- велика кількість готових синтаксичних конструкцій;

– простота освоєння;

– наявність значних обсягів документації;

– постійний розвиток [23].

F# є функціонально-орієнтованою мовою програмування із строгою типізацією даних з сімейства мов .NET Framework. Він об’єднує в собі методи як функціонального так і об’єктно-орієнтованого програмування. F# був розроблений Доном Саймом в Кембриджі, сьогодні підтримкою та розробкою F# займається компанія Microsoft Developer Division. Мова F# активно інтегрується в Visual Studio, вперше він був включений в версію Visual Studio 2010. Окрім цього компілятори F# також можуть бути встановлені на Mac та Linux. В F# використовуються функціональні типи, кортежі, розрахункові вирази.

Преваги F # є

– асинхронні методи програмування;

– простий синтаксис;

– автоматичне узагальнення;

– статична типізація даних.

Недоліками F# є:

– складності в написанні об’єктно орієнтованого коду;

– складності під час підтримки та розширення додаткі.

Отже, мова програмування C#, яка поєднує потужність і гнучкість універсальних мов програмування з високою ефективністю виконавчого коду й можливістю безпосереднього доступу до апаратних ресурсів комп’ютера – робить її найкращим вибором для реалізації поставлених завдань.

2.3.2 Вибір системи керування базами даних

Для розробки бази даних, що використовуватиметься в експертній системі було зроблено вибір СКБД, вході якого було розглянуто MS SQL Server та MySQL.

MS SQL Server є системою керування базами даних розробленою компанією Microsoft. Створення запитів проводиться на мові SQL, що взята за основу в стандарті програмування ANSI / ISO [24]. SQL Server також використовує розширення цієї мови під назвою Transact-SQL, що реалізовує значно потужніший функціонал, та надає додаткові можливості для обробки виключень, оголошення перемінних та роботи із збереженими процедурами.

Перевагами MS SQL Server є:

– масштабованість;

– ефективні інструменти керування доступом;

– безпека і надійність;

– взаємодія з БД розташованими на Azure.

Недоліками MS SQL Server є:

– відсутність можливості кроссплатформного використання ;

– значна ціна повної версії програмного продукту[25].

MySQL є системою керування реляційними базами даних. Сьогодні цю систему підтримує та розробляє компанія Oracle, але створення даної СКБД виконала компанія «ТсХ», що використовувала її для підвищення швидкодії при роботі з великими базами даних. Основним завданням, що ставилося в процесі розробки даної MySQL було створення конкуренто спроможної СКБД для альтернативи комерційним системам. При написанні запитів в MySQL використовується декларативна мова програмування SQL, що чудово підходить для роботи як середніми так і з великими проектами.

Для MySQL розроблено велику кількість фреймворків, що можуть використовуватися на багатьох платформах. Також MySQL має якісну документацію що дозволяє розробникам зручно користуватися даною СКБД. Завдяки значній популярності MySQL є високою ймовірністю того що вона і надалі підтримуватиметься розробниками [26].

До переваг MySQL можна віднести:

– підтримувати роботу над великих БД та легку масштабованість;

– наявність значної кількості фреймворків;

– зручність експлуатації;

– швидкодію;

– відкритий код.

Недоліками MySQL є:

– відсутність транзакцій;

– відсутність підтримки представлень [27].

Враховуючи особливості поставлених завдань для розробки експертної системи визначення рівня епідеміологічної ситуації, було вирішено обрати СКБД MS SQL Server

2.3.3 Вибір платформи для розробки користувацького інтерфейсу

В ході розробки експертної системи визначення рівня епідеміологічної ситуації, було проведено вибір платформи для розробки користувацького інтерфейсу, вході якого було розглянуто Windows Forms та Windows Presentation Foundation.

Windows Presentation Foundation є платформою для розробки користувацького інтерфейсу. WPF з’явилася разом із .NET Framework 3.0 у 2003 році. Дана платформа використовує мову розмітки для XMAL. WPF підтримує широкий вибір компонентів для створення програм, включаючи моделі програми, ресурси, елементи управління, графіку, документи та безпеку. В основі даної платформи лежить потужна інфраструктура DirectX, що дозволяє обробляти розроблений інтерфейс з апаратним прискоренням[28].

Перевагами WPF є:

– можливість розширювати інтерфейс за допомогою XAML;

– використання стилів, шаблонів та тем;

– незалежність від розширення екрану;

– підтримка роботи з аудіо та відео;

– апаратне прискорення графіки [29].

До недоліків WPF можна віднести:

– засмічення оперативної пам'яті екземплярами ResourceDictionary;

– складність роботи з модальними повідомленнями (MessageBox)

– втрата пам’яті при роботі з подіями;

– достатньо високий поріг входу для нових розробників [30].

Windows Forms є платформою для розробки користувацького інтерфейсу на базі бібліотеки класів .NET Framework. Windows Forms дозволяє розробляти додатки з повнофункціональним графічним інтерфейсом, які прості в розробці та оновленні. Дана платформа дозволяє опрацьовувати події взаємодії з елементами керування за допомогою програмного коду. Windows Forms дозволяє створювати однорідну структуру програмної моделі, яка дозволяє уникнути більшості помилок зв’язаних з використанням Windows API. Дана платформа надає можливості для створення кроссплатформеного графічного користувацького інтерфейсу [31].

Перевагами Windows Forms є:

– наявність великої кількості готових елементів управління;

– значна кількість сторонніх бібліотек;

– зручність розробки інтерфейсу у дизайнері Visual Studio;

– простота розробки[32].

Недоліками Windows Forms є:

– складність масштабування;

– достатньо довгий час обробки подій[33].

Враховуючи особливості поставлених завдань для розробки користувацького інтерфейсу експертної системи визначення рівня епідеміологічної небезпеки було вирішено обрати Windows Forms.

Результатом виконання кваліфікаційної роботи бакалавра має бути експертна система, призначена для визначення рівня епідеміологічної ситуації у

локальних межах в умовах адаптивного карантину із використанням нейромережевих технологій. Для написання програмного продукту було обрано платформу NET, мову програмування C#, СКБД MS SQL Server та платформу розробки графічного інтерфейсу Windows Forms.

Розділ 3

Програмна реалізація інформаційної системи

3.1 Структура та функціональне призначення складових системи

На основі розробленої структури програми було створено діаграму класів, що буде використовуватися при створенні проекту. Розроблена діаграма зображена на рисунку 3.1

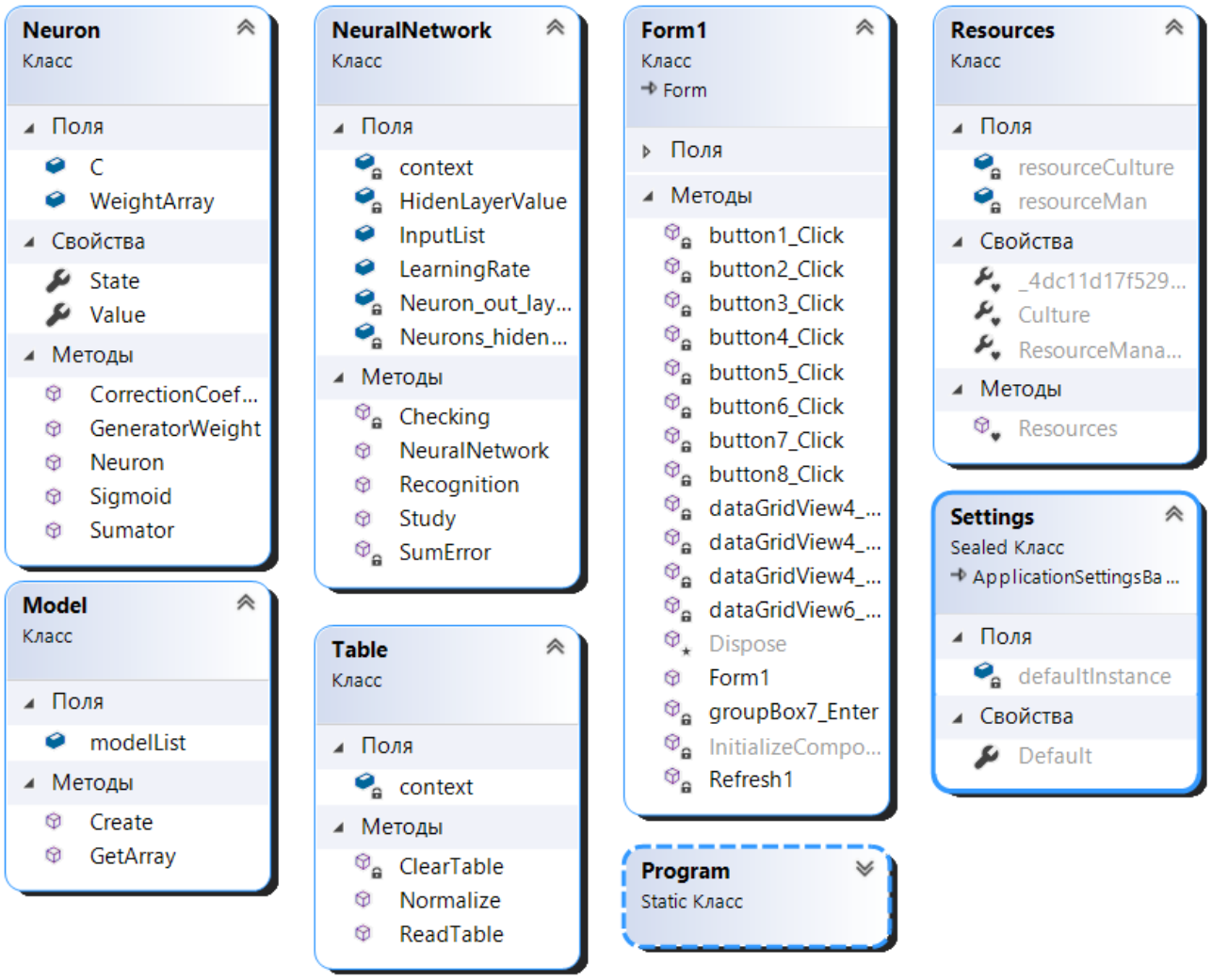


Рисунок 3.1 – Діаграма класів інформаційної системи

Основним класом в проекті є NeuralNetwork, при створені його об’єкту генеруються списки, у яких зберігаються значення нейронів прихованого та вихідного шару. У методі Study реалізується алгоритм навчання нейронної

мережі. Метод SumError виконує підрахунок помилок нейронів отриманих в процесі однієї ітерації навчання. Метод Checking контролює процес навчання, коли значення всіх нейронів вихідного шару досягають оптимального значення процес навчання зупиняється. За допомогою методу Recognition виконується перевірка тестової ситуації, дані які передав користувач подаються на входи навченої нейронної мережі.

Клас Neuron моделює архітектуру штучного нейрона. Поле WeightArray є масивом призначеним для зберігання значень вагових коефіцієнтів нейрона. Поле Value зберігає значення виходу нейрона. Метод Sumator виконує сумування вхідних значень нейрона. У методі Sigmoid виконується розрахунок значення функції активації, у даному випадку вона реалізована у вигляді сигмоїди. За допомогою методу CorrectionCoefficient відбувається перерахування ваг нейронна в процесі навчання. Метод GeneratorWeight проводить початкову ініціалізацію ваг нейронна випадковими значеннями.

На основі даних отриманих від користувача в класі Model відбувається генерація вхідної моделі. Клас Table виконує роботу зв’язану з записом та зчитуванням даних з БД.

Розроблена структура програми дозволяє повністю реалізувати функції експертної системи, а також робить можливим подальше розширення її функціоналу.

3.2 Особливості реалізації складових системи

Основним завданням, яке ставиться перед програмою є формування експертного висновку. Для цього в програмі використовується нейронна мережа, етапи роботи з якою видаленні в окремі модулі: «Вхідна модель», «Навчання» та «Експертний висновок».

Модуль «Вхідна модель» призначений для формування архітектури вхідних даних нейронної мережі. Після відкриття в програмі відповідної

закладки потрібно виконати заповнення представлених на ній списків. Значення даних списків відповідають за формування моделі вхідних даних (рисунок 3.2).

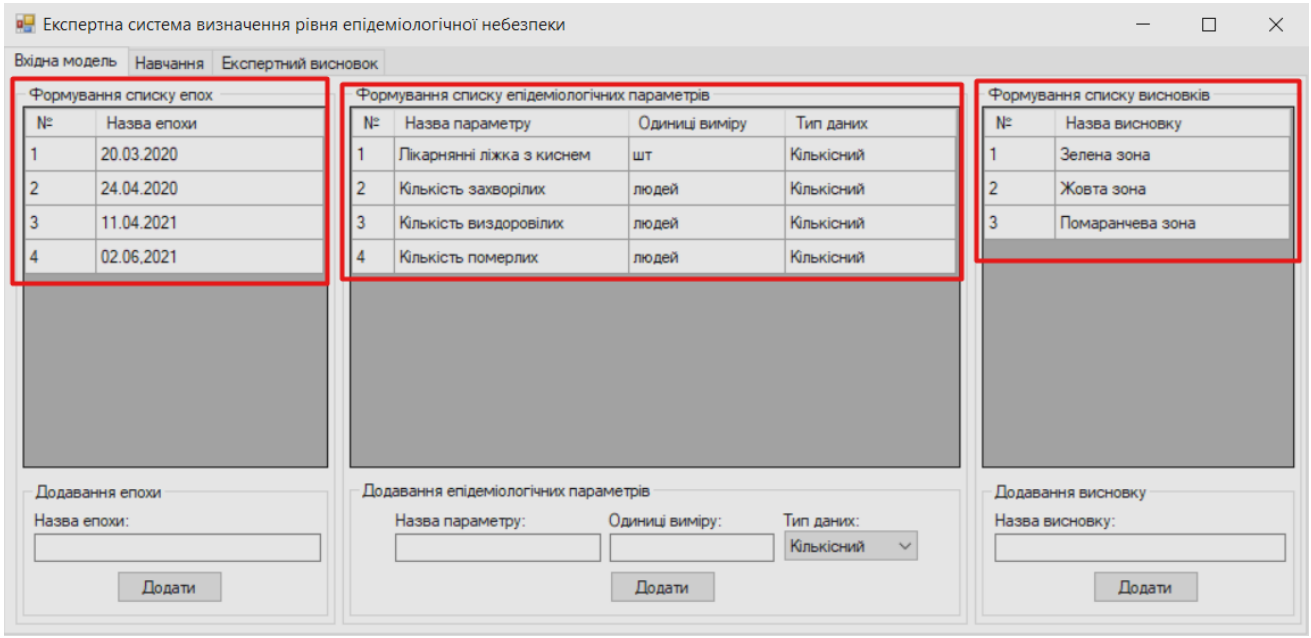


Рисунок 3.2 – Списки вхідних парметрів нейронної мережі

На основі сформованої архітектури вхідної моделі відбувається генерування відповідної їй архітектури нейронної мережі, формування якої виконується наступним програмним кодом:

```
public NeuralNetwork(List<double[]> modelList)
{
    Neurons_hiden_layer = new List<Neuron>();
    InputList = modelList;
    HidenLayerValue = new double[(int)Math.Ceiling((double)((InputList[0].Count() +
InputList.Count()) / 2))];
    for (int i = 0; i < HidenLayerValue.Count(); i++)
    {
        Neurons_hiden_layer.Add(new Neuron(InputList[0]));
        HidenLayerValue[i] = Neurons_hiden_layer[i].Value;
    }
    Neuron_out_layer = new List<Neuron>();
    for (int i = 0; i < InputList.Count(); i++)
    {
        Neuron_out_layer.Add(new Neuron(HidenLayerValue));
    }
}
```

В модулі «Навчання» виконується формування навчальних вибірок, які мають архітектуру розроблену в попередньому модулі та використовуються

нейронною мережею, як вхідні параметри. На основі їхніх значень відбувається навчання нейронної мережі (рисунок 3.3).

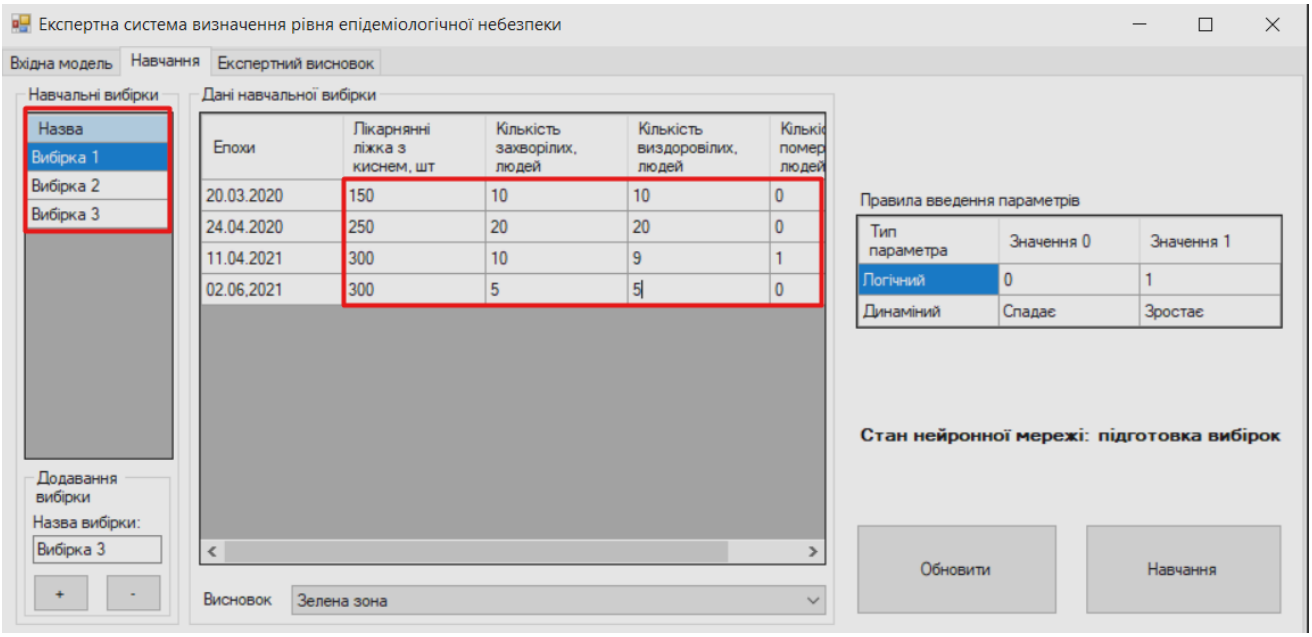


Рисунок 3.3 – Формування та заповнення навчальних вибірок

Коли формування навчальних вибірок закінчене можна розпочинати навчання нейронної мережі, для цього потрібно натиснути на кнопку «Навчання». Процес навчання продовжуватиметься до тих пір поки значення на всіх вихідних нейронах не досягне оптимального значення. Перевірка навченості нейронної мережі виконується наступним кодом:

```
bool Checking(List<double[]> InputList)
{
    int counter = 0;
    for (int k = 0; k < Neuron_out_layer.Count(); k++)
    {
        for (int i = 0; i < Neurons_hiden_layer.Count(); i++)
        {
            Neurons_hiden_layer[i].CorrectionCoefficient(InputList[k]);
            HidenLayerValue[i] = Neurons_hiden_layer[i].Value;
        }
        for (int i = 0; i < Neuron_out_layer.Count(); i++)
            Neuron_out_layer[i].CorrectionCoefficient(HidenLayerValue);
         if (Neuron_out_layer[k].Value >= 0.9)
            counter++;
     }
     if (counter == Neuron_out_layer.Count())
         return false;
     else
         return true;
}
```

Модуль «Експертний висновок» призначений для аналізу тестової ситуації та видачі самого імовірного експертного заключення. Дані тестової ситуації подаються на входи навченої нейронної мережі, після чого вона видає значення виходу з найбільшою імовірністю, як прогнозований експертний висновок (рисунок 3.4).

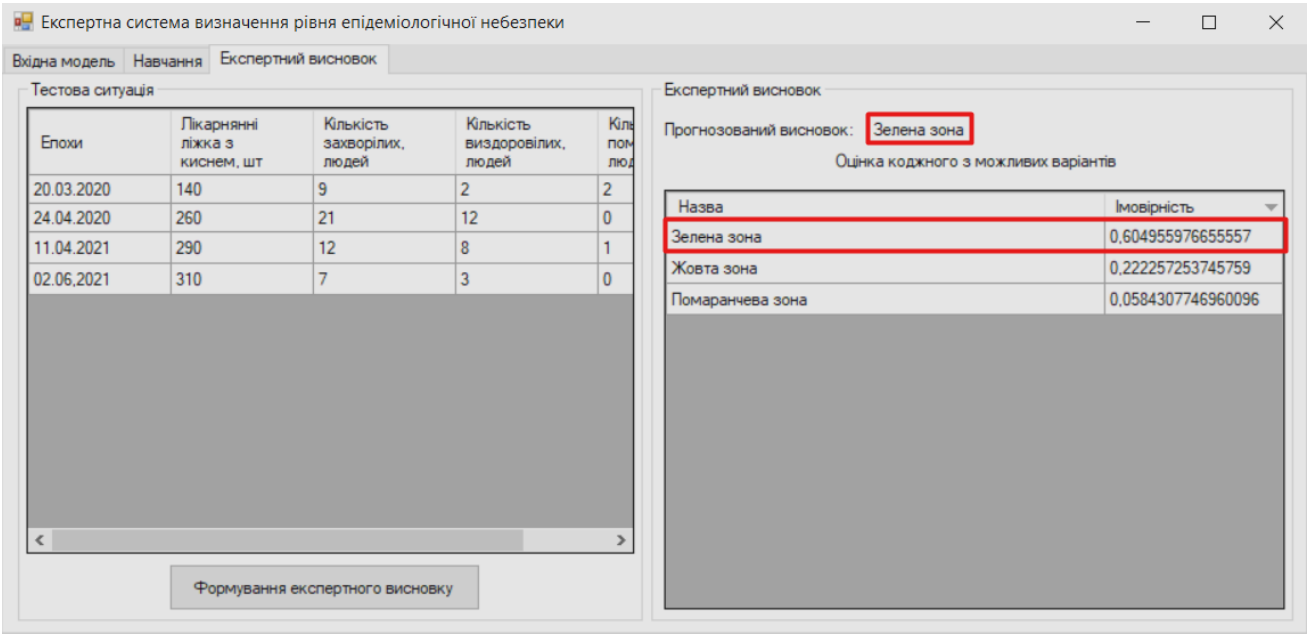


Рисунок 3.4 – Отримання екпертного висновку

Формування експертного висновку на основі значень тестового випадку проводиться наступним кодом:

```
public void Recognition(double[] path_letters)
{
   for (int i = 0; i < Neurons_hiden_layer.Count(); i++)
   {
       Neurons_hiden_layer[i].CorrectionCoefficient(list_result_arr);
       HidenLayerValue[i] = Neurons_hiden_layer[i].Value;
   }
   foreach (var k in context.TrainingSamples)
   {
       Neuron_out_layer[o].CorrectionCoefficient(HidenLayerValue);
       if (Neuron_out_layer[o].Value > res)
       {
           res = Neuron_out_layer[o].Value;
       }
    }
}
```

Реалізовані модулі дозволяють згрупувати елементи інтерфейсу програми у логічні блоки, що в свою чергу сприяє зменшенню непорозумінь в процесі роботи з даними розміщеними в модулях.

3.3 Тестування інформаційної системи

Відповідно до поставленого завдання у програмі має бути реалізований функціонал для розширення вхідної моделі та формування на її основі навчальних вибірок. Для його тестування було роблено шість текст-кейсів, за допомогою яких перевіряється додавання епох, епідеміологічних параметрів та експертних висновків, а також додавання, перегляд даних та видалення навчальних вибірок.

Перший тест-кейсі перевіряє можливість розширення вхідної моделі шляхом додавання нової епохи. У результаті проходження тест-кейсу відбудеться додавання нового елементу до списку епох (таблиця 3.1).

Таблиця 3.1 – Тест-кейс TS0001

<table>
<tr><td>Тест-кейс ID: TS0001</td><td>Пріоритет: 1</td><td>Створено: 10.03.2021,Овчарук О.М.</td></tr>
<tr><td colspan="3">Назва: Перевірка можливості додавання епохи
Вхідні дані:
Назва епохи: 07.06.2021</td></tr>
<tr><td colspan="2">Кроки</td><td>Очікуваний результат</td></tr>
<tr><td colspan="2">1. Запустити програму
2. Перейти на вкладку «Вхідна модель»
3. Ввести назву епохи
4. Натиснути кнопку «Додати» розташовану у блоці «Додавання епохи»</td><td>Додавання нової епохи</td></tr>
<tr><td colspan="3">Результат виконання тест-кейсу: пройдено успішно</td></tr>
</table>

Результатом проходження тест-кейсу є розширення вхідної моделі за допомогою додавання нової епохи. Результат проходження тест-кейсу зображено на рисунку 3.5

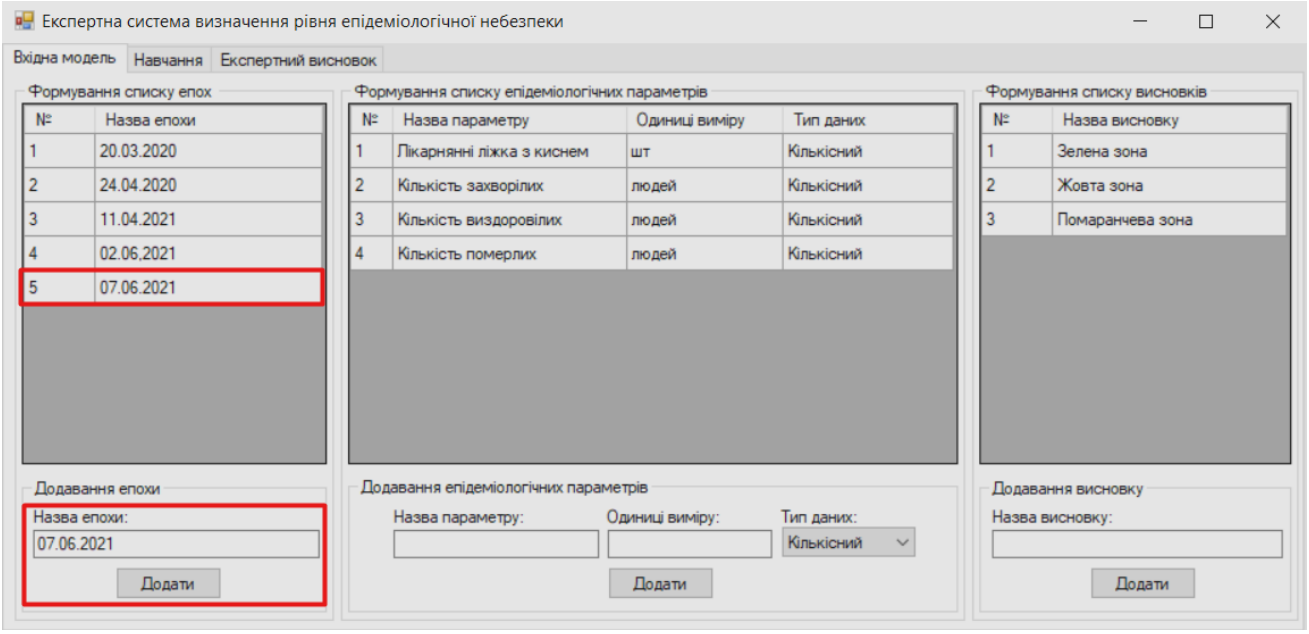


Рисунок 3.5 – Результат додавання епохи

Другий тест-кейс перевіряє можливість розширення вхідної моделі шляхом додавання нового епідеміологічного параметру. У результаті проходження тест-кейсу відбудеться збільшення списку епідеміологічних параметрів (таблиця 3.2).

Таблиця 3.2 – Тест-кейс TS0002

<table>
<tr><td>Тест-кейс ID: TS0002</td><td>Пріоритет: 1</td><td>Створено: 10.03.2021,Овчарук О.М.</td></tr>
<tr><td colspan="3">Назва: Перевірка можливості додавання епідеміологічних параметрів
Вхідні дані:
Назва параметру: Кількість транспорту
Одиниці виміру: шт
Тип даних: кількісний</td></tr>
<tr><td colspan="2">Кроки</td><td>Очікуваний результат</td></tr>
<tr><td colspan="2">1. Запустити програму
2. Перейти на вкладку «Вхідна модель»
3. Ввести вхідні дані
4. Натиснути кнопку «Додати» розташовану у блоці «Додавання епідеміологічних параметрів»</td><td>Додавання нового епідеміологічного параметру</td></tr>
<tr><td colspan="3">Результат виконання тест-кейсу: пройдено успішно</td></tr>
</table>

Після проходження тест-кейсу відбулося додавання нового епідеміологічного параметру. Результат розширення вхідної моделі продемонстровано на рисунку 3.6

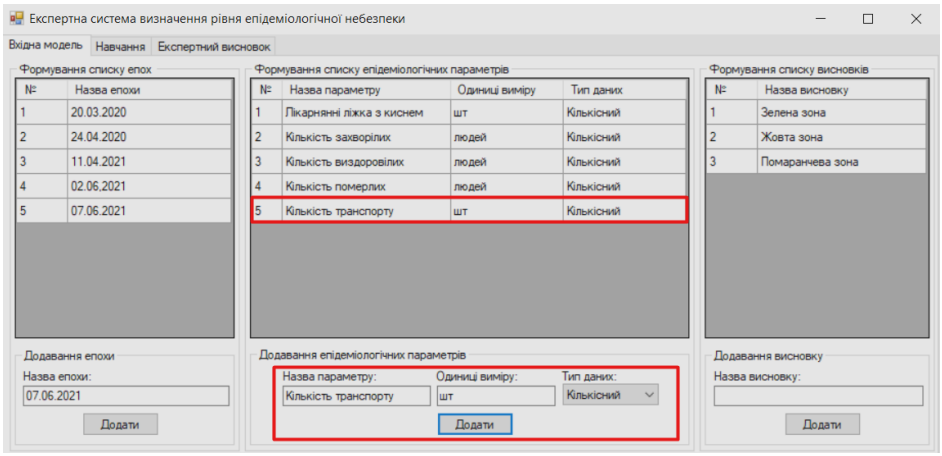


Рисунок 3.6 – Результати додавання нового епідеміологічного параметру

У третьому тест-кейсі тестується можливість додавання нових експертних висновків для експертної системи (таблиця 3.3).

Таблиця 3.3 – Тест-кейс TS0003

Тест-кейс ID: TS0003	**Пріоритет:** 1	**Створено:** 10.03.2021,Овчарук О.М.
Назва: Перевірка можливості додавання експертних висновків **Вхідні дані:** ***Назва висновку:*** Червона зона		
Кроки		**Очікуваний результат**
1. Запустити програму 2. Перейти на вкладку «Вхідна модель» 3. Ввести назву експертного висновку 4. Натиснути кнопку «Додати» розташовану у блоці «Додавання висновку»		Додавання нового експертного висновку
Результат виконання тест-кейсу: пройдено успішно		

Після проходження тест-кейсу відбулося розширення списку експертних висновків, результат його проходження зображено на рисунку 3.7

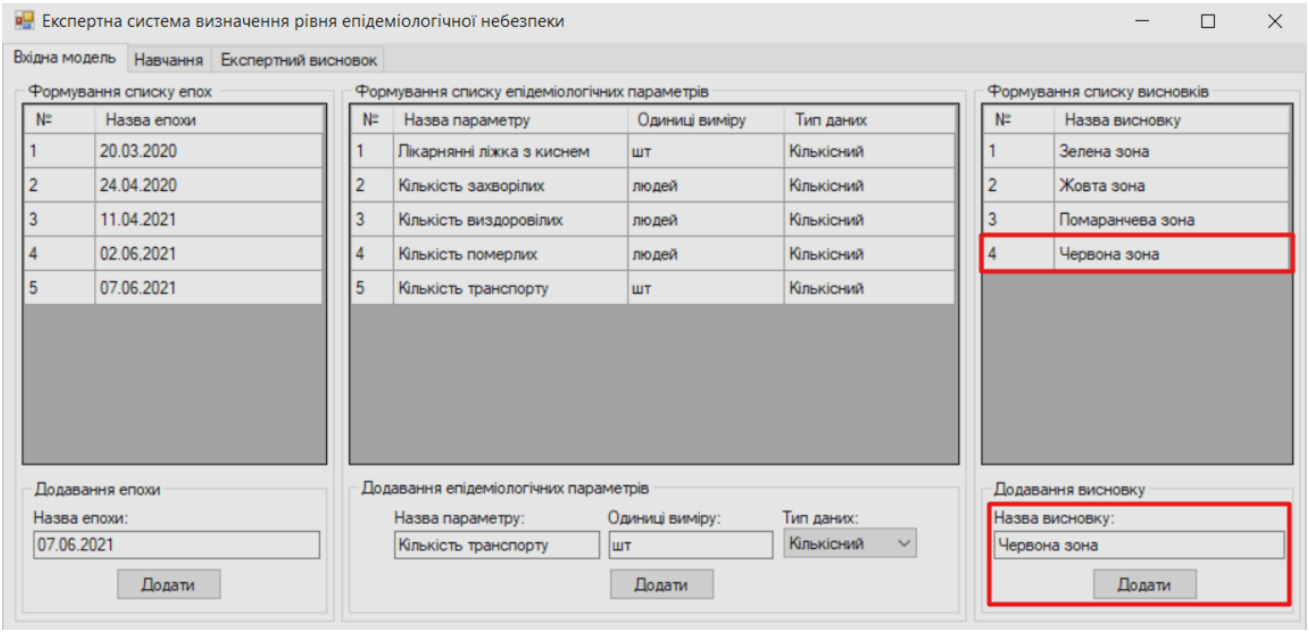


Рисунок 3.7 – Додавання нового висновку

Четвертий тест-кейс перевіряє можливість додавання нової навчальної вибірки (таблиця 3.4). У результаті успішного проходження тест-кейсу відбудеться розширення списку навчальних вибірок.

Таблиця 3.4 – Тест-кейс TS0004

Тест-кейс ID: TS0004	**Пріоритет:** 1	**Створено:** 10.03.2021,Овчарук О.М.
Назва: Перевірка додавання навчальної вибірки **Вхідні дані:** ***Кількість епідеміологічних параметрів:*** 5 ***Кількість епох:*** 5 ***Значення вхідних параметрів:*** довільні ***Назва навчальної вибірки:*** Тест-кейс 1		
Кроки		**Очікуваний результат**
1. Запустити програму 2. Сформувати архітектуру вхідної моделі 3. Перейти на вкладку «Навчання» 4. Ввести початкові дані 5. Натиснути кнопку «+»		Додавання нової навчальної вибірки
Результат виконання тест-кейсу: пройдено успішно		

У результаті проходження тест-кейсу було успішно додано навчальну вибірку. Результати проходження тест-кейсу зображенні на рисунок 3.8

П’ятий тест-кейс призначений для перевірки можливості перегляду даних раніше створеної навчальної вибірки (таблиця 3.5). У результаті успішного проходження тест-кейсу відбудеться завантаження даних вибраної навчальної вибірки.

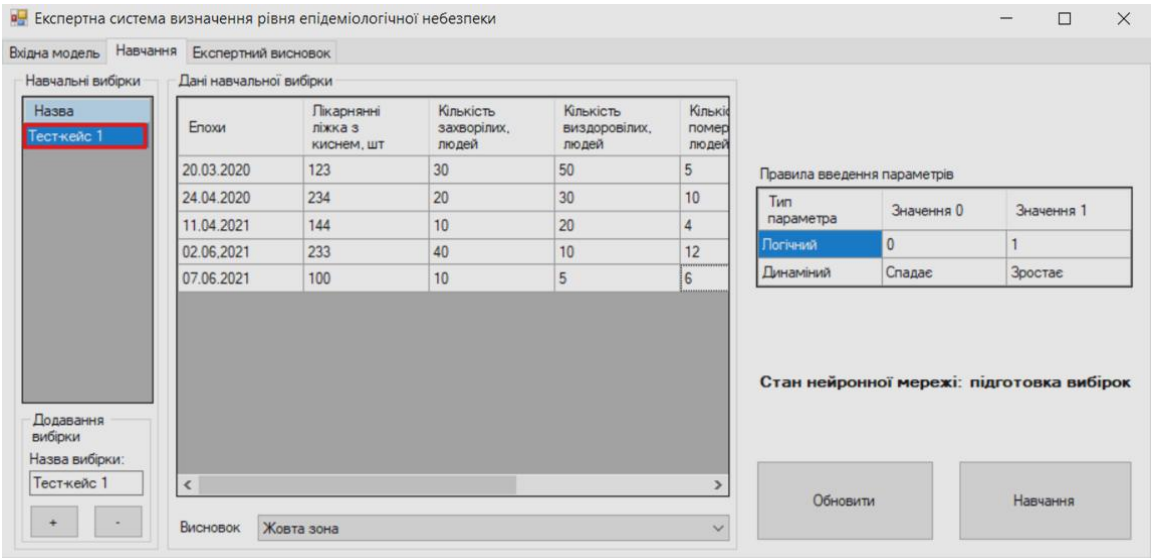


Рисунок 3.8 – Результат додавання навчальної вибірки

Таблиця 3.5 – Тест-кейс TS0005

Тест-кейс ID: TS0005	**Пріоритет:** 2	**Створено:** 10.03.2021,Овчарук О.М.
Назва: Перевірка можливості перегляду даних створеної навчальну вибірку **Вхідні дані:** ***Кількість навчальних вибірок:*** 3 ***Назва навчальної вибірки для якої проводитиметься перегляд:*** Тест-кейс 6		
Кроки	**Очікуваний результат**	
1. Запустити програму 2. Перейти на вкладку «Навчання» 3. Створити та заповнити довільними значеннями навчальні вибірки 4. Двічі натиснути на потрібну навчальну вибірку 5. Переглянути завантаженні дані	Відображення даних вибраної навчальної вибірки	
Результат виконання тест-кейсу: пройдено успішно		

У результаті проходження тест-кейсу було успішно відображено дані вибраної навчальної вибірки. Результат проходження тест-кейсу зображений на рисунку 3.9.

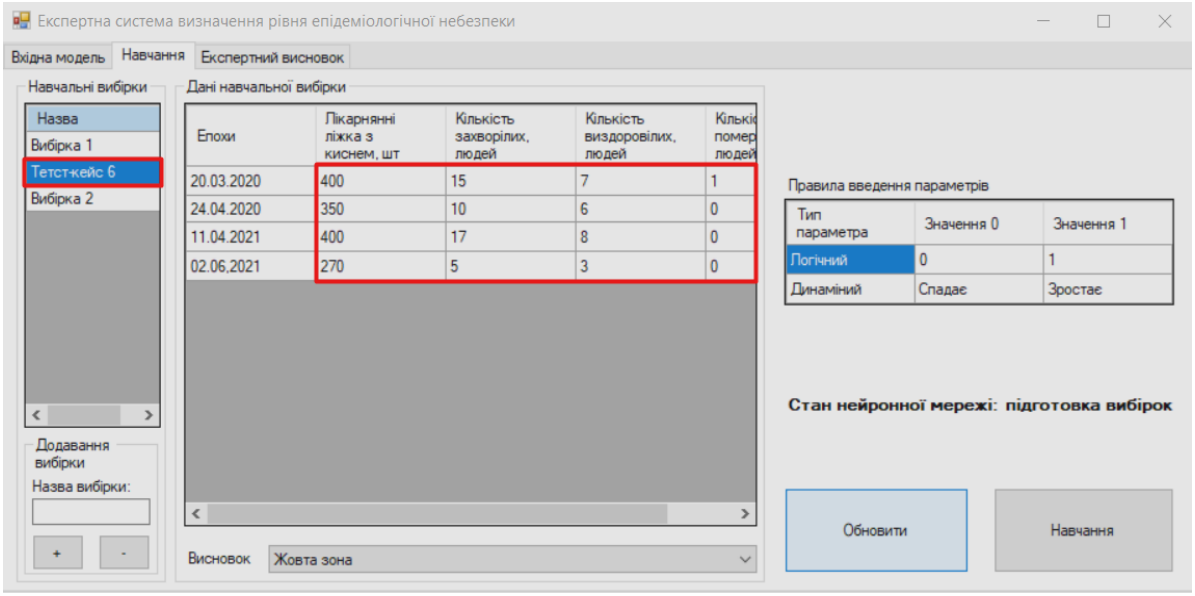


Рисунок 3.9 – Видалення навчальної вибірки

Шостий тест-кейс виконує перевірку видалення навчальної вибірки (таблиця 3.6). У результаті успішного проходження тест-кейсу відбудеться видалення вибірки.

Таблиця 3.6 – Тест-кейс TS0006

Тест-кейс ID: TS0006	**Пріоритет:** 1	**Створено:** 10.03.2021,Овчарук О.М.
Назва: Перевірка видалення навчальної вибірки **Вхідні дані:** ***Назва навчальної вибірки:*** Тест-кейс 1		
Кроки		**Очікуваний результат**
1. Запустити програму 2. Перейти на вкладку «Навчання» 3. Вибрати навчальну вибірку 4. Натиснути кнопку «-»		Видалення навчальної вибірки
Результат виконання тест-кейсу: пройдено успішно		

У результаті проходження тест-кейсу було успішно проведено видалення навчальної вибірки. Результат проходження тест-кейсу зображений на рисунку 3.10.

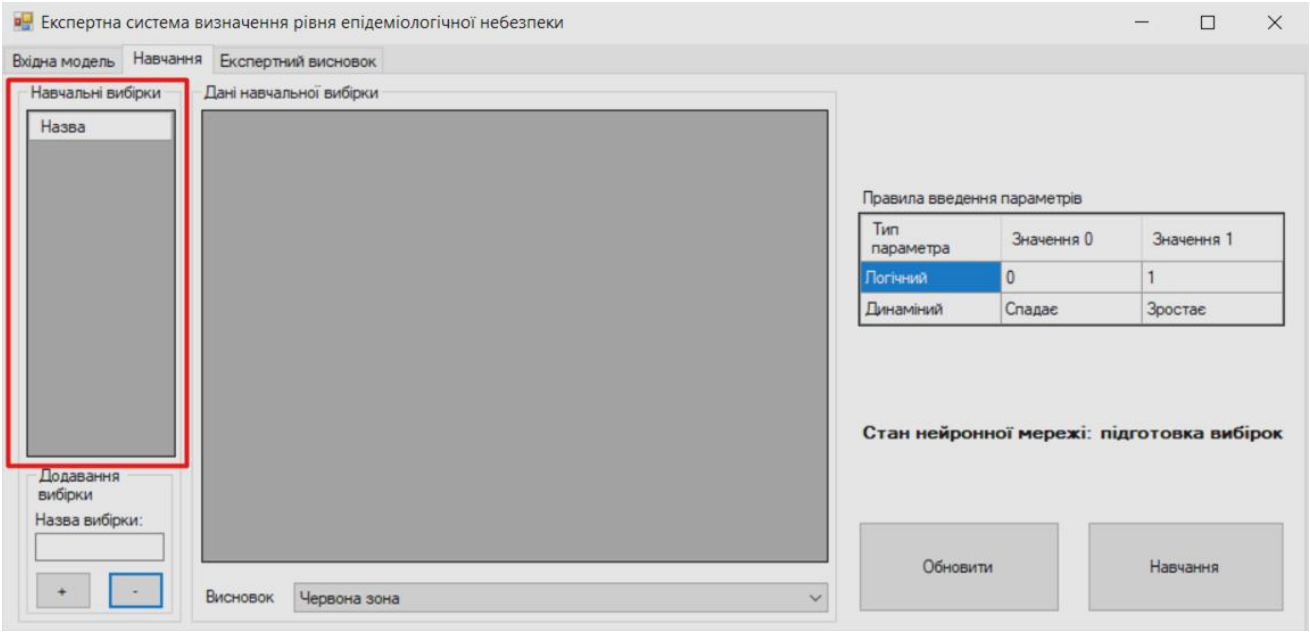


Рисунок 3.10 – Видалення навчальної вибірки

В процесі тестування функціоналу програми призначеного для розширення вхідної моделі та формування навчальних вибірок, помилок та некоректної роботи виявлено не було.

3.4 Інструкція користувача

Після запуску програми користувач має сформувати вхідну модель. На вкладці «Вхідна модель» знаходяться таблиці епох, епідеміологічних параметрів та висновків, які необхідно заповнити. Щоб додати епоху потрібно ввести її назву та натиснути на кнопку «Додати», розташовану в блоці «Додавання епохи». У результаті натиснення на кнопку відбудеться додавання нового запису в базу даних. Якщо процес додавання закінчився вдало користувач буде сповіщений відповідним повідомленням, після чого новий запис відобразиться на формі (рисунок 3.11).

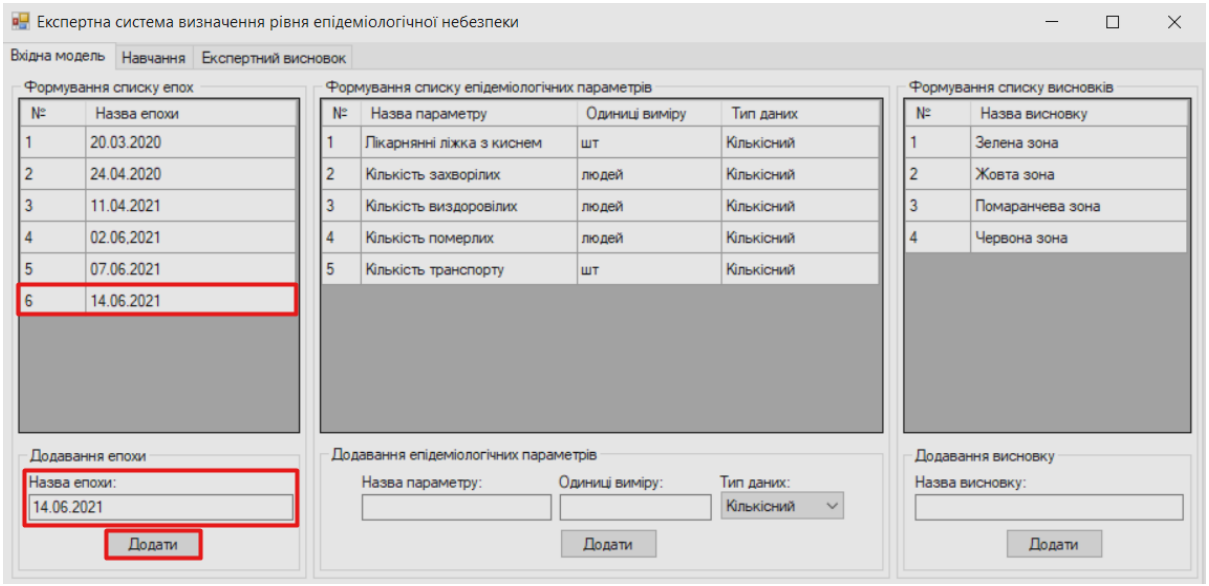


Рисунок 3.11 – Додавання нового запису до списку епох

Щоб додати новий епідеміологічний параметр потрібно ввести його назву, одиниці виміру та вибрати з випадаючого списку тип даних. Коли всі поля заповненні вхідними даними потрібно натиснути на кнопку «Додати» розташовану в блоці «Додавання епідеміологічних параметрів». Після натискання кнопки перед користувачем з’явиться діалогове вікно, яке проінформує його про завершення додавання епідеміологічного параметру (рисунок 3.12).

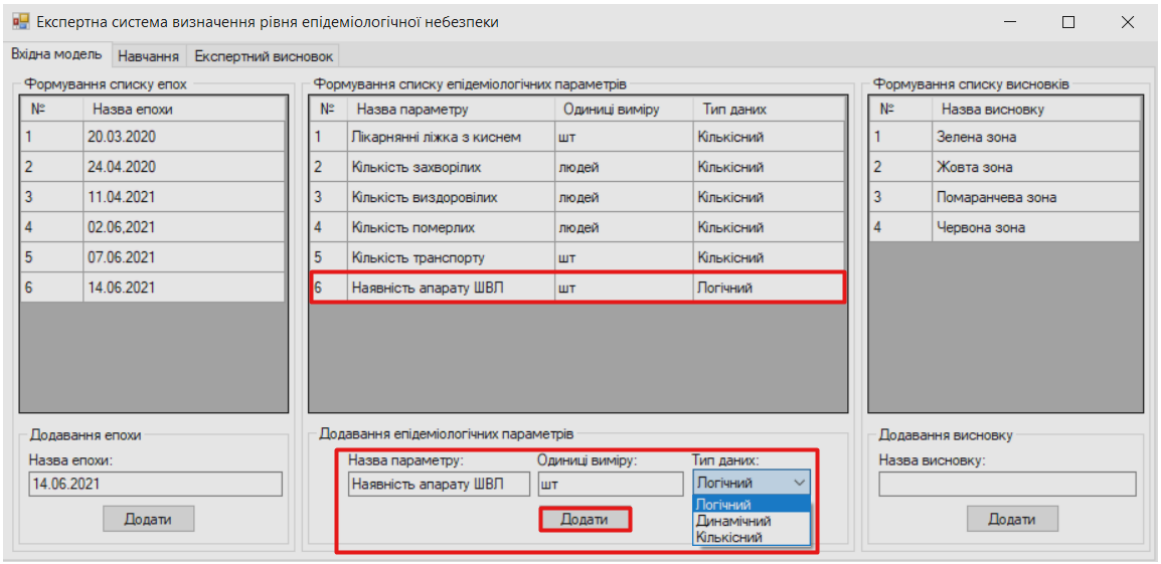


Рисунок 3.12 – Додаванння епідеміологічних параметрів

Окрім цього на вкладці «Вхідна модель» можна переглянути перелік висновків експертної системи та додати до його складу новий елемент, для цього потрібно вказати назву нового висновку та натиснути на кнопку «Додати» розташовану у блоці «Додавання висновку». Після чого до списку висновків потрапить запис з вказаною раніше назвою (рисунок 3.13).

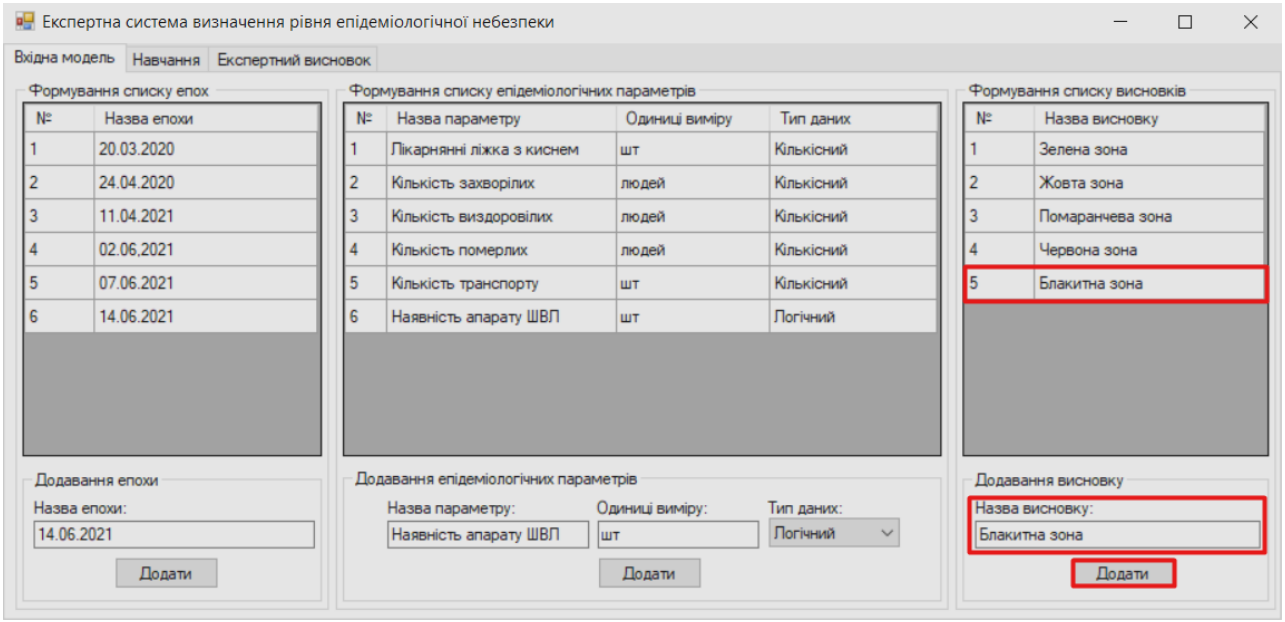


Рисунок 3.13 – Розширення списку результатів

Коли вхідна модель набуде бажаного вигляду користувач має перейти на вкладку «Навчання». Для оновлення даних про стан моделі потрібно натиснути кнопку «Оновити» у результаті чого перед користувачем відобразиться таблиця з вказаними раніше епідеміологічними параметрами та епохами вхідної моделі, а також розшириться випадаючий список з експертними висновками (рисунок 3.14).

Після оновлення вхідної моделі користувач має сформувати навчальні вибірки. Для цього потрібно у відповідних полях таблиці «Дані навчальної вибірки» вказати їх числові значення. Далі користувач має вибрати експертний висновок з випадаючого списку та ввести назву навчальної вибірки. Коли введення даних буде закінчено користувачеві потрібно буде натиснути на кнопку

«+». У результаті проведених дій до списку навчальних вибірок додається новий елемент (рисунок 3.15).

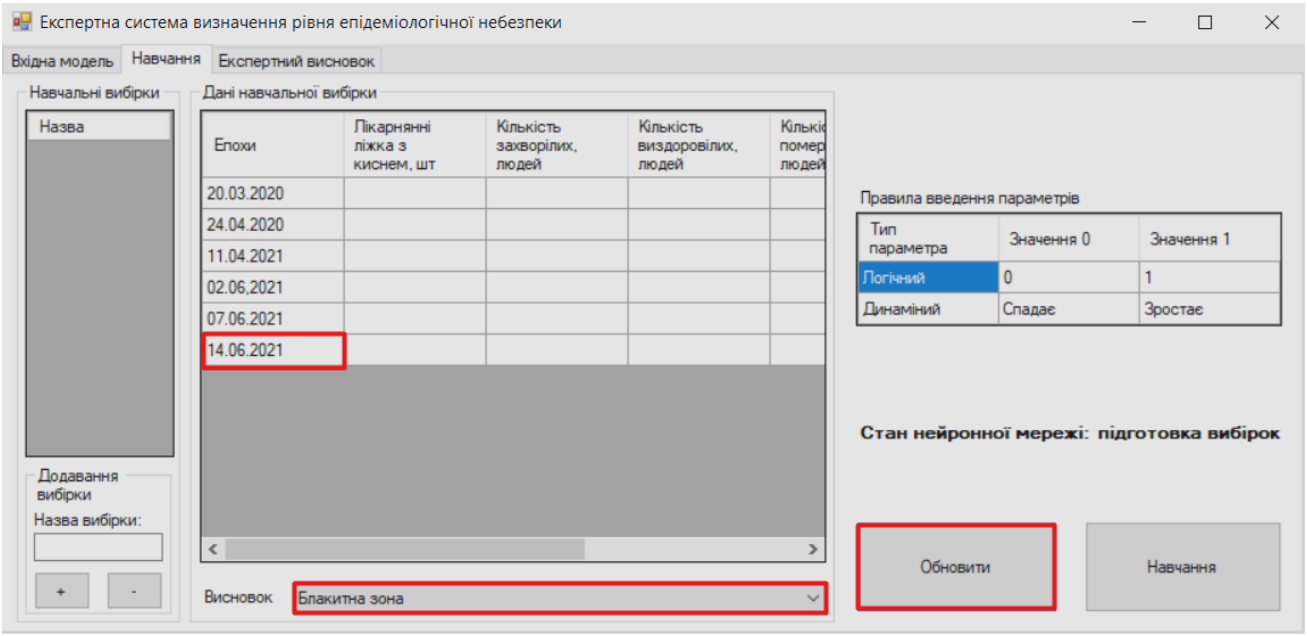


Рисунок 3.14 – Оновлення даних моделі

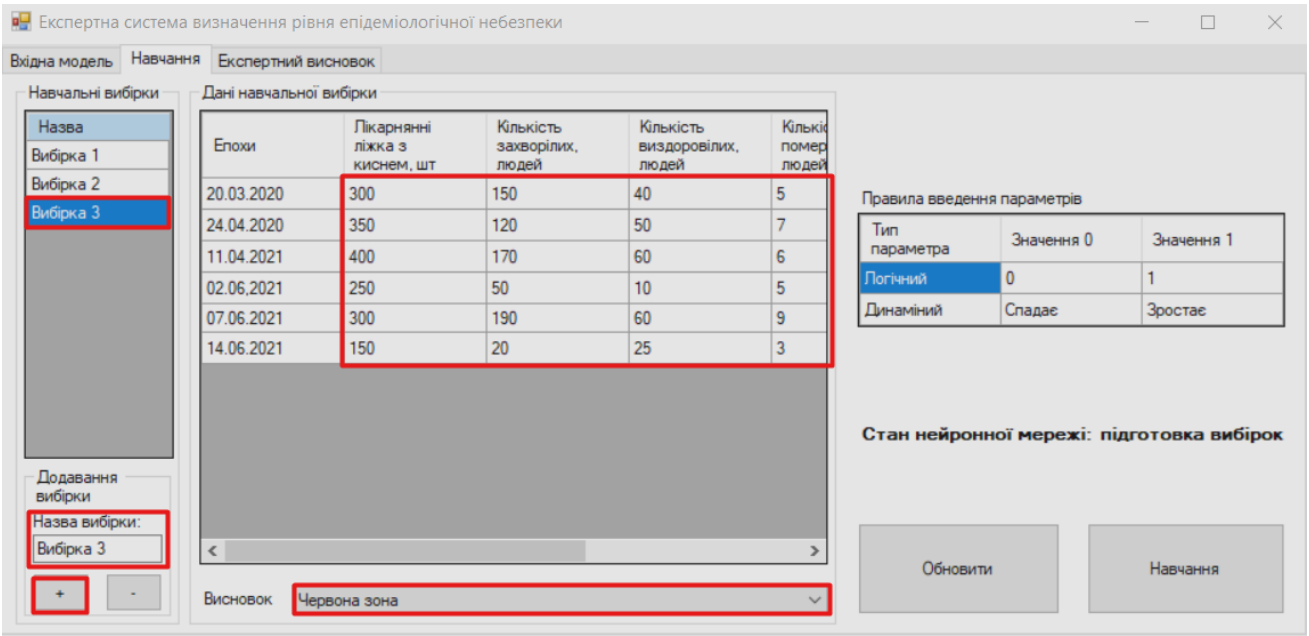


Рисунок 3.15 – Формування навчальних вибірок

Якщо користувач захоче переглянути дані раніше доданої вибірки то йому потрібно буде двічі натиснути на її назву в списку «Навчальні вибірки». У

результаті чого в таблицю «Дані навчальної вибірки» завантажаться значення параметрів вибраної вибірки (рисунок 3.16).

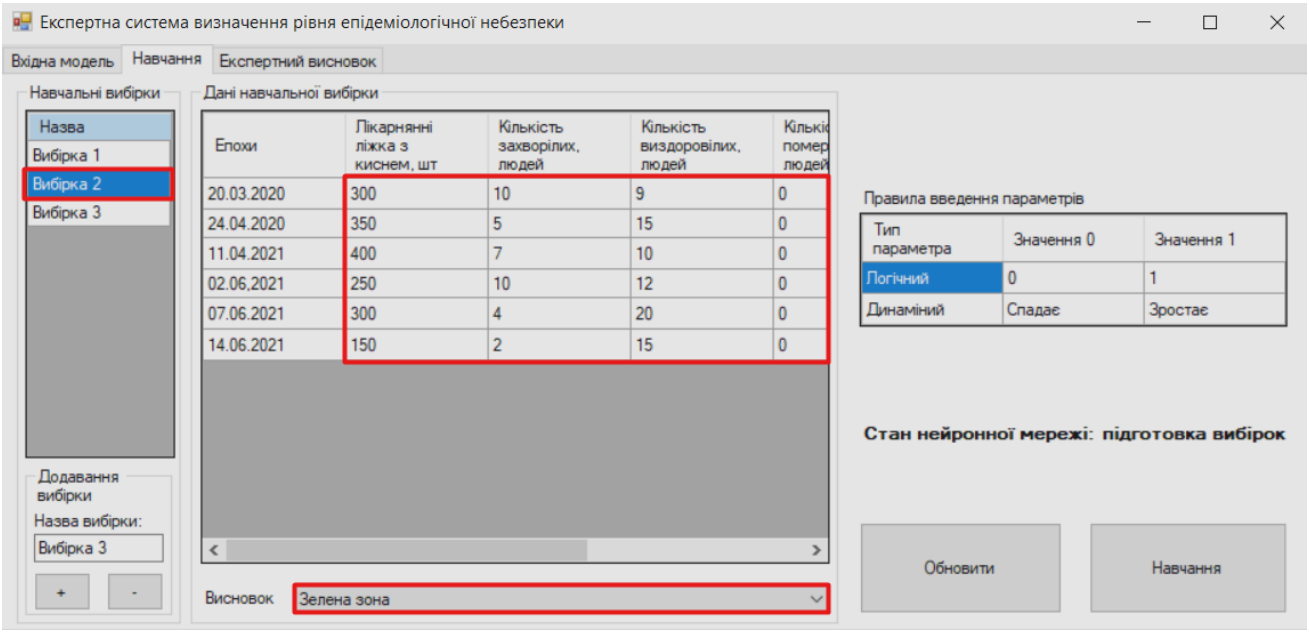


Рисунок 3.16 – Перегляд даних вибраної навчальної вибірки

Окрім цього, якщо користувач додав не потрібну навчальну вибірку, він може її видали. Для цього йому потрібно вибрати її назву з відповідного переліку та натиснути на кнопку «-». У результаті чого відбудеться видалення навчальної вибірки та всіх її даних.

Коли користувач закінчить формування навчальних вибірок йому потрібно натиснути на кнопку «Навчання», у результаті чого розпочнеться процес навчання нейронної мережі, після закінчення якого користувач отримає відповідне сповіщення (рисунок 3.17).

Далі користувачеві потрібно перейти на вкладку «Експертний висновок». На завантаженій вкладці необхідно ввести дані тестової ситуації та натиснути кнопку «Формування експертного висновку». У результаті перед користувачем з’явиться прогнозований експертний висновок та виведуться назви й імовірності виникнення інших висновків (рисунок 3.18).

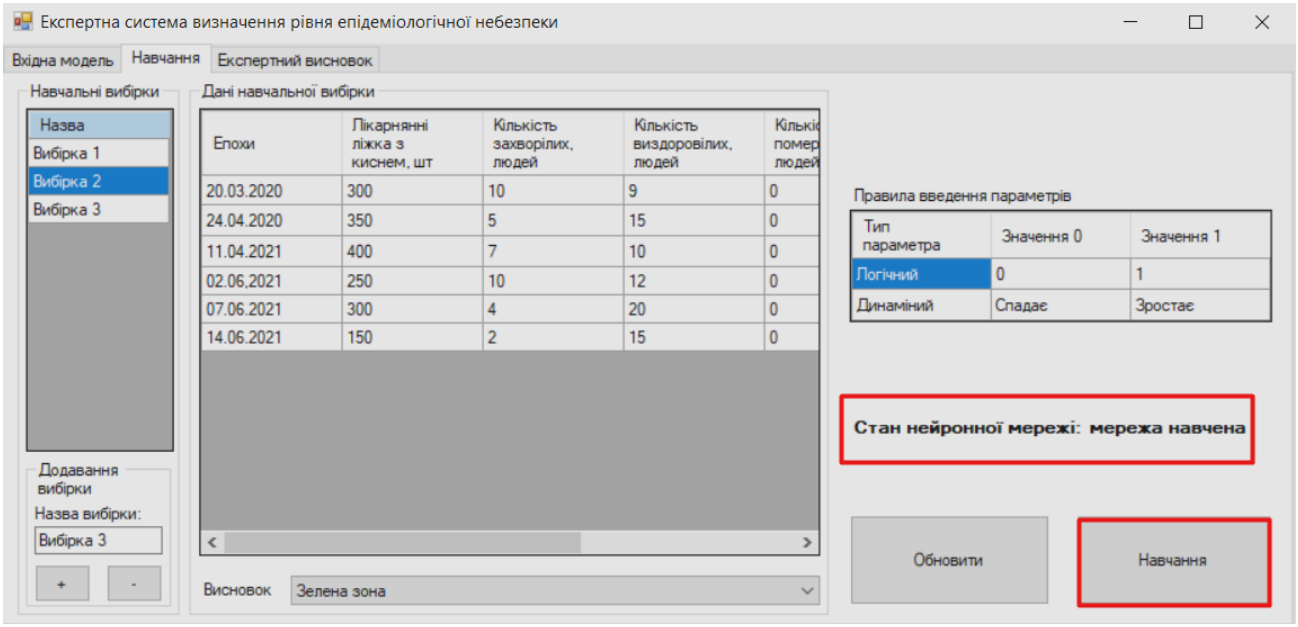


Рисунок 3.17 – Навчання нейронної мережі

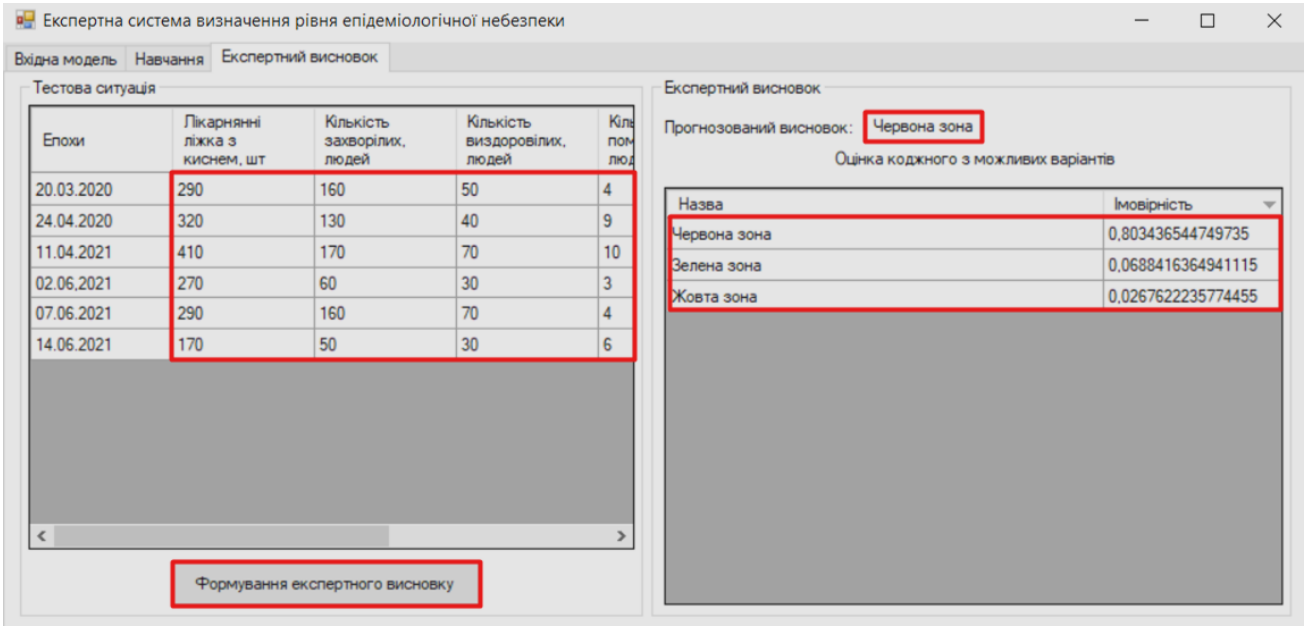


Рисунок 3.18 – Заповнення даних тестової ситуації

Розглядаючи розроблений програмний продукт потрібно відмітити вдалу реалізацію його інтерфейсу, що наглядно розділяє елементи експертної системи на зрозумілі користувачеві групи, розміщенні у відповідних закладках.

3.5 Вимоги до розгортання інформаційної системи

Мінімальні вимоги до апаратних засобів:

- Процесор: x86, x32, 2000GH;
- Оперативна пам’ять: 512 MB;
- Дисплей: 800x600;
- Жорсткий диск: HDD (128GB).

Мінімальні вимоги до програмних засобів

- ОС: Windows 10;
- Фреймворк: .NET Framework 4.7.2;
- СКБД: Microsoft SQL Server 2017.

Висновки

В результаті виконання кваліфікаційної роботи бакалавра було розроблено експертну систему визначення рівня епідеміологічної небезпеки на платформі .NET із використанням нейромережевих технологій. Даний програмний продукт виконує наступні функції:

– параметризована побудова вхідних моделей за розмірністю кількості параметрів епохи та кількості епох;

– параметризована побудова вихідних моделей за розмірністю кількості можливих висновків експертної системи;

– генерація архітектури нейронної мережі згідно параметрів вхідних та вихідних моделей;

– одержання від користувача атрибутів для кожного з параметрів вхідної моделі (назви параметрів, одиниці виміру параметрів, типи даних параметрів) та вихідної моделі (назви епох);

– використання бази даних для збереження та відтворення параметрів роботи експертної системи у вигляді параметрів вхідної моделі, параметрів вихідної моделі та їх атрибутів;

– одержання від користувача значень параметрів вхідної та вихідної моделі (значень назви параметрів, одиниць виміру параметрів, типів даних параметрів та назв епох);

– одержання від користувача значень параметрів тестового випадку (значень назви параметрів, одиниць виміру параметрів, типів даних параметрів та назв епох);

– формування експертного висновку нейромережею, який містить прогнозований варіант для тестового випадку та оцінки кожного з можливих варіантів.

Реалізація даних функцій експертної системи визначення рівня епідеміологічної небезпеки призначена для визначення рівня епідеміологічної ситуації в регіоні в умовах адаптивного карантину.

Для розробки інформаційної системи на платформі .NET було використано мову програмування C#, а також систему керування базами даних MS SQL Server.

Розроблена експертна система визначення рівня епідеміологічної небезпеки призначена для визначення рівня епідеміологічної ситуації у локальних межах в умовах адаптивного карантину із використанням нейромережевих технологій. Експертна система визначення рівня епідеміологічної небезпеки дозволяє створювати навчальні моделі з різною кількістю вхідних параметрів та епох, при створенні вхідних параметрів надаючи можливість вказувати їхні одиниці виміру та тип, й здійснювати формування експертного висновку з рекомендацією найбільш вірогідного результату.

Запропонований у роботі підхід, втілений у вигляді розробленої експертної системи визначення рівня епідеміологічної небезпеки, дозволяє врахувати те, що для визначення рівня епідеміологічної небезпеки слід враховувати не тільки поточний стан справ, а й ситуацію, що йому передувала, а також є доцільним при прийнятті рішень керуватись даними розвитку подій за аналогічних ситуацій.

При роботі над кваліфікаційною роботою бакалавра автором виконано доповіді на XI Всеукраїнській науково-практичній конференції «Актуальні проблеми комп’ютерних наук АПКН-2019» та XII Всеукраїнській науково-практичній конференції «Актуальні проблеми комп’ютерних наук АПКН-2020», а також виконано 2 наукових публікації [33, 34].

Перелік посилань

1. Коронавірусна хвороба 2019. URL: https://vh.dnmu.ru/index.php/VHE-2-2018/article/download/318/327Інфекційні захворювання: Підручник / В.И. Покровский, С.Г. Пак, Н.И. Брико, Б.К. Данилкин. – М.: ГЭОТАР-Медіа, 2007. – 816 с.

2. Що таке епідемія. URL: http://sepi.multycourse.com.ua/ua/page/15/50

3. Як виникають і поширюються інфекційні захворювання. URL: http://sepi.multycourse.com.ua/ua/page/15/53

4. Інфекційні хвороби. Навчальний посібник / О.В. Рябоконь, Т.Є. Оніщенко, Ю.Ю. Рябоконь – Запоріжжя, 2015. – 11 с.

5. Карантинні заходи. URL: https://covid19.gov.ua/karantynni-zakhody

6. Система моніторингу поширення епідемії коронавірусу. URL: https://covid19.rnbo.gov.ua/

7. Система охорони здоров'я України. URL: https://health-security.rnbo.gov.ua/

8. Центра системних наук та інженерії університету Джонса Хопкінса. URL: https://www.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6

9. Myers M. F., Rogers D. J., Cox J., Flahault A., Hay S. I. Forecasting Disease Risk for Increased Epidemic Preparedness in Public Health // Advances in Parasitology. — 2000. — Vol. 47. — P. 309–330.

10. Soebiyanto R. P., Adimi F., Kiang R. K. Modeling and Predicting Seasonal Influenza Transmission in Warm Regions Using Climatological Parameters // PLoS ONE. — 2010. — Vol. 5, № 3

11. Bai Y., Jin Z. Prediction of SARS epidemic by BP neural networks with online prediction strategy // Chaos, Solitons and Fractals. — 2005. — Vol. 26, № 2. — P. 559–569.

12. Початок роботи з .NET Framework. URL: https://docs.microsoft.com/ru-ru/dotnet/framework/get-started/

13. NET Framework. URL: https://ru.wikipedia.org/wiki/.NET_Framework

14. Загальні відомості про платформу .NET Framework. URL: https://docs.microsoft.com/ru-ru/dotnet/framework/get-started/overview

15. Переваги та недоліки розробки на платформі .NET Framework. URL: https://www.cisin.com/coffee-break/ru/enterprise/the-good-and-the-bad-of-net-framework-development.html

16. Java як мова програмування та платформа. URL: https://comaqa.gitbook.io/java-automation/java.-vvedenie/java-kak-yazyk-i-platforma

17. Що таке JVM. URL: https://topjava.ru/blog/what-is-the-jvm

18. Платформа Java особливості та перваги. URL: https://cutt.ly/mnEnQ3l

19. Короткий огляд мови C#. URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/

20. Мова програмування C# та платформа. URL: https://metanit.com/sharp/tutorial/1.1.php

21. Об'єктно-орієнтований підхід до програмування. URL: http://www.znannya.org/?view=csharp-oop

22. 9 причин вивчити C#. URL: https://foxminded.com.ua/news/9-prychyn-yzuchyt-yazyk-c/

23. SQL Server. URL: https://uk.wikipedia.org/wiki/Microsoft_SQL_Server

24. Реляційні СКБД – порівняння MySQL та SQL-сервер. URL: https://www.hostinger.com.ua/rukovodstva/reljacionnye-subd-sravnenie-mysql-i-sql-server/

25. Що таке MySQL. URL: https://mchost.ru/articles/chto-takoe-mysql/

26. Основні переваги СКБД MySQL. URL: https://studfile.net/preview/5607354/page:4/

27. Початок роботи з WPF. URL: https://docs.microsoft.com/ru-ru/visualstudio/designers/getting-started-with-wpf?view=vs-2019

28. Особливості платформи WPF. URL: https://metanit.com/sharp/wpf/1.php

29. Складності в роботі з WPF. URL: https://habr.com/ru/post/262299/#1

30. Windows Forms: Сучасна модель програмування для створення GUI додатків: http://www.codenet.ru/progr/cpp/WinForms.php

31. WPF vs Windows Forms. URL: https://cutt.ly/gnEr3mS

32. Відмінності між WPF та Windows Forms. URL: https://www.educba.com/winforms-vs-wpf/

33. Овчарук О. М., Мазурець О. В. Математична модель фасеткового дорозпізнавального перетворення зображень. Збірник наукових праць за матеріалами XI всеукраїнської науково-практичної конференції «Актуальні проблеми комп'ютерних наук АПКН-2019» – Хмельницький, 2019, Т.1. – С.151-152.

34. Овчарук О. М., Мазурець О. В. Метод фасеткового дорозпізнавального перетворення зображень для нейромережевого розпізнавання. Збірник наукових праць за матеріалами XII Всеукраїнської науково-практичної конференції «Актуальні проблеми комп'ютерних наук АПКН-2020» – Хмельницький, 2020. – С.203-208.

ДОДАТКИ

Додаток А

Програмні коди основних модулів експертної системи визначення рівня епідеміологічної небезпеки

NeuralNetwork.cs

```
using ExpertSystemCOVID.Models;
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ExpertSystemCOVID
{
    class NeuralNetwork
    {
        COVIDEntities1 context = new COVIDEntities1();
        public static double LearningRate = 5;
        List<Neuron> Neuron_out_layer;
        List<Neuron> Neurons_hiden_layer;
        public List<double[]> InputList;
        double[] HidenLayerValue;
        public NeuralNetwork(List<double[]> modelList)
        {
            Neurons_hiden_layer = new List<Neuron>();
            InputList = modelList;
            HidenLayerValue = new double[(int)Math.Ceiling((double)((InputList[0].Count() +
InputList.Count()) / 2))];
            for (int i = 0; i < HidenLayerValue.Count(); i++)
            {
                Neurons_hiden_layer.Add(new Neuron(InputList[0]));
                HidenLayerValue[i] = Neurons_hiden_layer[i].Value;
            }
            Neuron_out_layer = new List<Neuron>();
            for (int i = 0; i < InputList.Count(); i++)
            {
                Neuron_out_layer.Add(new Neuron(HidenLayerValue));
            }
        }


        bool Checking(List<double[]> InputList)
        {
            int counter = 0;
            for (int k = 0; k < Neuron_out_layer.Count(); k++)
            {
                for (int i = 0; i < Neurons_hiden_layer.Count(); i++)
                {
                    Neurons_hiden_layer[i].CorrectionCoefficient(InputList[k]);
                    HidenLayerValue[i] = Neurons_hiden_layer[i].Value;
                }

                for (int i = 0; i < Neuron_out_layer.Count(); i++)
                    Neuron_out_layer[i].CorrectionCoefficient(HidenLayerValue);

                if (Neuron_out_layer[k].Value >= 0.9)
```

```
                counter++;
            }
            if (counter == Neuron_out_layer.Count())
                return false;
            else
                return true;
        }

        double SumError(int counter, double[] error_out)
        {
            double count = 0;
            for (int i = 0; i < Neuron_out_layer.Count(); i++)
            {
                count += Neuron_out_layer[i].WeightArray[counter] * error_out[i];
            }

            return count;
        }

        public void Study()
        {
            double[] error_out = new double[Neuron_out_layer.Count()];
            double[] error_hiden = new double[Neurons_hiden_layer.Count()];
            int k = 0;
            while (Checking(InputList))
            {
                if (k != 0)
                {
                    Neuron_out_layer[k].State = 1;
                    Neuron_out_layer[k - 1].State = 0;
                }
                else if (Neuron_out_layer.Count() == 1)
                    Neuron_out_layer[k].State = 1;
                else
                {
                    Neuron_out_layer[k].State = 1;
                    Neuron_out_layer[Neuron_out_layer.Count() - 1].State = 0;
                }
                Parallel.For(0, Neurons_hiden_layer.Count(), i =>
                {
                    Neurons_hiden_layer[i].CorrectionCoefficient(InputList[k]);
                    HidenLayerValue[i] = Neurons_hiden_layer[i].Value;
                });
                Parallel.For(0, Neuron_out_layer.Count(), i =>
                {
                    Neuron_out_layer[i].CorrectionCoefficient(HidenLayerValue);
                });

                Parallel.For(0, Neuron_out_layer.Count(), i =>
                {
                    error_out[i] = (Neuron_out_layer[i].State - Neuron_out_layer[i].Value)
                            * Neuron_out_layer[i].Value * (1 - Neuron_out_layer[i].Value);
                });

                Parallel.For(0, error_hiden.Count(), i =>
                {
                    error_hiden[i] = SumError(i, error_out) *
                             Neurons_hiden_layer[i].Value * (1 -
Neurons_hiden_layer[i].Value);
                });

                Parallel.For(0, Neuron_out_layer.Count(), i =>
                {
                    for (int j = 0; j < Neuron_out_layer[0].WeightArray.Count(); j++)
                    {
```

```
                        Neuron_out_layer[i].WeightArray[j] =
Neuron_out_layer[i].WeightArray[j] +
                            LearningRate * error_out[i] * Neurons_hiden_layer[j].Value;

                    }
                });
                Parallel.For(0, Neurons_hiden_layer.Count(), i =>
                {
                    for (int j = 0; j < Neurons_hiden_layer[0].WeightArray.Count(); j++)
                    {
                        Neurons_hiden_layer[i].WeightArray[j] =
Neurons_hiden_layer[i].WeightArray[j] +
                            LearningRate * error_hiden[i] * InputList[k][j];
                    }
                });

                if (k < Neuron_out_layer.Count() - 1)
                {
                    k++;
                }
                else
                    k = 0;

            }

        }

        public void Recognition( DataGridView dataGridView, double[] path_letters, Label
label)
        {

            double[] list_result_arr = path_letters;
            dataGridView.Rows.Clear();
            for (int i = 0; i < Neurons_hiden_layer.Count(); i++)
            {
                Neurons_hiden_layer[i].CorrectionCoefficient(list_result_arr);
                HidenLayerValue[i] = Neurons_hiden_layer[i].Value;
            }
            int o = 0;
            double res = 0;
            foreach (var k in context.TrainingSamples)
            {
                Neuron_out_layer[o].CorrectionCoefficient(HidenLayerValue);
                dataGridView.Rows.Add( k.Results.Name, Neuron_out_layer[o].Value);
                if (Neuron_out_layer[o].Value > res)
                {
                    res = Neuron_out_layer[o].Value;
                    label.Text = k.Results.Name;
                }
                o++;
            }
            dataGridView.DefaultCellStyle.SelectionBackColor =
dataGridView.DefaultCellStyle.BackColor;
            dataGridView.DefaultCellStyle.SelectionForeColor =
dataGridView.DefaultCellStyle.ForeColor;

        }
    }
}
```

Form1.cs

```
using ExpertSystemCOVID.Models;
using System;
using System.Collections.Generic;
using System.ComponentModel;
```

```
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ExpertSystemCOVID
{
    public partial class Form1 : Form
    {
        COVIDEntities1 context = new COVIDEntities1();
        NeuralNetwork Learn;

        public Form1()
        {
            InitializeComponent();

            dataGridView4.SelectionMode = DataGridViewSelectionMode.FullRowSelect;
            dataGridView4.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.AllCells;

            dataGridView1.DefaultCellStyle.SelectionBackColor =
dataGridView1.DefaultCellStyle.BackColor;
            dataGridView1.DefaultCellStyle.SelectionForeColor =
dataGridView1.DefaultCellStyle.ForeColor;

            dataGridView2.DefaultCellStyle.SelectionBackColor =
dataGridView1.DefaultCellStyle.BackColor;
            dataGridView2.DefaultCellStyle.SelectionForeColor =
dataGridView1.DefaultCellStyle.ForeColor;

            dataGridView3.DefaultCellStyle.SelectionBackColor =
dataGridView1.DefaultCellStyle.BackColor;
            dataGridView3.DefaultCellStyle.SelectionForeColor =
dataGridView1.DefaultCellStyle.ForeColor;

            var DataType = context.DataType;
            foreach (var i in DataType)
                comboBox1.Items.Add(i.Name);
            comboBox1.SelectedIndex = comboBox1.Items.Count - 1;

            var ComboResult = context.Results;
            foreach (var i in ComboResult)
                comboBox2.Items.Add(i.Name);
            comboBox2.SelectedIndex = comboBox2.Items.Count - 1;

            var epoch = context.Epoch;
            foreach (var i in epoch)
                dataGridView1.Rows.Add(i.ID, i.Name);

            var parameters = context.Parameters;
            foreach (var i in parameters)
                dataGridView2.Rows.Add(dataGridView2.Rows.Count + 1, i.Name,
i.UnitsMeasurement, i.DataType.Name);

            var results = context.Results;
            foreach (var i in results)
                dataGridView3.Rows.Add(dataGridView3.Rows.Count + 1, i.Name);

            var trainingSamples = context.TrainingSamples;
            foreach (var i in trainingSamples)
                dataGridView4.Rows.Add(i.ID ,i.Name);


            string[] row15 = new string[] { "Логічний", "0", "1"};
```

```
            string[] row25 = new string[] { "Динаміний", "Спадає", "Зростає" };

            object[] rows5 = new object[] { row15, row25 };

            foreach (string[] rowArray in rows5)
            {
                dataGridView6.Rows.Add(rowArray);
            }

        }

        private void dataGridView6_CellContentClick(object sender, DataGridViewCellEventArgs
e)
        {

        }

        private void button1_Click(object sender, EventArgs e)
        {
            try
            {
                context.Epoch.Add(new Epoch
                {
                    Name = textBox1.Text,
                });
                context.SaveChanges();
                MessageBox.Show("Об'єкт успішно збережено");

                //Refresh
                dataGridView1.Rows.Clear();
                var epoch = context.Epoch;
                foreach (var i in epoch)
                    dataGridView1.Rows.Add(i.ID, i.Name);

            }
            catch
            {
                MessageBox.Show("Заповність всі поля");
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {

            try
            {
                context.Parameters.Add(new Parameters
                {
                    Name = textBox2.Text,
                    UnitsMeasurement = textBox3.Text,
                    FK_DataType = context.DataType.Where(u => u.Name ==
comboBox1.SelectedItem.ToString()).FirstOrDefault().ID,
                });
                context.SaveChanges();
                MessageBox.Show("Об'єкт успішно збережено");
                //Refresh
                dataGridView2.Rows.Clear();
                var parameters = context.Parameters;
                foreach (var i in parameters)
                    dataGridView2.Rows.Add(dataGridView2.Rows.Count + 1, i.Name,
i.UnitsMeasurement, i.DataType.Name);
            }
            catch
            {
                MessageBox.Show("Заповність всі поля");
            }
```

```
        }

        private void button3_Click(object sender, EventArgs e)
        {

            try
            {
                context.Results.Add(new Results
                {
                    Name = textBox4.Text,
                });
                context.SaveChanges();
                MessageBox.Show("Об'єкт успішно збережено");

                //Refresh
                dataGridView3.Rows.Clear();
                var results = context.Results;
                foreach (var i in results)
                    dataGridView3.Rows.Add(dataGridView3.Rows.Count+1, i.Name);

                comboBox2.Items.Add(textBox4.Text);
            }
            catch
            {
                MessageBox.Show("Заповність всі поля");
            }
        }
        Dictionary<int, DataGridView> sample = new Dictionary<int, DataGridView>();



        private void button4_Click(object sender, EventArgs e)
        {


            try
            {
                context.TrainingSamples.Add(new TrainingSamples
                {
                    Name = textBox5.Text,
                    FK_Result = context.Results.Where(u => u.Name ==
comboBox2.SelectedItem.ToString()).FirstOrDefault().ID
                });
                context.SaveChanges();

                for (int i = 0; i < dataGridView5.Rows.Count; i++)
                {
                    for (int j = 1; j < dataGridView5.Columns.Count; j++)
                    {
                        string str1 = dataGridView5.Columns[j].HeaderText.Substring(0,
dataGridView5.Columns[j].HeaderText.IndexOf(','));
                        string str2 = dataGridView5[0, i].Value.ToString();
                        context.ParameterValues.Add(new ParameterValues
                        {
                            Value = Convert.ToDouble(dataGridView5[j, i].Value),
                            FK_Parameter = context.Parameters.Where(u => u.Name ==
str1).FirstOrDefault().ID,
                            FK_Epoch = context.Epoch.Where(u => u.Name ==
str2).FirstOrDefault().ID,
                            FK_TrainingSamples = context.TrainingSamples.Where(u => u.Name ==
textBox5.Text).FirstOrDefault().ID
                        });
                        context.SaveChanges();
                    }
                }
```

```
                MessageBox.Show("Об'єкт успішно збережено");

                //Refresh
                dataGridView4.Rows.Clear();
                var trainingSamples = context.TrainingSamples;
                foreach (var i in trainingSamples)
                    dataGridView4.Rows.Add(i.ID, i.Name);

            }
            catch
            {
                MessageBox.Show("Заповність всі поля");
            }

        }

        private void button5_Click(object sender, EventArgs e)
        {
            int ID = Convert.ToInt32(dataGridView4.CurrentCell.Value);

            var delete = context.ParameterValues.Where(u => u.FK_TrainingSamples == ID);
            foreach(var i in delete)
            context.ParameterValues.Remove(i);

            context.TrainingSamples.Remove(context.TrainingSamples.Where(u => u.ID ==
ID).FirstOrDefault());

            context.SaveChanges();

            dataGridView4.Rows.Clear();
            var trainingSamples = context.TrainingSamples;
            foreach (var i in trainingSamples)
                dataGridView4.Rows.Add(i.ID, i.Name);
        }


        private void button6_Click(object sender, EventArgs e)
        {
            List<double[]> inputsList = new List<double[]>();



            int[] arrID = new int[dataGridView4.Rows.Count];
            int p = 0;

            foreach (DataGridViewRow row in dataGridView4.Rows)
            {
                arrID[p] = Convert.ToInt32(row.Cells[0].Value);
                p++;
            }

            for (int i = 0, j =0; i < arrID.Length; i++)
            {
                j = arrID[i];
                var q = context.ParameterValues.Where(u => u.TrainingSamples.ID == j);

                double[] Training = new double[q.Count()];
                int z = 0;
                foreach (var l in q)
                {
                    Training[z] = (double)l.Value;
                    z++;
                }
                inputsList.Add(Training);
```

```
            }
            label12.Text = "навчання";
            Learn = new NeuralNetwork(inputsList);

            Learn.Study();
            label12.Text = "мережа навчена";

        }

        private void button7_Click(object sender, EventArgs e)
        {
            double[] arr = new double[dataGridView7.Rows.Count* (dataGridView7.Columns.Count-
1)];
            for (int i = 0, k=0; i < dataGridView7.Rows.Count; i++)
            {
                for (int j = 1; j < dataGridView7.Columns.Count; j++)
                {
                    string str1 = dataGridView7.Columns[j].HeaderText.Substring(0,
dataGridView7.Columns[j].HeaderText.IndexOf(','));
                    string str2 = dataGridView7[0, i].Value.ToString();
                    arr[k] = Convert.ToDouble(dataGridView7[j, i].Value);
                    k++;
                }
            }
            Learn.Recognition( dataGridView8, arr, label9);

        }

        private void button8_Click(object sender, EventArgs e)
        {
            Refresh1(dataGridView5);
            Refresh1(dataGridView7);

            dataGridView4.Rows.Clear();
            var trainingSamples = context.TrainingSamples;
            foreach (var i in trainingSamples)
                dataGridView4.Rows.Add(i.Name);
        }

        private void dataGridView4_CellClick(object sender, DataGridViewCellEventArgs e)
        {

        }

        void Refresh1(DataGridView dataGrid)
        {
            dataGrid.Rows.Clear();
            dataGrid.Columns.Clear();
            dataGrid.Columns.Add("column" + dataGrid.Columns.Count + 1, "Епохи");

            var parametrs = context.Parameters;
            foreach (var i in parametrs)
            {
                dataGrid.Columns.Add("column" + dataGrid.Columns.Count + 1, i.Name +", "+
i.UnitsMeasurement);
            }

            var epoch = context.Epoch;
            foreach (var i in epoch)
            {
                dataGrid.Rows.Add(i.Name.ToString());
            }
            dataGrid.Columns[0].ReadOnly = true;
            dataGrid.DefaultCellStyle.SelectionBackColor =
dataGridView1.DefaultCellStyle.BackColor;
```

```
            dataGrid.DefaultCellStyle.SelectionForeColor =
dataGridView1.DefaultCellStyle.ForeColor;
        }

        private void dataGridView4_CellDoubleClick(object sender, DataGridViewCellEventArgs
e)
        {
            dataGridView5.Rows.Clear();
            dataGridView5.Columns.Clear();
            dataGridView5.Columns.Add("column" + dataGridView5.Columns.Count + 1, "Епохи");



            int ID = (int)dataGridView4[e.ColumnIndex, e.RowIndex].Value;

            var columnID = context.ParameterValues.Where(u => u.FK_TrainingSamples ==
ID).GroupBy(u => u.FK_Parameter);
            var rowID = context.ParameterValues.Where(u => u.FK_TrainingSamples ==
ID).GroupBy(u => u.FK_Epoch);

            foreach (var i in columnID)
            {
                dataGridView5.Columns.Add("column" + dataGridView5.Columns.Count + 1,
context.Parameters.Where(u => u.ID == i.Key).FirstOrDefault().Name);
            }

            foreach (var i in rowID)
            {
                dataGridView5.Rows.Add(context.Epoch.Where(u => u.ID ==
i.Key).FirstOrDefault().Name);
            }

            for (int i = 0; i < dataGridView5.Rows.Count; i++)
            {
                for (int j = 1; j < dataGridView5.Columns.Count; j++)
                {
                    string str1 = dataGridView5.Columns[j].HeaderText;
                    int ID1 = context.Parameters.Where(u => u.Name ==
str1).FirstOrDefault().ID;
                    string str2 = dataGridView5[0, i].Value.ToString();
                    int ID2 = context.Epoch.Where(u => u.Name == str2).FirstOrDefault().ID;
                    dataGridView5[j, i].Value = context.ParameterValues.Where(u =>
u.FK_Parameter == ID1 && u.FK_Epoch == ID2 &&
                    u.FK_TrainingSamples == ID).FirstOrDefault().Value;

                }
            }

            comboBox2.SelectedIndex = context.TrainingSamples.Where(u => u.ID ==
ID).FirstOrDefault().Results.ID-1;
        }
```

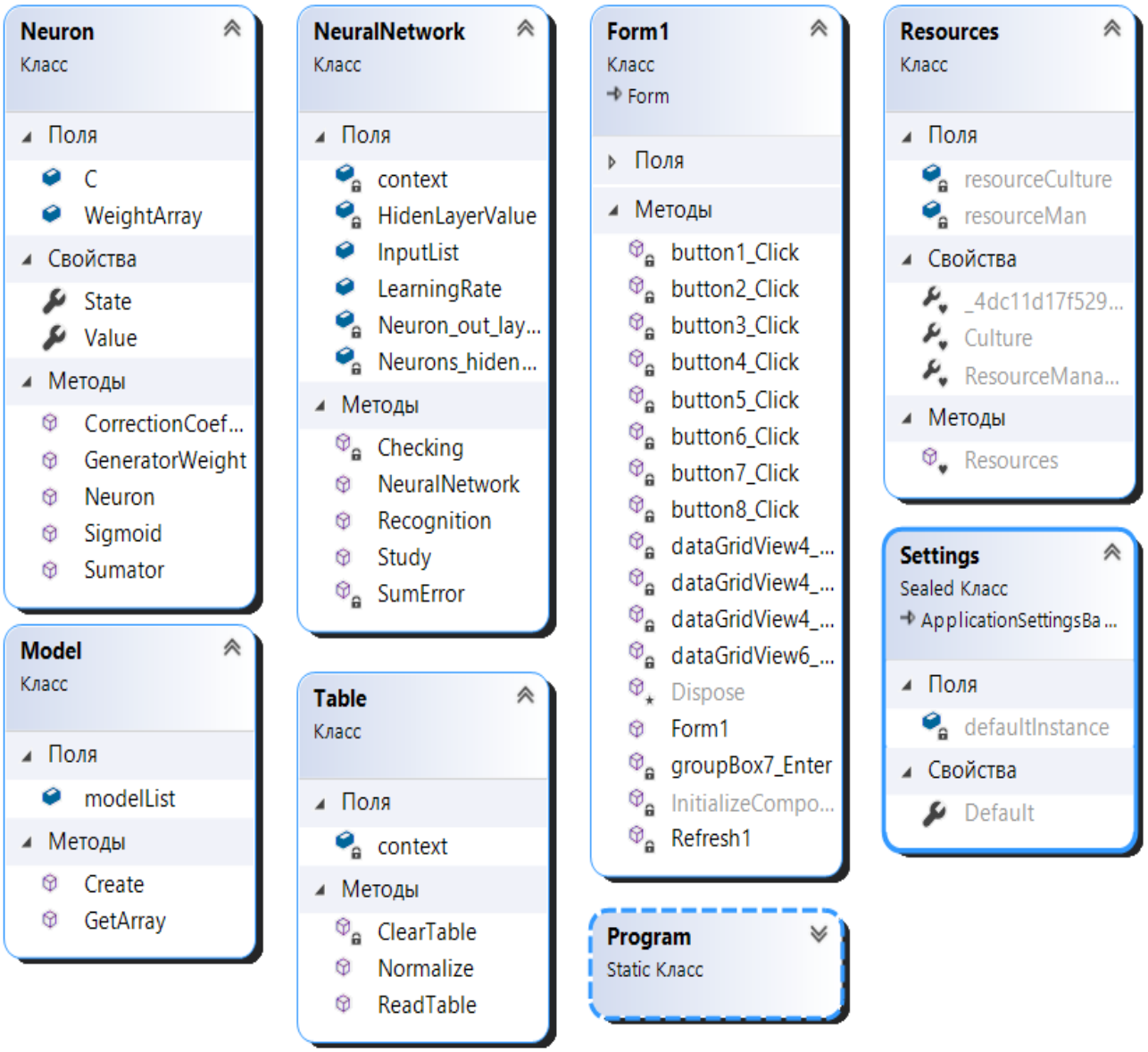
Neuron.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ExpertSystemCOVID
{

    class Neuron
    {
        public static double C = 0.02f;
```

```
        public double[] WeightArray;
        public int State { get; set; }
        public double Value { get; set; }

        public Neuron(double[] arr_x)
        {
            WeightArray = new double[arr_x.Count()];

            Array.Copy(GeneratorWeight(WeightArray.Count()), WeightArray,
WeightArray.Count());

            Value = Sigmoid(Sumator(arr_x, WeightArray));

        }

        public double Sigmoid(double sum)
        {
            return (double)(1 / (1.0 + Math.Pow(Math.E, -sum * C)));
        }

        public double Sumator(double[] array_x, double[] array_w)
        {
            double sum = 0;

            for (int i = 0; i < array_x.Count(); i++)
                sum += array_x[i] * array_w[i];

            return sum;
        }

        public double[] GeneratorWeight(int Length)
        {
            double[] arr = new double[Length];
            Random ran = new Random();
            for (int i = 0; i < arr.Count(); i++)
                arr[i] = (double)ran.NextDouble();

            return arr;
        }

        public void CorrectionCoefficient(double[] arr_x)
        {
            Value = Sigmoid(Sumator(arr_x, WeightArray));
        }

    }
}
```

Додаток Б

Діаграма класів експертної системи визначення рівня епідеміологічної небезпеки



Додаток Б

Схема інформаційної технології визначення рівня епідеміологічної небезпеки

Додаток Г

Презентаційний матеріал

Кваліфікаційна робота бакалавра

Експертна система визначення рівня епідеміологічної небезпеки

Виконав: студент 4 курсу групи КН-17-1 Овчарук О.М.

Керівник: к.т.н., доцент кафедри КНІТ Мазурець О.В.

Актуальність

Тема дослідження темпів поширення інфекційних захворювань сьогодні є особливо актуальною, оскільки в світі вирує пандемія коронавірусу. Для боротьби з коронавірусом на території України було впроваджено адаптивний карантин, який дозволяє впроваджувати один з видів карантинних обмежень в залежності від стану епідеміологічної ситуації в регіоні.

Отже, на сьогоднішній момент розробка комп'ютерного додатку, що реалізовуватиме можливості аналізу епідеміологічної ситуації в регіоні та буде брати до уваги стан не тільки поточної епідеміологічної ситуації, а також і тої їй передувала є актуальною.

Завдання

Метою кваліфікаційної роботи бакалавра є розробка експертної системи визначення рівня епідеміологічної небезпеки.

- Вимогою до системи є можливість створювати навчальні моделі з різною кількістю вхідних параметрів та епох, а при створенні вхідних параметрів надавати можливість вказувати їхні одиниці виміру та тип; за цими параметрами необхідно надати користувачеві можливість створювати навчальні вибірки.
- Слід забезпечити можливість формування експертного висновку з рекомендацією найбільш вірогідного результату, що особливо актуально для визначення рівня епідеміологічної ситуації у локальних межах в умовах адаптивного карантину. Такий підхід дозволяє врахувати те, що для визначення рівня епідеміологічної небезпеки слід брати до уваги не тільки поточний стан справ, а й ситуацію, що йому передувала, а також є доцільним при прийнятті рішень керуватись даними розвитку подій за аналогічних ситуацій.

Схема інформаційної технології визначення рівня епідеміологічної небезпеки

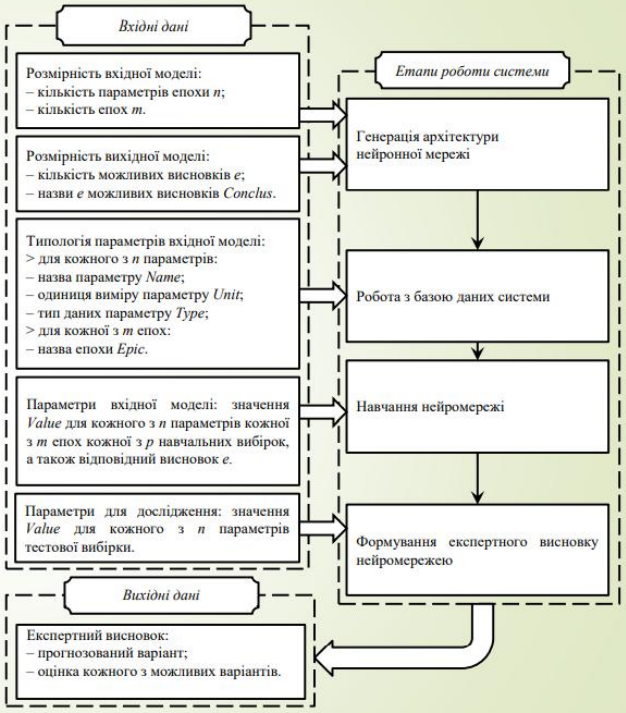
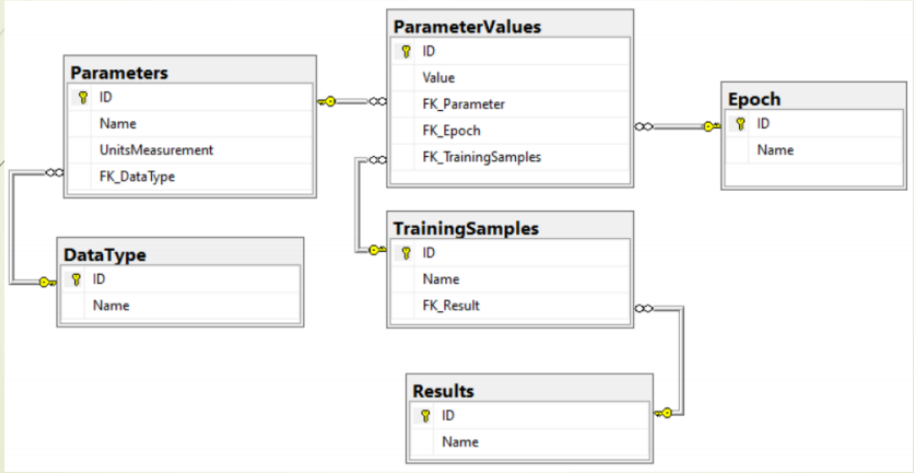
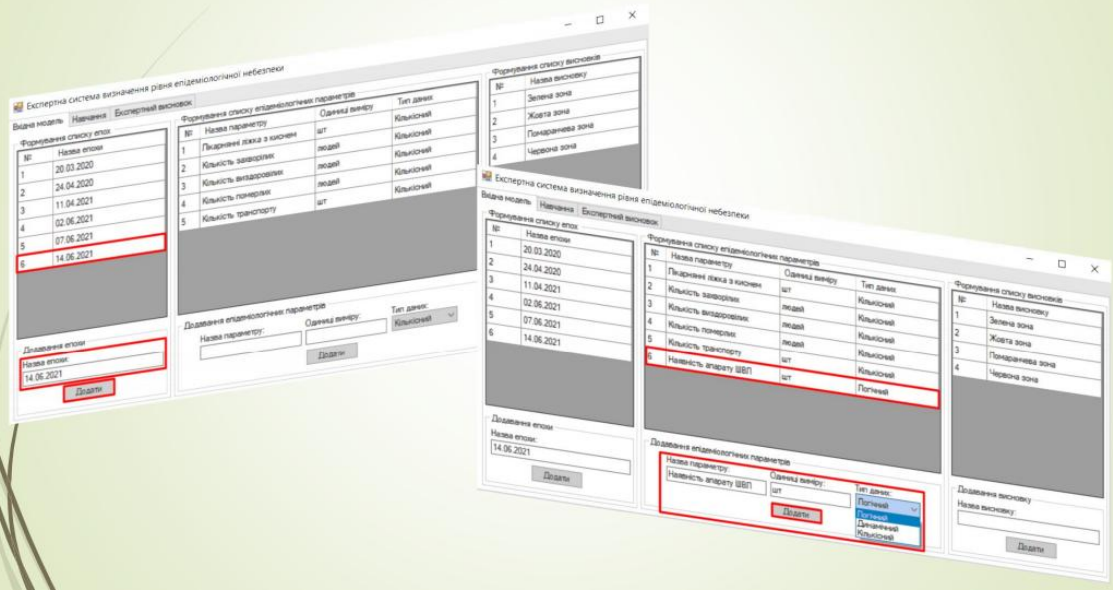


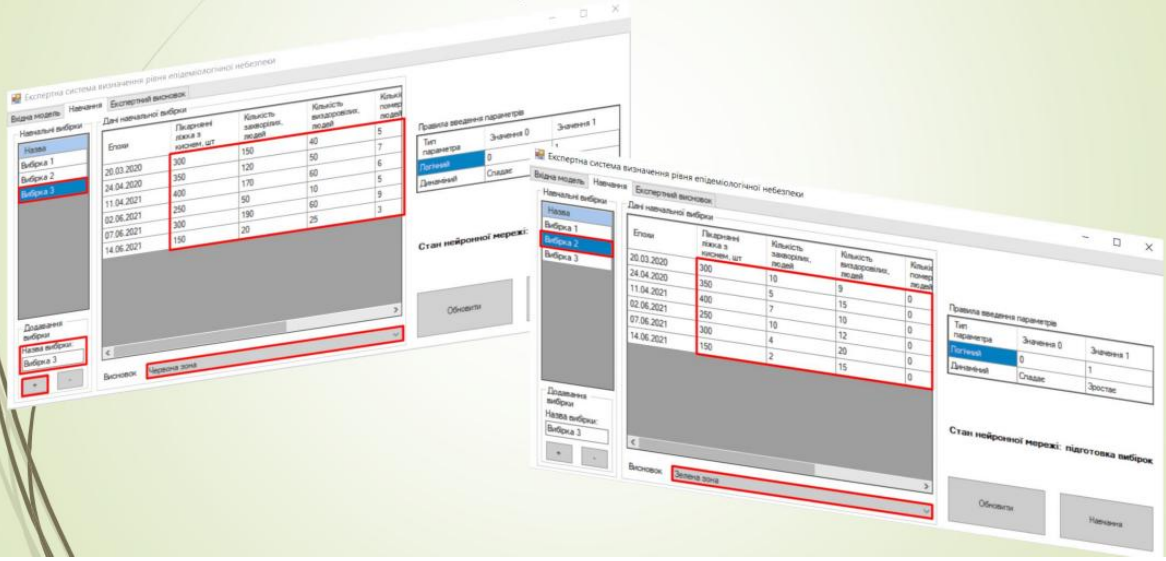
Схема бази даних



Додавання параметрів вхідної моделі



Формування навчальних вибірок та робота з ними



Висновки

В результаті виконання кваліфікаційної роботи бакалавра було розроблено експертну систему визначення рівня епідеміологічної небезпеки на платформі .NET із використанням нейромережевих технологій.

Розроблена експертна система визначення рівня епідеміологічної небезпеки призначена для визначення рівня епідеміологічної ситуації у локальних межах в умовах адаптивного карантину із використанням нейромережевих технологій. Експертна система визначення рівня епідеміологічної небезпеки дозволяє створювати навчальні моделі з різною кількістю вхідних параметрів та епох, при створенні вхідних параметрів надаючи можливість вказувати їхні одиниці виміру та тип, й здійснювати формування експертного висновку з рекомендацією найбільш вірогідного результату.

При роботі над кваліфікаційною роботою бакалавра автором виконано доповіді на XI Всеукраїнській науково-практичній конференції «Актуальні проблеми комп'ютерних наук АПКН-2019» та XII Всеукраїнській науково-практичній конференції «Актуальні проблеми комп'ютерних наук АПКН-2020», а також виконано 2 наукових публікації

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ГОЛОВІ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ

Направляється студент Овчарук О. М. (прізвище, ініціали) на захист дипломного проекту (роботи)

за спеціальністю 122 - Комп'ютерні науки

На тему: Експертна система визначення рівня епідеміологічної небезпеки

Дипломний проект (робота), рецензія і довідка про перевірку на плагіат додаються.

Декан факультету _____ (підпис) САВЕНКО О. С. (прізвище та ініціали)

ДОВІДКА УСПІШНОСТІ

Овчарук О. М. за період навчання на факультеті програмування та компютерних і телекомунікаційних систем з 2017 по 2021 роки. повністю виконав навчальний план спеціальності з такими розподілом оцінок за:
національною шкалою: відмінно 93,75 %, добре 6,25 %, задовільно 0,00 %.
шкалою ЄКТС: A 94,55 %, B 1,82 %, C 3,64 %, D 0,00 %, E 0,00 %.

Методист факультету _____ (підпис) _____ (прізвище та ініціали)

ВИСНОВОК КЕРІВНИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ) ТА ОБГРУНТУВАННЯ ОЦІНКИ

Студент Овчарук О. М. в своїй роботі використав нейронну мережу перцептрон для визначення рівня епідеміологічної небезпеки в регіонах України приймаючі рішення. Робота має інноваційну складову та виконана на високому професійному рівні. Є участь в конференціях та наукові публікації. Всі поставлені в роботі завдання Овчарук О. М. виконав.

Оцінка дипломного проекту (роботи) відмінно

Керівник дипломного прокту (роботи) _____ (підпис) Мазурець О. В. (прізвище та ініціали)

"08" 06 2021 р.

ВИСНОВОК КАФЕДРИ ПРО ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ)

Дипломний проект (роботу) розглянуто. Студент Овчарук О. М. допускається до захисту цього

Завідувач кафедри КНІТ (назва)

_____ (підпис, прізвище, ініціали)

"08" 06 2021 р.

Wed Jun 09 14:09:24 EEST 2021, Петровський Сергій Степанович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальное совпадение с одним документом 11.0%

Словари проверки: en_US, ru_RU, ua_UA. **Ошибок в документах: 13%**

ID: 92896 Название: Експертна система визначення рівня епідеміологічної небезпеки Добавлено в БД: 2021-06-09 Авторы: О.М. Овчарук Руководители: О.В. Мазурець Консультанты: Опоненты:	Документ		Суммарное совпадение по Базе Данных	
	Символы	Лексемы	Символы	Лексемы
	52998	454	8174 (15%)	82 (18%)

Источник плагиата

ID	Описание	Наличие плагиата в документе	
		Символы	Лексемы
90117	Название: ЗВІТ з професійної практики Добавлено в БД: 2021-05-11 Авторы: Овчарук О.М. Руководители: Скрипник Т.К. Консультанты: Опоненты:	5624 (11.0%)	57 (13.0%)

UNICHECK

Ім'я користувача:
Кафедра КН

ID перевірки:
1008246226

Дата перевірки:
09.06.2021 15:05:44 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
09.06.2021 15:10:05 EEST

ID користувача:
100005671

Назва документа: **КРБ Овчарук 20210608 4 фінал Lite**

Кількість сторінок: **53** Кількість слів: **7834** Кількість символів: **61205** Розмір файлу: **4.03 MB** ID файлу: **1008318499**

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

13.3%
Схожість

Найбільша схожість: **6.41%** з джерелом з Бібліотеки (ID файлу: **1008302248**)

6.14% Джерела з Інтернету	150	Сторінка 55
8.92% Джерела з Бібліотеки	68	Сторінка 56

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Підозріле форматування	15 сторінок

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: *Експертна система визначення рівня епідеміологічної небезпеки*

Автор: *студент групи КН-17-1 Овчарук Олександр Миколайович*

Спеціальність: *122 – Комп'ютерні науки*

Освітня програма: *освітньо-професійна*

Науковий керівник: *доцент кафедри КНІТ Мазурець О.В.*

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	*відповідає*
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) запозичення розміщені в розділах аналізу існуючих аналогів та прототипів, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;

2) усі запозичення фрагментарні;

3) до запозичень входять фрагменти програмного коду, що на мають авторства і містять поширені конструкції;

4) серед запозичень знаходяться загальновідомі терміни, скорочення та визначення.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 13.3 % і адресується до першоджерел, що, з урахуванням наведених обґрунтувань, відповідає характеру дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ О. В. Мазурець

Гарант ОП _____ О. В. Мазурець

Завідувач кафедри КНІТ _____ О. В. Бармак

Хмельницький національний університет
Кафедра комп'ютерних наук та інформаційних технологій

РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра

студента *гр. КН-17-1 Овчарука Олександра Миколайовича*

за темою *Експертна система визначення рівня епідеміологічної небезпеки*

1. Актуальність і значення теми *При формуванні експертного рішення щодо впровадження одного з видів карантинних обмежень враховується велика кількість параметрів, зібраних за великий період часу, зокрема наявність ліжок з підведеним киснем, кількість вільних та занятих лікарняних ліжок, прогрес захворюваності тощо. В цих умовах, актуальною є розробка експертної системи визначення рівня епідеміологічної небезпеки.*

2. Оцінка запропонованих моделей, підходів, алгоритмів, інформаційної складової та засобів розробки *Розроблені автором інформаційна технологія та структура компонентів інформаційної системи дозволяють забезпечити логічний взаємозв'язок елементів експертної системи та ефективно вирішувати задачу визначення рівня епідеміологічної небезпеки. Розроблена експертна система визначення рівня епідеміологічної небезпеки дозволяє створювати навчальні моделі з різною кількістю вхідних параметрів та епох, при створенні вхідних параметрів надаючи можливість вказувати їхні одиниці виміру та тип, й здійснювати формування експертного висновку з рекомендацією найбільш вірогідного результату.*

3. Оцінка розробленої інформаційної системи, її практична цінність та економічна доцільність *Розроблена експертна система визначення рівня епідеміологічної небезпеки призначена для визначення рівня епідеміологічної ситуації у локальних межах в умовах адаптивного карантину із використанням нейромережевих технологій.*

4. Загальний висновок та оцінка *Всі поставлені завдання роботи виконані повністю і на високому професійному рівні. Автором виконано доповіді на двох конференціях та наявні дві наукових публікації. За структурою, практичними цінностями, поставленій меті та вирішеними завданнями дана робота відповідає вимогам вищої школи і вимогам, що пред'являються до освітньо-кваліфікаційного рівня «бакалавр», тому її автор Овчарук О.М. заслуговує присвоєння кваліфікації бакалавра з комп'ютерних наук.*

Робота заслуговує на оцінку «відмінно».

Рецензент к.т.н., доц. каф. ТМіТ [signature] Очковський О.Ю.