Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання»

Виконав: студент 2 курсу, групи 2ІСТ-24м
спеціальності 126 – «Інформаційні системи та технології»
_____ Сергій НЕВОЛЯ
Керівник: к.т.н., доц. каф. САІТ
_____ Сергій ЖУКОВ
«27» 11 _____ 2025 р.
Опонент: к.т.н., доц. каф. КН
_____ Олексій СІЛАГІН
«03» 12 _____ 2025 р.

Допущено до захисту

Завідувач кафедри САІТ
_____ д.т.н., проф. Віталій МОКІН
«28» 11 _____ 2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра системного аналізу та інформаційних технологій
Рівень вищої освіти – другий (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 126 Інформаційні системи та технології
Освітньо-професійна програма – Інформаційні технології аналізу даних та зображень

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
_____ д.т.н., проф. Віталій МОКІН
«25» 09 2025 року

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Неволі Сергію Дмитровичу

1. Тема роботи: "Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання",
керівник роботи: Сергій ЖУКОВ, к.т.н., доц. каф. САІТ,
затверджені наказом ВНТУ від «24» 09 2025 року №313
2. Строк подання студентом роботи «28» 11 2025 року
3. Вихідні дані до роботи:
Набір даних «Lung Cancer Prediction» з відкритої бібліотеки платформи Kaggle.
4. Зміст текстової частини:
   - аналіз предметної області передбачення раку легень;
   - розвідувальний аналіз та обробка даних;
   - розробка інформаційної технології та аналіз результатів;
   - економічна частина.
5. Перелік ілюстративного матеріалу:
   - Розвідувальний аналіз даних;
   - Блок-схема алгоритму роботи інформаційної технології;
   - UML Component діаграма інформаційної технології;
   - UML Use-Case діаграма інформаційної технології;
   - Діаграма архітектури інформаційної технології;
   - Результати усіх моделей;
   - Графік впливовості ознак на найкращу модель.

6. Консультанти розділів МКР

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4	Олександр ЛЕСЬКО, к. е. н., проф. каф. ЕПВМ	05.11.2025	15.11.202

7. Дата видачі завдання «15» 09 2025 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналіз предметної області	15.09.2025	25.09.2025	виконано
2	Вибір оптимальних інформаційних технологій	25.09.2025	05.10.2025	виконано
3	Обробка даних	05.10.2025	10.10.2025	виконано
4	Розробка інформаційної технології	10.10.2025	05.11.2025	виконано
5	Економічна частина	05.11.2025	15.11.2025	виконано
6	Оформлення матеріалів до захисту МКР	15.11.2025	25.11.2025	виконано

Студент _____ Сергій НЕВОЛЯ

Керівник роботи _____ Сергій ЖУКОВ

АНОТАЦІЯ

- УДК 004.9:616.24-006

Неволя С. Д. Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання. Магістерська кваліфікаційна робота зі спеціальності 126 – інформаційні системи та технології, освітньо-професійна програма – інформаційні технології аналізу даних та зображень. Вінниця: ВНТУ, 2025. 129 с.

На укр. мові. Бібліогр.: 27 назв; рис.: 92; табл.: 11.

В магістерській кваліфікаційній роботі проведено огляд та аналіз різноманітних методів машинного навчання, що можуть бути використані для класифікації даних. Проведено розвідувальний аналіз даних та побудовано відповідні візуалізації. Побудовано моделі для передбачення та розроблено інформаційну технологію для передбачення раку легень.

Ілюстративна частина складається з 9 плакатів, що включають в себе результати передбачень розроблених моделей.

У розділі економічної частини розглянуто питання про доцільність розроблення та впровадження інформаційної технології передбачення раку легень методами машинного навчання.

Ключові слова: інформаційна технологія, Python, аналіз даних, машинне навчання, передбачення раку легень, Kaggle, рак легень.

ABSTRACT

Nevolya S. D. Information Technology of Analysis and Prediction of Lung Cancer by Machine Learning Methods. Master's Qualification Work in the Specialty 126 – Information Systems and Technologies, Educational and Professional Program – Information Technologies of Data and Image Analysis. Vinnytsia: VNTU, 2025. 129 p.

In Ukrainian. Language. Refs.: 27 titles; Fig.: 92; Tabl.: 11.

In the master's qualification work, a review and analysis of various machine learning methods that can be used to classify data was carried out. Intelligence analysis of data was carried out and appropriate visualizations were built. Models for prediction were built and an algorithm of information technology for predicting lung cancer was developed.

The illustrative part consists of 9, including the results of predictions of the developed models.

In the section of the economic part, the issue of the feasibility of developing and implementing information technology for predicting lung cancer by machine learning methods is considered.

Keywords: information technology, Python, data analysis, machine learning, lung cancer prediction, Kaggle, lung cancer.

ЗМІСТ

ВСТУП

Актуальність теми. Найбільш розповсюдженим у світі видом раку є рак легень, у той же час він і є одним із найбільш смертоносних. На ранніх стадіях діагностувати рак можуть лише такі методи як магнітно-резонансна та комп'ютерна томографії. Проблема в тому що симптоми раку легень, такі як кашель, є досить поширеними для багатьох хвороб, і зазвичай такі симптоми списують на інші захворювання, що погіршує становище пацієнта адже він не отримує необхідну діагностику оскільки лікарі зазвичай не підозрюють рак легень при таких простих симптомах.

На превеликий жаль ефективних ліків від раку, що могли б забезпечити стабільний позитивний результат у лікуванні, на даний момент немає. На сьогоднішній день медицина здатна відносно стабільно протистояти раку лише на початкових стадіях розвитку хвороби. З огляду на це, найефективнішою стратегією боротьби з онкологічними захворюваннями залишається вчасно проведена діагностика, а ще краще профілактика. Отже, розробка та дослідження нових ефективних методів прогнозування цієї хвороби є дуже актуальною.

Мета і задачі дослідження. Метою дослідження є підвищення точності передбачення ризику захворіти на рак легень шляхом розроблення інформаційної технології.

Для досягнення поставленої мети необхідно розв'язати такі задачі:

– провести аналіз проблематики діагностування раку легень;

– обрати найбільш оптимальні моделі машинного навчання та інформаційні технології;

– провести розвідувальний аналіз даних та здійснити генерування нових ознак;

– розробити інформаційну технологію для передбачення ризиків захворіти раком легень;

– виконати розрахунок економічної частини.

Об’єктом дослідження є процес розроблення інформаційної технології передбачення ризику раку легень у пацієнтів методами машинного навчання.

Предметом дослідження є інформаційна технологія аналізу та передбачення раку легень методами машинного навчання.

Методи дослідження. У роботі застосовано сучасні підходи аналізу даних, методи машинного навчання та аналізу впливовості ознак.

Новизна одержаних результатів. Дістала подальший розвиток інформаційна технологія передбачення ризику раку легень завдяки застосуванню сучасних методів аналізу та обробки даних, що дозволило підвищити точність передбачення ризику раку легень.

Практична цінність. Розроблено алгоритмічне забезпечення та програмні модулі мовою Python для реалізації інформаційної технології, що забезпечило отримання результатів, здатних підвищити ефективність виявлення раку легень у пацієнтів та покращити якість раннього попередження захворювання.

Апробація та публікації результатів магістерської кваліфікаційної роботи. Опубліковано тези на LV Всеукраїнській науково-технічній конференції підрозділів Вінницького національного технічного університету (м. Вінниця, 2025-2026 рр.) [1].

1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ПЕРЕДБАЧЕННЯ РАКУ ЛЕГЕНЬ

1.1 Аналіз проблематики раку легень

Рак виділяють як одну із най важчих хвороб у світі, оскільки від нього не знайшли ефективних ліків. Такий статус ця хвороба отримала через свою високу розповсюдженість та напрочуд високу летальність, особливо на пізніх стадіях розвитку. Найпоширеніший вид даної хвороби це рак легень, він же і є найбільш смертоносним з усіх видів. У 2022 році, згідно зі статистикою ВООЗ, рак легенів залишався однією з головних онкологічних проблем. Він становив 12,4% від загальної кількості нових випадків раку, досягнувши показника в 2,5 мільйона пацієнтів. Також це захворювання було основною причиною смертності від раку, забравши життя 1,8 мільйона осіб, що дорівнює 18,7% від усіх онкологічних смертей [2].

Хоча повноцінний засіб для повного усунення хвороби на всіх її етапах ще не знайдено, його поява не врятує тих, хто вже отримав діагноз. Це пов'язано з тим, що виявлення хвороби нерідко відбувається на пізніх стадіях, залишаючи критично мало часу для запровадження результативного лікування. Зважаючи на це, найефективнішими засобами боротьби із хворобою є її рання діагностика та профілактика захворюваності, а саме регулярні обстеження, особливо якщо людина перебуває під впливом певних ризиків. Оскільки успіх лікування безпосередньо корелює з ранньою діагностикою, актуальність розробки та вивчення ефективних методів прогнозування цієї хвороби критично важлива. Це не лише покращить результати для пацієнтів, але й суттєво зменшить фінансове навантаження на систему охорони здоров'я, пов'язане з лікуванням запущених стадій, що також є стимулом розробки.

1.2 Фактори ризику раку легень

Імовірність розвитку раку легень визначається не лише прямими ознаками, але й наявністю факторів ризику. Природна поява раку пов'язана з мутаціями, які мають певну частоту виникнення; чинники ризику є каталізаторами, що збільшують цю частоту, тим самим прискорюючи або уможливлюючи перехід мутації у злоякісний процес. Тому нижче буде проаналізовано ключові фактори ризику раку легень, які були офіційно досліджені та підтверджені наукою.

Куріння залишається ключовим і найбільш значущим чинником у розвитку раку легень. Ця залежність безпосередньо пояснює домінуюче становище легеневої онкології серед усіх ракових захворювань. Варто зазначити, що поширеність цієї звички критично висока: на планеті палить приблизно 1,1 мільярда осіб, що становить 14% світового населення. Така колосальна кількість людей є прямою цільовою аудиторією для ризику.

Пасивне куріння являє собою не менш підступну загрозу. Його механізм це вторинне вдихання диму, який виділяється при горінні тютюнового виробу або видихається активним курцем. Експерти неодноразово підтверджували, що шкідливий вплив цього фактору на клітини легень співставний із прямим курінням. Статистичні дані є загрозливими: у осіб, які постійно проживають із курцями, ризик виникнення раку легень зростає на 15–25% порівняно з тими, хто не піддається цьому впливу [3].

Канцерогени це хімічні сполуки, здатні провокувати розвиток онкологічних захворювань. Їхня дія відбувається шляхом безпосередньої взаємодії зі структурою ДНК клітин, що призводить до виникнення мутацій і подальшого формування злоякісних пухлин. Прикладами таких небезпечних речовин є смола, чадний газ, формальдегід, бензол, радон та азбест. Слід підкреслити, що значна концентрація цих сполук міститься у сигаретах або утворюється в результаті хімічних реакцій під час їхнього горіння.

Забруднене повітря містить широкий спектр шкідливих агентів, які негативно впливають на організм в цілому і, зокрема, на дихальну систему. До них

належать різноманітні хімікати, токсичні гази, димові частки та промисловий пил. Рівень забруднення оцінюється на основі показників надходження цих шкідливих компонентів в атмосферу. Джерелами цих надходжень можуть бути автотранспорт, великі промислові підприємства, теплові електростанції, аграрний сектор, а також природні явища, як-от лісові пожежі чи вулканічна активність.

Генетична схильність до розвитку раку легень може мати як успадкований, так і набутий характер. Вроджені схильності виникають у разі, якщо в сімейному анамнезі були випадки цього захворювання, або є результатом певних збоїв на етапі ембріонального розвитку. Натомість набута генетична схильність формується внаслідок змін у ДНК, спричинених впливом зовнішніх канцерогенних факторів, таких як тютюновий дим, радон, азбест та інші токсичні речовини.

Ризик виникнення раку легень значно збільшується з віком. Це пояснюється низкою чинників, але головним чином через природнє старіння організму та поступовим ослабленням імунної системи порівняно з молодими роками. Статистичні дані підтверджують, що особи молодше сорока років мають найнижчу ймовірність захворіти, за умови відсутності інших провокуючих чинників. Ризики прогресивно зростають із кожним наступним роком. Так, найвищі показники спостерігаються у віковій групі 59–70 років, особливо якщо йдеться про людей, які курять протягом кількох десятиліть.

Існують також фактори як от хронічні захворювання легень, вплив іонізуючого випромінювання, слабкий імунітет і т. д. [3].

1.3 Аналіз середовища та мови програмування для розробки технології передбачення раку легень

Передбачення такої хвороби як рак легень орієнтуючись на симптоми пацієнтів не допомагають ефективно лікувати цю хворобу. Адже проблема полягає у тому що симптомами раку легень сприймаються спочатку саме як звичайні хвороби, більш прості та звичні для лікарів. З дуже малим шансом через скарги

пацієнта на кашель лікарі запідозрять рак легень. З огляду на вищезазначене, найбільш доцільним рішенням є запобігання розвитку даного захворювання. Один із шляхів профілактики може полягати у створенні спеціалізованого алгоритму. Ця технологія, використовуючи комплекс діагностичних показників, які не вимагають застосування складних апаратних обстежень (наприклад, комп'ютерної або магнітно-резонансної томографії), матиме здатність сигналізувати про потенційну небезпеку для здоров'я людини.

Для вирішення задачі розробки подібної інформаційної технології для прогнозування ризиків доцільніше буде використовувати таку мову програмування як Python.

Python – є динамічною, об'єктно-орієнтованою та інтерпретованою мовою кодування й суворою динамічною типізацією. Її створення ініціював Гвідо ван Россум, програміст із Нідерландів, у 1990 році. Цей рік вважається роком її офіційного випуску, хоча робота розпочалася раніше. Назва була обрана розробником на честь відомого британського комедійного телесеріалу «Літаючий цирк Монті Пайтона», що транслювався у 70-х роках минулого століття [4].

Python сьогодні вважається однією з найбільш універсальних мов програмування, і її популярність зростає завдяки простому синтаксису та великій кількості готових рішень. Вона орієнтована на зрозумілість коду, тому навіть новачки можуть швидко почати писати свої перші програми, не витрачаючи багато часу на вивчення складних конструкцій. Саме ця простота робить Python зручним інструментом для навчання та побудови проєктів будь-якої складності.

Розглядаючи сфери, де Python застосовується найчастіше, варто відзначити його роль у робототехніці. За допомогою цієї мови можна керувати датчиками, обробляти сигнали від сенсорів, створювати модулі для автономних систем та взаємодіяти з мікроконтролерами через спеціальні бібліотеки. Python дозволяє швидко тестувати та вдосконалювати алгоритми, що робить його дуже корисним у створенні прототипів роботів і механізмів.

Ще однією великою сферою використання Python є машинне навчання та штучний інтелект. Перевага цієї мови полягає в тому, що вона має безліч

спеціальних бібліотек, які спрощують роботу з моделями, математичними розрахунками та великими масивами даних. Завдяки цьому на Python створюють програми, здатні розпізнавати зображення, аналізувати текст, прогнозувати поведінку систем, працювати з нейронними мережами й навіть генерувати новий контент.

Важливою областю також є інженерія даних. Python часто використовують для збирання, очищення та структурування інформації перед її подальшим аналізом. Мова дає можливість автоматизувати різні етапи обробки даних, інтегруватися з базами даних та будувати конвеєри обробки інформаційних потоків. Це робить її незамінною у компаніях, які працюють із великими обсягами інформації [4].

Автоматизація це ще один напрям, де Python проявляє себе надзвичайно ефективно. З його допомогою розробники створюють скрипти, що виконують рутинні процеси без участі людини, наприклад обслуговують сервери, керують файлами, перевіряють системи або проводять технічний моніторинг. Це суттєво економить час і зменшує кількість помилок, які виникають від ручного виконання повторюваних завдань.

Окремо варто згадати можливість використання Python навіть для програмування мікроконтролерів. Наприклад, існують спеціальні версії мови, такі як MicroPython та CircuitPython, які дозволяють писати компактний та ефективний код для невеликих пристроїв. Це відкриває шлях до створення «розумних» гаджетів, інтернет-пристроїв та різних електронних модулів.

Якщо поглянути на сучасні мови програмування ширше, стає очевидним, що універсального інструмента, який би однаково добре підходив для будь-якого типу завдань, не існує. Кожна мова має свої сильні й слабкі сторони. Python, наприклад, чудово справляється із задачами високого рівня, але у сферах на кшталт розробки драйверів обладнання або створення ядра операційних систем його можливостей недостатньо. Це пов’язано з тим, що він не забезпечує належного рівня продуктивності та не має суворої типізації, яка є критичною для низького рівня програмування. Водночас популярність Python у сучасних IT-галузях пояснюється

його глибокою інтеграцією у сферу штучного інтелекту та роботу з даними. Високі позиції Python у рейтингах мов програмування лише підтверджують його активний розвиток і стійке зростання в спільноті розробників [5].

Однією з характерних проблем, із якою користувачі Python можуть зіткнутися, є порівняно невисока швидкість виконання програм. Це пояснюється тим, що Python працює як інтерпретована мова. Перед виконанням код перетворюється в байт-код, який обробляється віртуальною машиною Python. У результаті програми, написані на Python, зазвичай повільніші за аналогічні рішення, створені на C або інших компільованих мовах. Попри це, у сучасних умовах апаратне забезпечення настільки потужне, що різниця у швидкодії часто не є критичною. Натомість швидкість написання та гнучкість розробки виходять на перший план, і саме тут Python демонструє свою силу. До того ж мову можна підсилити модулями, створеними на C або C++, що дає змогу прискорити частини програм і компенсувати недоліки продуктивності.

Особливо значущим аспектом Python є екосистема бібліотек, що охоплює майже всі напрямки машинного навчання та роботи з великими обсягами інформації. Ці інструменти дозволяють розробникам легко збирати, очищувати, структурувати й аналізувати дані, а також будувати, тестувати та візуалізувати моделі різної складності. Розмаїття готових рішень значно скорочує час розробки та дає можливість зосередитись на логічному аспекті моделі, а не на технічних деталях її реалізації. До того ж Python добре працює з асинхронними процесами, що робить його ефективним вибором для обробки великих масивів інформації. Усе це в комплексі формує середовище, у якому мова стає одним із найоптимальніших варіантів для вирішення задач машинного навчання та аналітики.

Оскільки завдання належить до медичної галузі, для його реалізації доречно використовувати популярні Python-бібліотеки, такі як scikit-learn, TensorFlow чи PyTorch.

Бібліотека scikit-learn займає важливе місце у сфері аналізу медичних даних завдяки великому набору алгоритмів машинного навчання, які дозволяють вирішувати широкий спектр задач — від простих класифікаційних моделей до

складних методів прогнозування. Вона вирізняється зручністю використання, стабільністю роботи та широкою підтримкою наукової спільноти, що робить її одним з найбільш популярних інструментів у медичній аналітиці.

Однією з ключових переваг scikit-learn є її універсальність. У бібліотеці зібрано моделі, які добре підходять для різних типів медичних завдань. Наприклад, алгоритми класифікації застосовуються для розпізнавання захворювань на основі даних пацієнтів, таких як симптоми, результати лабораторних аналізів або фізіологічні параметри. Логістична регресія, дерева рішень, метод опорних векторів чи ансамблеві підходи дають змогу виявити закономірності, які складно помітити вручну, та допомагають лікарям отримувати додаткові інструменти для прийняття рішень.

Крім класифікації, scikit-learn пропонує широкий вибір моделей для прогнозування медичних ризиків. Регресійні методи дозволяють оцінювати ймовірність розвитку певного стану в майбутньому, що є особливо важливим для задач профілактики та раннього втручання. Аналіз виживаності, моделі градієнтного підсилення та інші алгоритми допомагають будувати індивідуальні прогностичні моделі, які можуть використовуватися для оцінки ризику рецидиву, прогнозу прогресування хвороби або визначення ефективності лікування. Важливим аспектом є також можливість попередньої обробки даних, яку забезпечує scikit-learn. У медичній практиці дані часто містять пропуски, шум або значні відмінності між пацієнтами. Інструменти для нормалізації, масштабування, кодування категоріальних змінних та вибору релевантних ознак допомагають перетворити «сирі» медичні записи у якісний набір для моделювання. Це дозволяє підвищити точність прогнозів та стабільність алгоритмів [6].

Додатковою сильною стороною scikit-learn є її здатність забезпечувати оперативне випробування множинних алгоритмів для розв'язання одного завдання. У сфері медичних досліджень критично важливо мати можливість зіставляти різні підходи і точно оцінювати їхню результативність на основі клінічних даних. Завдяки уніфікованому програмному інтерфейсу для тренування та валідації

моделей, науковець може легко проводити експерименти з різноманітними алгоритмами, щоб визначити оптимальний для конкретної патології або проблеми.

Усі ці технічні особливості перетворюють scikit-learn на високоефективний інструмент для опрацювання медичної інформації. Ця бібліотека сприяє створенню прогностичних систем, які можуть суттєво підтримувати роботу лікарів, підвищувати достовірність діагностичних висновків та прогнозування перебігу хвороби, а також забезпечувати підґрунтя для розробки індивідуалізованих стратегій терапії.

Завдячуючи інтуїтивному синтаксису, адаптивності та варіативним вибором бібліотек, Python ідеально пасує для реалізації завдання передбачення онкології легень.

Для дослідження теми та впровадження технічної частини було обрано платформу для машинного навчання та науки про дані Kaggle. Онлайн-платформа Kaggle є однією з найвідоміших світових спільнот, присвячених машинному навчанню, аналізу даних та дослідженням у сфері штучного інтелекту. Вона виступає одночасно як навчальне середовище, місце для експериментів та простір для конкуренції між фахівцями різного рівня від студентів до провідних дослідників. Kaggle надає відкритий доступ до тисяч датасетів різної тематики: медичні зображення, фінансові дані, результати опитувань, текстові корпуси, дані про пацієнтів, генетичні послідовності та багато іншого. Це значно спрощує навчання та роботу над проєктами з машинного навчання, адже користувачі отримують готовий матеріал для моделювання без необхідності збирати дані самостійно [7]. Зазначимо також що датасет, який використовуватиметься в дослідженні був взятий з відкритої бібліотеки Kaggle.

1.4 Аналіз оптимальних методів машинного навчання

Мову Python можна розглядати як інструмент, що пропонує широкий набір алгоритмів та підходів для розв’язання різноманітних обчислювальних і

аналітичних задач. У межах цієї роботи основна увага зосереджена на прогнозуванні ризиків раку легень, що класифікується за трьома рівнями ризику: низьким, середнім та високим. Така постановка задачі природним чином належить до напряму класифікації. Нижче наведено приклади найпоширеніших методів побудови моделей прогнозування, які часто застосовуються у Python:

Логістична регресія (Logistic Regression) – це метод регресійного аналізу, основною метою якого є дослідження взаємозв'язку між так званими предикторами та регресорами, які являють собою змінні, в яких не прослідковується залежність та між змінною, яка навпаки прямо залежить від них. За основу методу взята логістична функція, яка з математичної точки зору розраховує відсоткові шанси для усіх наявних класів вхідних даних та обирає варіант що має найбільшу ймовірність. Цей метод широко застосовується у задачах класифікації, за умови що вхідні дані є категоріальними [8].

Метод опорних векторів (Support Vector Machines – SVM) – є одним із популярних алгоритмів машинного навчання, який застосовується для розв'язання задач класифікації та регресії. Його принцип роботи полягає в тому, що дані відображаються у багатовимірному просторі, де модель будує гіперплощину, яка оптимально розділяє об'єкти різних класів. Коли надходять нові дані, класифікація здійснюється на основі того, по який бік гіперплощини вони розташовані.

Цей підхід ефективний у випадках, коли навчальна вибірка є невеликою, але має значну варіативність. SVM відзначається високою точністю, гнучкістю та надійністю, однак має підвищену обчислювальну складність і чутливість до наявності аномальних спостережень [9].

Випадковий ліс (Random Forest) – це ансамблевий алгоритм машинного навчання, який застосовується для розв'язання задач класифікації та регресії. Принцип його роботи полягає у створенні великої кількості дерев рішень, кожне з яких навчається на випадково вибраній підмножині даних.

Основою методу є принцип беггінгу (Bootstrap Aggregating), цей підхід ґрунтується на ідеї формування кількох незалежних моделей, де кожне дерево будується на окремому наборі даних, отриманому шляхом випадкового вибору

елементів. Такий спосіб забезпечує надійні результати, добре запобігає перенавчанню, легко реалізується та демонструє високу ефективність як для великих, так і для малих наборів даних. Водночас, недоліком методу є його відносно висока обчислювальна складність [10].

Надлишкові дерева (Extra Trees) – це ансамблевий метод машинного навчання, ідея методу полягає у генерації значної кількості дерев рішень, а потім комбінуванні їхніх окремих передбачень задля досягнення максимальної точності.

Особливістю цього підходу є випадковий вибір ознак: під час побудови кожного вузла випадковим чином обираються ознаки для розгалуження. Це підвищує стійкість моделі до перенавчання порівняно, наприклад, із Random Forest, де для кожного розгалуження можуть використовуватись усі доступні ознаки. Алгоритм, як правило, формує глибокі дерева, тобто дерева з великою кількістю рівнів, що дозволяє краще відображати складні взаємозв’язки між ознаками та класами.

До основних переваг методу належать висока точність класифікації, стійкість до забруднення, які можуть бути в даних та які негативно впливають на більшість моделей, а також простота реалізації й налаштування. Водночас, можливість перенавчання все ж залишається — хоча вона менша, ніж у Random Forest, та все ж може стати вагомою при надмірній кількості чи глибині дерев [11].

Дерева рішень (Decision Trees) – ця модель широк використовується у задачах передбачення та розподілу між класами, яка особливо ефективна в задачах із чітко визначеною послідовністю дій і правилами прийняття рішень.

Структурно дерево рішень складається з вузлів і ребер: вузли відображають певні рішення або перевірки умов, тоді як ребра показують можливі шляхи переходу між ними. Початкова точка, з якої починається аналіз, називається кореневим вузлом. Вузли, що не мають подальших відгалужень, називаються листовими, саме вони містять кінцеві результати або класифікаційні висновки.

Процес побудови дерева відбувається рекурсивно: кожен вузол розділяється на підвузли на основі найкращого атрибуту, який найточніше розподіляє дані на підмножини з подібними цільовими значеннями. Розгалуження триває доти, поки

не сформуються листові вузли. Прогнозування нових даних здійснюється шляхом проходження від кореня до відповідного листа відповідно до значень ознак [12].

До основних переваг моделі належать її проста та інтуїтивно зрозуміла структура, здатність ефективно працювати з великими обсягами даних і стійкість до відсутніх значень у вибірці. Особливо важливо зазначити, що дерева рішень добре підходять для медичної діагностики, оскільки дозволяють визначати захворювання на основі симптомів пацієнтів.

Однією з найвагоміших переваг є можливість візуалізації: існує широкий набір інструментів, які дозволяють не лише легко інтерпретувати процес прийняття рішень і результати моделі, а й наочно представити їх у вигляді зрозумілої схеми чи графіка.

Втім, модель має й певні недоліки до яких можна віднести схильність до перенавчання у разі надто глибокої структури дерева, та високий вплив шумів, що може знижувати її здатність узагальнювати дані.

Adaptive Boosting є методом машинного навчання основна мета якого полягає в підвищенні ефективності класифікаторів шляхом послідовного навчання кількох слабких моделей та їх об’єднання для формування одного більш потужного класифікатора.

Алгоритм працює ітеративно: під час кожної ітерації створюється слабкий класифікатор, якому надається певна вага, що відображає його точність. У процесі навчання особлива увага приділяється складним прикладам, на яких попередні класифікатори найчастіше помилялися. Модель поступово вдосконалюється, коригуючи ваги після кожної ітерації, доки не буде досягнуто мінімальної похибки або встановленої кількості повторів.

Цей метод вирізняється простотою, зрозумілістю та гнучкістю, адже його можна комбінувати з різними типами класифікаторів. Водночас AdaBoost є чутливим до шуму в даних і може бути схильним до запам’ятовування даних [13].

XGB Classifier – основа даного методу ґрунтується на принципі градієнтного бустингу, для подальшого використання під час формування лісу із дерев рішень. Його робота полягає в послідовному навчанню слабких моделей дерев рішень із

поступовим виправленням помилок попередніх і мінімізацією загальної похибки. Нові моделі навчаються на залишкових помилках попередніх, що дозволяє підвищувати точність прогнозів.

Метод застосовує розподілені обчислення, завдяки чому забезпечується висока швидкість обробки даних. Основними перевагами XGB Classifier є його точність, гнучкість у налаштуванні та наявність різних механізмів регуляризації, які допомагають запобігти перенавчанню — проблемі, до якої метод особливо схильний. Тому правильне налаштування параметрів моделі є важливою умовою успішного навчання та ефективного застосування [14].

Мультишаровий перцептрон (Multi-Layer Perceptron) – один із широко відомих методів у машинному навчанні, який часто називають мережею з повністю зв’язаними нейронами. Принцип роботи даного методу базується на використанні нейронної мережі, яка побудована на основі послідовних, зв’язаних між собою, шарів нейронів для виконання задач класифікації або прогнозування.

Структура багатошарового перцептрона складається з вхідного шару, одного чи кількох прихованих шарів та вихідного шару. Вхідний шар приймає початкові дані, у прихованих шарах відбуваються обчислення та перетворення сигналів, а вихідний шар формує кінцевий результат — прогноз класифікації, що базується на попередніх обчисленнях.

Метою алгоритму є мінімізація помилки класифікації, тобто зменшення різниці між прогнозованими та фактичними значеннями. До основних переваг цього методу належить здатність виявляти складні, неочевидні залежності між змінними, а також його ефективність у задачах класифікації та регресії. Проте до недоліків можна віднести значні вимоги до обчислювальних ресурсів і схильність до перенавчання, особливо коли обсяг навчальних даних є недостатнім [15, 16].

Полегшений градієнтний бустинг (Light Gradient Boosting Machine) – це високопродуктивний метод машинного навчання, який належить до класу градієнтного бустингу над деревами рішень. Його основна ідея полягає у створенні ансамблю слабких моделей, зазвичай невеликих дерев рішень, при цьому усі нові моделі прагнуть виправити помилки попередніх. В противагу традиційним

методам бустингу, LightGBM має власні інноваційні підходи, що роблять навчання швидшим і ефективнішим. Модель створює структуру, у якій кожне дерево навчається на залишкових помилках попередніх, поступово зменшуючи похибку прогнозування. Особливістю LightGBM є побудова дерев по напрямку глибини (leaf-wise growth), а не рівнями, як у більшості алгоритмів. Це дозволяє досягати більшої точності, хоча іноді може призводити до перенавчання.

Перевагами методу є висока швидкість навчання, ефективність при роботі з великими наборами даних, вміння працювати із числовими та категоріальними ознаками, а також вбудовані засоби, які автоматично регулюють навчання та знижують ризик перенавчання. У медичних дослідженнях LightGBM часто застосовується для прогнозування результатів лікування, виявлення ризиків захворювань та аналізу клінічних факторів.

Недоліками моделі можна вважати чутливість до вибору гіперпараметрів, які потребують ретельного налаштування для уникнення перенавчання [17].

Метод k-Nearest Neighbors (метод k найближчих сусідів) – це один із найпростіших, але водночас інтуїтивно зрозумілих методів класифікації та регресії. Його основна ідея полягає в тому, що об’єкт належить до класу, якому належить більшість його найближчих сусідів у просторі ознак. Тобто рішення приймається не на основі побудови складної моделі, а шляхом аналізу схожості між об’єктами.

Під час прогнозування k-NN обчислює відстань (частіше всього евклідову) між новим зразком та всіма наявними у тренувальній вибірці, після чого обирає k найближчих. Клас нової точки визначається як найчастіший серед цих сусідів (для класифікації) або як середнє значення (для регресії).

До переваг методу належать простота реалізації та висока гнучкість при роботі з різними типами даних. У медичному контексті метод найближчих сусідів часто використовується для класифікації пацієнтів за схожістю симптомів.

Серед недоліків можна виділити низьку швидкість прогнозування при великих об’ємах даних, чутливість до масштабування ознак та схильність до зниження точності при наявності шуму або нерівномірному розподілі даних [18].

1.5 Висновки

У межах розділу здійснено ґрунтовний аналіз проблематики діагностики та лікування раку легень у сучасних умовах, що дозволило визначити ключові недоліки наявних підходів. Встановлено, що традиційні методи діагностування часто забезпечують виявлення захворювання на пізніх стадіях, що зумовлює високий рівень смертності та потребує впровадження інноваційних технологій для раннього прогнозування ризиків розвитку легеневої онкології.

Проведено аналіз основних факторів, що впливають на ймовірність виникнення раку легень, із виокремленням найбільш критичних для подальшого моделювання. Додатково опрацьовано інструментарій, необхідний для реалізації інформаційної технології, зокрема програмну мову Python та платформу Kaggle, що забезпечують широкі можливості для проведення аналізу даних і побудови моделей машинного навчання.

Також розглянуто та оцінено релевантні методи машинного навчання, які доцільно застосувати для створення інформаційної технології прогнозування ризику раку легень. Це забезпечує методологічне підґрунтя для подальшої практичної реалізації системи, спрямованої на підвищення точності прогнозування та прийняття рішень у медичній галузі.

2. РОЗВІДУВАЛЬНИЙ АНАЛІЗ ТА ОБРОБКА ДАНИХ

2.1 Попередня обробка даних

При пошуку даних для проведення дослідження було надано перевагу набору даних у Kaggle, що має назву «Lung Cancer Prediction». Цей набір даних був створений та завантажений на платформу користувачем «The Devastator». Набір даних знаходиться у відкритому доступі [19]. Наданий датасет включає різноманітну інформацію про пацієнтів із ризиком розвитку раку легень, яку надає журнал Nature Medicine. Серед доступних характеристик присутні вік, вплив забрудненого повітря, паління, споживання алкоголю, професійні та генетичні фактори ризику, а також 19 інших параметрів. Загальна кількість записаних пацієнтів у наборі становить 1000 осіб (рис. 2.1).

index	Patient Id	Age	Gender	Air Pollution	Alcohol use	Dust Allergy	OccuPational Hazards	Genetic Risk	chronic Lung Disease	...	Fatigue	Weight Loss	Shortness of Breath	Wheezing
0	P1	33	1	2	4	5	4	3	2	...	3	4	2	2
1	P10	17	1	3	1	5	3	4	2	...	1	3	7	8
2	P100	35	1	4	5	6	5	5	4	...	8	7	9	2
3	P1000	37	1	7	7	7	7	6	7	...	4	2	3	1
4	P101	46	1	6	8	7	7	7	6	...	3	2	4	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	P995	44	1	6	7	7	7	7	6	...	5	3	2	7
996	P996	37	2	6	8	7	7	7	6	...	9	6	5	7
997	P997	25	2	4	5	6	5	5	4	...	8	7	9	2
998	P998	18	2	6	8	7	7	7	6	...	3	2	4	1
999	P999	47	1	6	5	6	5	5	4	...	8	7	9	2

Рисунок 2.1 – Набір даних «Lung Cancer Prediction»

Проаналізуємо повний список характеристик, які містить датасет. Ознаки «Індексу», «ID людини» та «Віку» є числовими. До категоріальних ознак, які характеризують рівень впливу ознаки на людину відносяться: «Стать людини»,

«Забруднене повітря», «Вживання алкоголю», «Пилова алергія», «Професійні ризики», «Генетичні ризики», «Хронічні захворювання легень», «Збалансована дієта», «Ожиріння», «Куріння», «Пасивне куріння», «Біль у грудях», «Кашель з кров'ю», «Втома», «Втрата ваги», «Задишка», «Хрипи», «Труднощі ковтання», «Булини нігтів».

Провівши аналіз ознак було вирішено прибрати «Індекс» та «ID людини», тому що для подальшого дослідження ці характеристики нам не потрібні та будуть заважати (рис. 2.2).

```
df.drop(columns=['index', 'Patient Id'], axis=1, inplace=True)
df
```

	Age	Gender	Air Pollution	Alcohol use	Dust Allergy	OccuPational Hazards	Genetic Risk	chronic Lung Disease	Balanced Diet	Obesity	...	Fatigue	Weight Loss	Shortness of Breath	Wheezing
0	33	1	2	4	5	4	3	2	2	4	...	3	4	2	2
1	17	1	3	1	5	3	4	2	2	2	...	1	3	7	8
2	35	1	4	5	6	5	5	4	6	7	...	8	7	9	2
3	37	1	7	7	7	7	6	7	7	7	...	4	2	3	1
4	46	1	6	8	7	7	7	6	7	7	...	3	2	4	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
995	44	1	6	7	7	7	7	6	7	7	...	5	3	2	7
996	37	2	6	8	7	7	7	6	7	7	...	9	6	5	7
997	25	2	4	5	6	5	5	4	6	7	...	8	7	9	2
998	18	2	6	8	7	7	7	6	7	7	...	3	2	4	1
999	47	1	6	5	6	5	5	4	6	7	...	8	7	9	2

1000 rows × 24 columns

Рисунок 2.2 – Очищення непотрібних ознак

Далі проведемо аналіз та дослідимо дані на наявність шумів, для запобігання їх негативному впливу під час моделювання. Розмір набору даних невеликий, тому для аналізу вистачить функції describe(), яка продемонструє нам кількість можливих шумів та де вони знаходяться (рис. 2.3).

```
df.describe()
```

	Age	Gender	Air Pollution	Alcohol use	Dust Allergy	OccuPational Hazards	Genetic Risk	chronic Lung Disease	Balanced Diet	Obesity
count	1000.000000	1000.000000	1000.0000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000
mean	37.174000	1.402000	3.8400	4.563000	5.165000	4.840000	4.580000	4.380000	4.491000	4.465000
std	12.005493	0.490547	2.0304	2.620477	1.980833	2.107805	2.126999	1.848518	2.135528	2.124921
min	14.000000	1.000000	1.0000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
25%	27.750000	1.000000	2.0000	2.000000	4.000000	3.000000	2.000000	3.000000	2.000000	3.000000
50%	36.000000	1.000000	3.0000	5.000000	6.000000	5.000000	5.000000	4.000000	4.000000	4.000000
75%	45.000000	2.000000	6.0000	7.000000	7.000000	7.000000	7.000000	6.000000	7.000000	7.000000
max	73.000000	2.000000	8.0000	8.000000	8.000000	8.000000	7.000000	7.000000	7.000000	7.000000

Coughing of Blood	Fatigue	Weight Loss	Shortness of Breath	Wheezing	Swallowing Difficulty	Clubbing of Finger Nails	Frequent Cold	Dry Cough	Snoring
1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000
4.859000	3.856000	3.855000	4.240000	3.777000	3.746000	3.923000	3.536000	3.853000	2.926000
2.427965	2.244616	2.206546	2.285087	2.041921	2.270383	2.388048	1.832502	2.039007	1.474686
1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000
3.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000
4.000000	3.000000	3.000000	4.000000	4.000000	4.000000	4.000000	3.000000	4.000000	3.000000
7.000000	5.000000	6.000000	6.000000	5.000000	5.000000	5.000000	5.000000	6.000000	4.000000
9.000000	9.000000	8.000000	9.000000	8.000000	8.000000	9.000000	7.000000	7.000000	7.000000

Рисунок 2.3 – Ініціалізація функції describe()

За результатами можемо бачити що значень які б очевидно вказували на аномалії в наборі не зафіксовано. Категоріальні ознаки в нормі та в межах між одиницею та дев’яткою.

Зупинимось детальніше на ознаках «Вік» та «Стать людини». Для зручності використання гендерна ознака поділяється на чоловіків та жінок та позначається одиницею та двійкою відповідно. Вікова ознака включає у собі перелік прожитих років де мінімальне значення віку особи 14 років, та максимальне значення 73 роки.

Для проведемо аналіз пропущених значень, у записах за допомогою функції info() (рис. 2.4). Якщо у наборі даних є записи із пропусками, функція покаже ці пропуски позначивши їх при цьому як NaN. Якщо такі записи будуть виявлені,

необхідно буде видалити їх, тому що вони надають невірне представлення та неповну картину даних, що негативно вплине на моделювання.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 24 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       1000 non-null   int64
 1   Gender                    1000 non-null   int64
 2   Air Pollution             1000 non-null   int64
 3   Alcohol use               1000 non-null   int64
 4   Dust Allergy              1000 non-null   int64
 5   OccuPational Hazards      1000 non-null   int64
 6   Genetic Risk              1000 non-null   int64
 7   chronic Lung Disease      1000 non-null   int64
 8   Balanced Diet             1000 non-null   int64
 9   Obesity                   1000 non-null   int64
 10  Smoking                   1000 non-null   int64
 11  Passive Smoker            1000 non-null   int64
 12  Chest Pain                1000 non-null   int64
 13  Coughing of Blood         1000 non-null   int64
 14  Fatigue                   1000 non-null   int64
 15  Weight Loss               1000 non-null   int64
 16  Shortness of Breath       1000 non-null   int64
 17  Wheezing                  1000 non-null   int64
 18  Swallowing Difficulty     1000 non-null   int64
 19  Clubbing of Finger Nails  1000 non-null   int64
 20  Frequent Cold             1000 non-null   int64
 21  Dry Cough                 1000 non-null   int64
 22  Snoring                   1000 non-null   int64
 23  Level                     1000 non-null   object
dtypes: int64(23), object(1)
memory usage: 187.6+ KB
```

Рисунок 2.4 – Ініціалізація функції info()

Аналіз за допомогою функції info() показав, що в цьому датасеті немає пропущених значень, з чого можна зробити висновок що параметри кожної людини заповнені повністю. Функція також показала кількість ознак, їхні типи даних та кількість пам’яті, яка виділяється для зберігання даних. Як видно з рисунку 2.4 перші 23 параметра мають числові значення та останній параметр має тип даних «об’єкт». Ознака «Level», яка є нашою цільовою ознакою при прогнозуванні, містить у собі рівні ризику захворіти на жахливий рак легень (рис. 2.5).

```
print('Cancer Levels: ', df['Level'].unique())
```

```
Cancer Levels:  ['Low' 'Medium' 'High']
```

Рисунок 2.5 – Ознака «Level» та її значення

Продемонстрований рисунок 2.5 показує три унікальні класи параметру «Level» , які відповідають низькому, середньому і високому ризикам виникнення раку легень. Щоб під час моделювання не виникало проблем із розпізнаванням цих унікальних класів перетворимо їх у числову класифікацію (рис. 2.6).

```
# Replace "level" with Integer
print('\n')
print('Cancer Levels: ', df['Level'].unique())

# Replacing levels with int
df["Level"].replace({'High': 2, 'Medium': 1, 'Low': 0}, inplace=True)
print('Cancer Levels: ', df['Level'].unique())

print('\nColumns in dataframe: \n', df.columns)
```

```
Cancer Levels:  ['Low' 'Medium' 'High']
Cancer Levels:  [0 1 2]

Columns in dataframe:
 Index(['Age', 'Gender', 'Air Pollution', 'Alcohol use', 'Dust Allergy',
       'OccuPational Hazards', 'Genetic Risk', 'chronic Lung Disease',
       'Balanced Diet', 'Obesity', 'Smoking', 'Passive Smoker', 'Chest Pain',
       'Coughing of Blood', 'Fatigue', 'Weight Loss', 'Shortness of Breath',
       'Wheezing', 'Swallowing Difficulty', 'Clubbing of Finger Nails',
       'Frequent Cold', 'Dry Cough', 'Snoring', 'Level'],
      dtype='object')
```

Рисунок 2.6 – Заміна класів ознаки «Level» на числові

2.2 Розвідувальний аналіз даних

Для кращого розуміння розподілу даних проведемо детальний аналіз по кожній ознаці окремо. Рисунок 2.7 показує графіки типу QQ-plot і Boxplot, за допомогою такого типу графіків можна проаналізувати розподіл даних по необхідних ознаках, у даному випадку це «Вік».

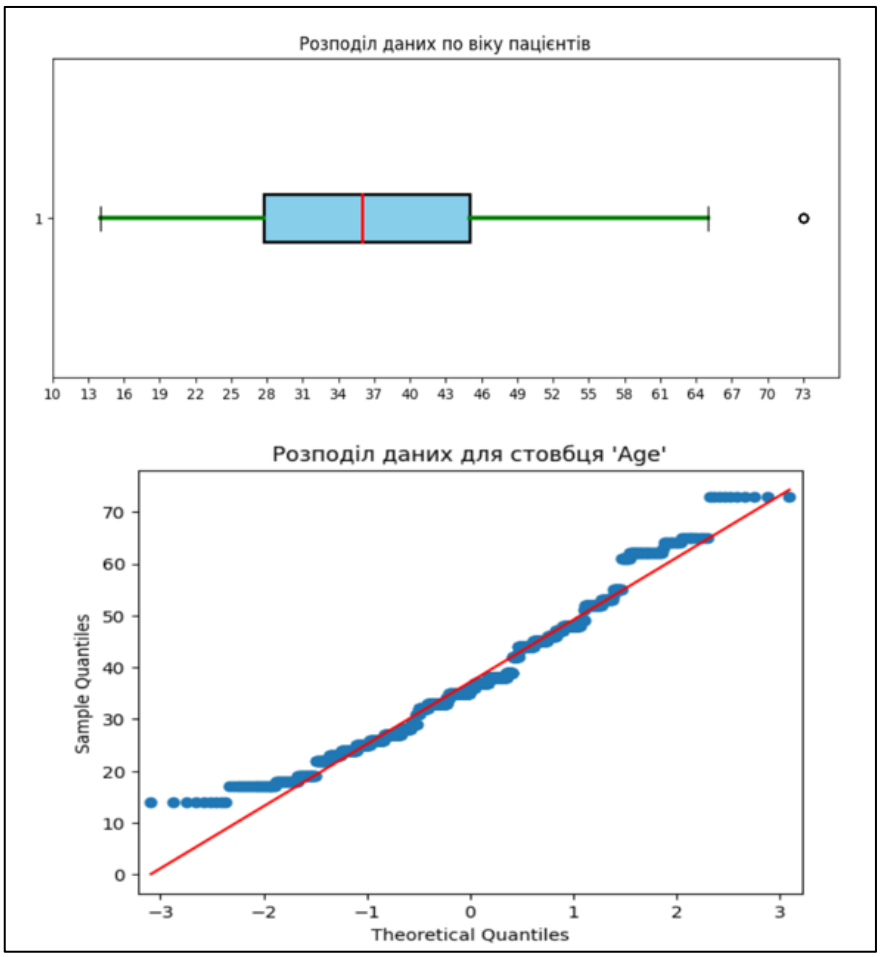

Рисунок 2.7 – Графіки розподілу ознаки «Вік»

Представлені графіки показують, що віковий розподіл пацієнтів в середньому розташований у категорії людей віком від 27 до 44 років, що вказує на те що більшість це пацієнти середнього віку. Проаналізувавши графік Boxplot бачимо нестачу даних між 65 роками до 73 років, що є характеристикою того що набір даних є малим для нормального розподілу по віковій категорії. Для того щоб покращити дані для сприйняття моделями машинного навчання обмежимо пацієнтів до віку від 18 до 60 років. Наявні ознаки датасету не мають значного впливу на дітей, тому що вони не піддаються впливу більшості цих ознак, та дослідження дитячого раку потрібно проводити окремо. Наявна прірва у даних між 60 і 70 роками може негативно вплинути на прогнозування деяких моделей, тому також було вирішено прибрати частину даних після 60 років. Також було прийняте рішення добавити нову ознаку, яка буде відображати класифікацію вікових

категорій від 18 до 29 років, від 29 до 45 та від 45 до 60, що буде дорівнювати категоріальним числам 1, 2 та 3 відповідно (рис. 2.8).

```
df1 = df1[(df1['Age'] >= 18) & (df1['Age'] <= 60)]

df1['Age'].unique()
df1['Age_group'] = pd.cut(df1['Age'], bins=[17, 29, 45 ,61], labels=[1, 2, 3])
df1.columns = [c.replace(" ", "_") for c in df1.columns]
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 900 entries, 0 to 999
Data columns (total 25 columns):
 #   Column                    Non-Null Count  Dtype
```

Рисунок 2.8 – Створення нової категоріальної ознаки за віком

Далі побудуємо матриці кореляції, які покажуть нам наскільки сильно чи навпаки слабко ознаки в наборі даних корелюються між собою. Для обчислення значень кореляції використовуватиметься функція corr() (рис. 2.9).

```
df_corr = df.corr(method='spearman')
df_corr
```

	Age	Gender	Air Pollution	Alcohol use	Dust Allergy	OccuPational Hazards	Genetic Risk	chronic Lung Disease	Balanced Diet
Age	1.000000	-0.165672	0.062103	0.139518	0.040532	0.054822	0.040005	0.108806	-0.0336
Gender	-0.165672	1.000000	-0.243384	-0.236563	-0.213486	-0.181904	-0.223501	-0.212435	-0.1018
Air Pollution	0.062103	-0.243384	1.000000	0.694256	0.640920	0.553041	0.654058	0.597733	0.49676
Alcohol use	0.139518	-0.236563	0.694256	1.000000	0.837587	0.851185	0.848291	0.750601	0.57775
Dust Allergy	0.040532	-0.213486	0.640920	0.837587	1.000000	0.875420	0.819746	0.691008	0.64739
OccuPational Hazards	0.054822	-0.181904	0.553041	0.851185	0.875420	1.000000	0.877435	0.859535	0.68111
Genetic Risk	0.040005	-0.223501	0.654058	0.848291	0.819746	0.877435	1.000000	0.791538	0.63745
chronic Lung Disease	0.108806	-0.212435	0.597733	0.750601	0.691008	0.859535	0.791538	1.000000	0.62432
Balanced Diet	-0.033680	-0.101899	0.496763	0.577756	0.647398	0.681119	0.637459	0.624321	1.00000
Obesity	0.060273	-0.118361	0.579275	0.624148	0.680932	0.712281	0.702909	0.588089	0.63890
Smoking	0.040817	-0.191490	0.358466	0.515559	0.385391	0.465207	0.479731	0.518253	0.66040
Passive Smoker	0.015009	-0.155438	0.524948	0.490288	0.582508	0.522784	0.535195	0.524457	0.71850
Chest Pain	0.018189	-0.211177	0.571273	0.693858	0.689782	0.785827	0.817352	0.788712	0.79429
Coughing of Blood	0.083885	-0.151523	0.552360	0.607764	0.598343	0.618346	0.545186	0.591911	0.68367

Рисунок 2.9 – Обчислення кореляції функцією corr()

Зручним та практичним інструментом для аналізу кореляційних значень є теплові карти. Зручність та практичність таких карт досягається за рахунок використання кольорів та їхньої насиченості на противагу числам. Це дає змогу легко оцінити залежності між ознаками, чим вищою є насиченість кольору тим більша залежність. Продемонструємо їх на рисунках 2.10 та 2.11.

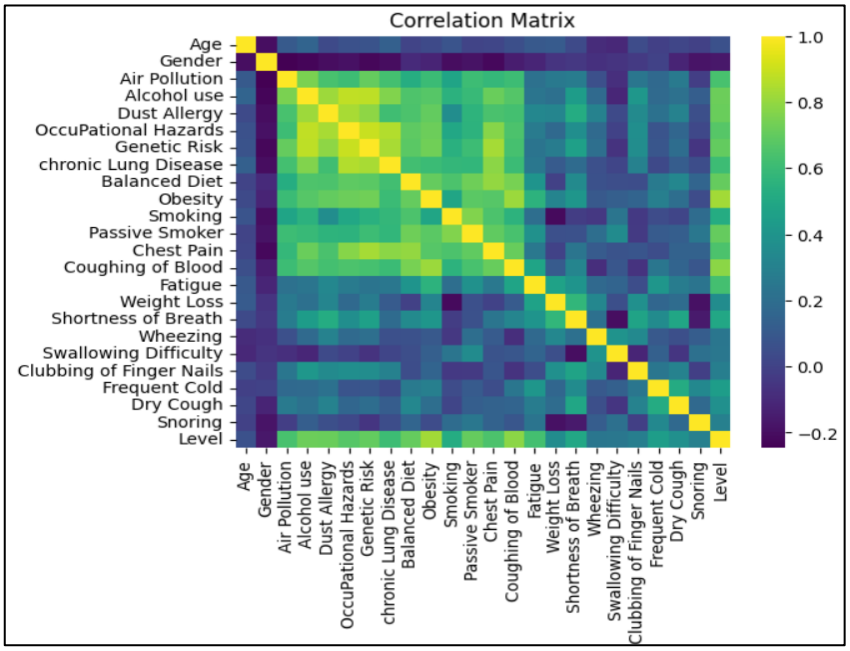

Рисунок 2.10 – Кореляційна теплова карта

Створення подібних візуальних карт за допомогою існуючих програмних модулів є цілком не складним. Їх зручність використання також продиктована тим, що для оцінки взаємозв'язків між параметрами не потрібен детальний та довгий числовий аналіз, для цього достатньо лише один раз поглянути на карту. Це швидкий та інтуїтивно зрозумілий спосіб зрозуміти взаємозв'язки маж ознаками, а також продемонструвати їх іншим [20].

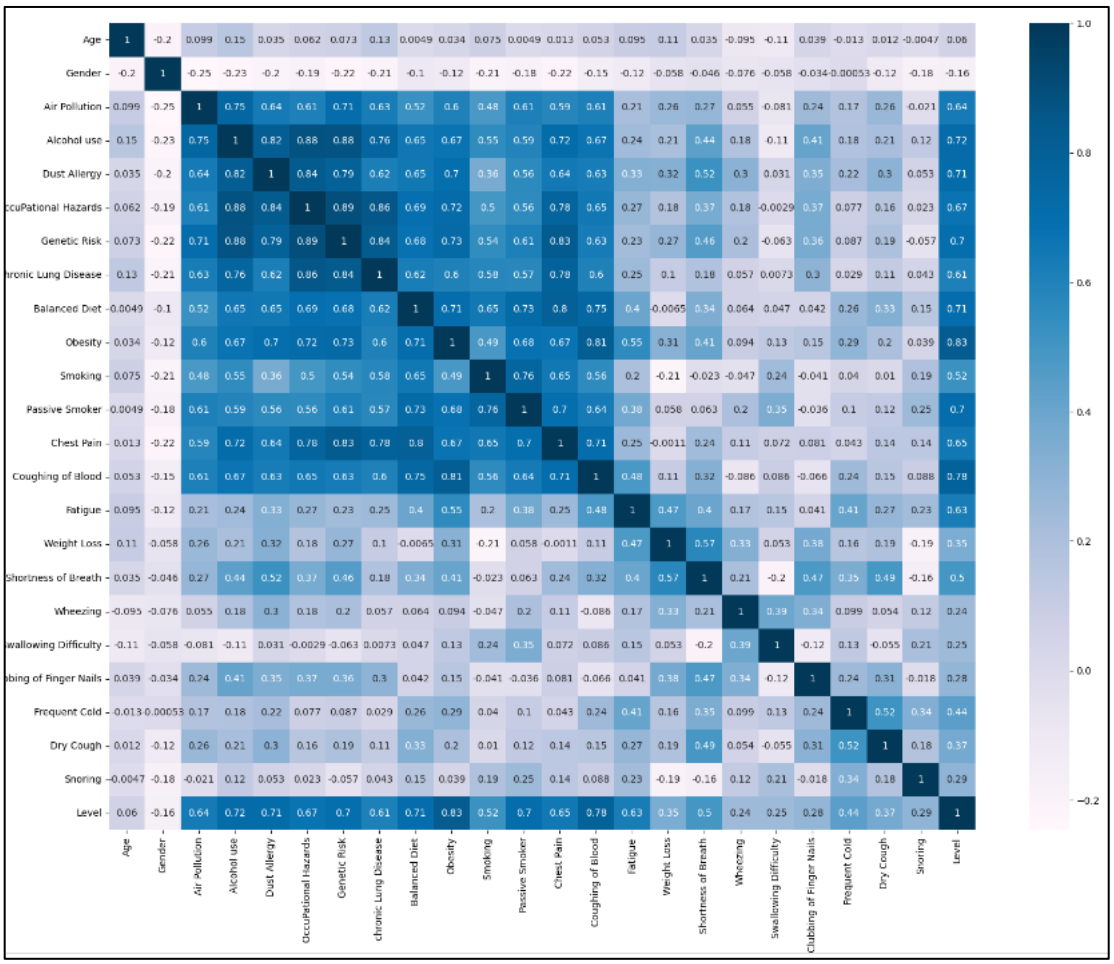

Рисунок 2.11 – Теплова карта залежності ознак

Аналізуючи теплову карту залежності ознак, зображену на рисунку 2.11, прослідковується високий рівень кореляція між основними ознаками такими як «Куріння», «Пасивне куріння», «Вживання алкоголю», «Ожиріння» та «Алергія на пил», які відображені у лівому куті. Високі показники кореляції пояснюються тим що ці параметри можуть бути співставні, наприклад коли людина довгий період часу являється пасивним курцем, то у неї тютюн може викликати залежність, через що можливість появлення такої шкідливої звички як куріння збільшується.

Провівши аналіз залежності між цільовою ознакою «Level» та іншими, особливо на фоні виділяється зв’язок із «Ожиріння», «Вживання алкоголю» та «Генетичні ризики», де оцінка цих параметрів коливається в районі від 0.7 до 0.83.

Далі спробуємо побудувати графік із рівень розподілу даних відповідно до віку пацієнтів та їх ризиків захворіти (рис. 2.12).

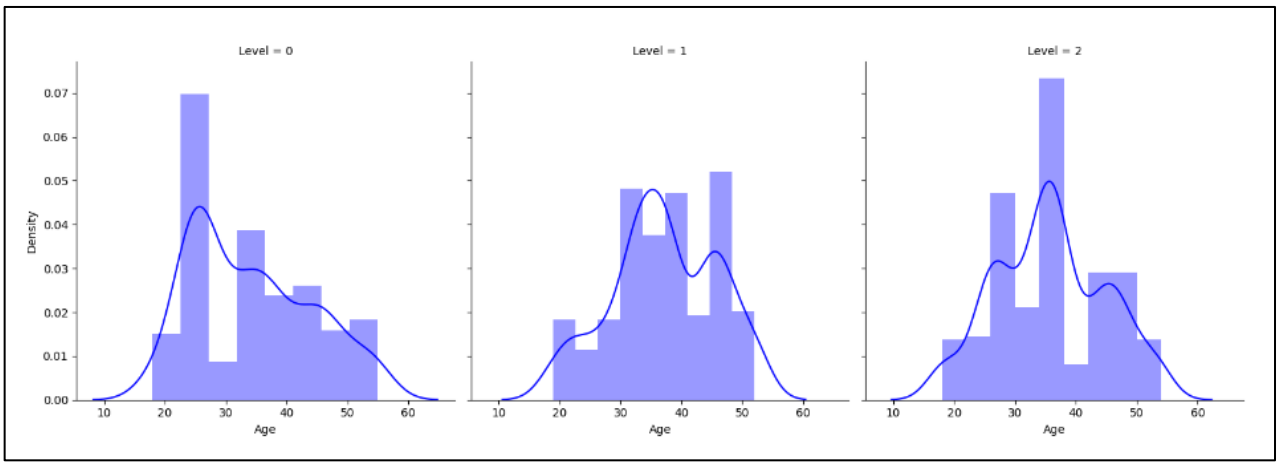

Рисунок 2.12 – Графік кількісного розподілу людей за віком відносно рівнів ризику

З графіків можна помітити, що серед людей віком від 25 років переважає саме низький рівень ризику, середній рівень ризику розподілений досить однаково серед пацієнтів віком від 30 до 45 років, тоді як високий рівень ризику частіше спостерігається у людей близько 35 років. Ця інформація свідчить про те, що наявні дані розподілені на зразок до світового розподілу захворювання онкологією легень. Не є секретом що молоді люди хворіють на рак легень у меншій кількості, адже молодий імунітет більш стійкий та ліпше протистоїть хворобі та набагато успішніше може боротися із мутаціями. Не варто забувати про термін впливу різних факторів на організм. До прикладу дві людини, які почали курити у свої 20 років будуть мати різні рівні ризику, адже у першої людини стаж куріння буде всього лише 5 років бо їй 25 років на даний момент, а у другої людини стаж куріння буде вимірюватись уже у 25 років, адже їй на даний момент 45 років, тому для них обох вплив куріння на шанси виникнення раку легень відрізняються, адже легені людини це не вічний механізм, і під постійними та тривалими навантаженнями він все частіше буде давати збій, тому краще за все відмовитись за період свого життя від цієї шкідливої звички. Це також стосується усіх параметрів, для яких термін впливу збільшує ризики захворіти.

Побудуємо графік кількісного розподілу спираючись на «Стать» як основний параметр (рис. 2.13).

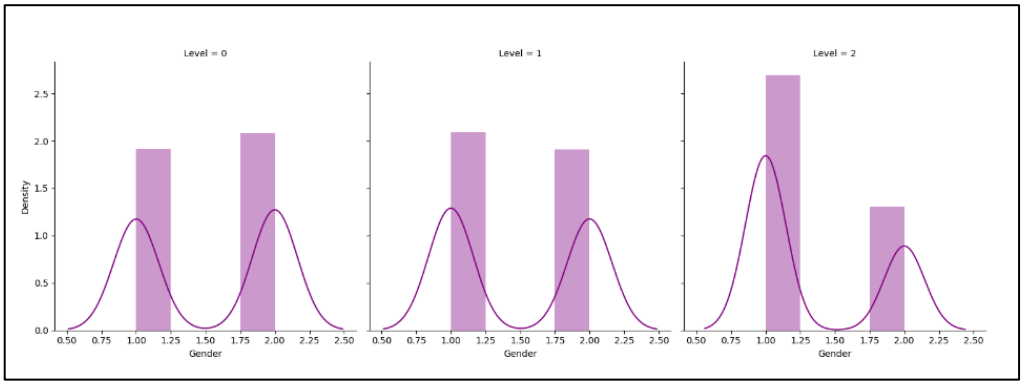

Рисунок 2.13 – Розподіл людей за ознакою «Стать» гендерною ознакою відносно рівнів ризику

Аналізуючи візуалізацію за гендерною ознакою, можна зробити висновок, що низький ризик та середній кількісно розподілені рівномірно. Натомість у категорії високого ризику спостерігається помітна перевага у кількості чоловіків. Це може свідчити про більшу вразливість чоловіків до факторів, що підвищують ймовірність розвитку захворювання. Таке явище пояснюється тим, що чоловіки частіше знаходяться під впливом поганих звичок, в тому числі і куріння, а також вони частіше працюють у галузях із підвищеним негативним впливом шкідливих речовин де вони контактують із хімікатами, металургійними відходами, газами та пестицидами. Цю тенденцію наочно демонструє графік на рисунку 2.14.

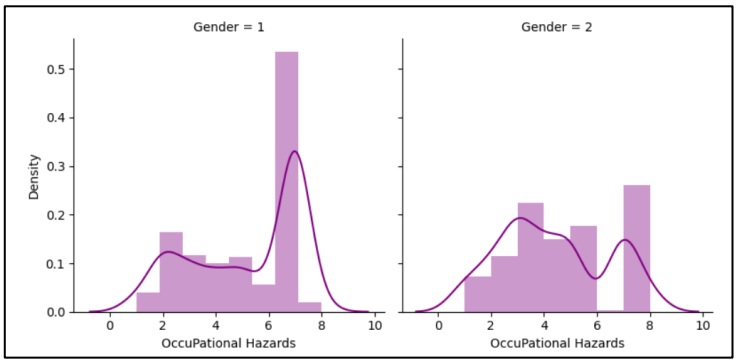

Рисунок 2.14 – Кількісний розподіл чоловіків та жінок відносно «Професійних ризиків»

Розглянемо як розподілені дані за іншими ознаками, що входять до набору даних (рис. 2.15-2.28).

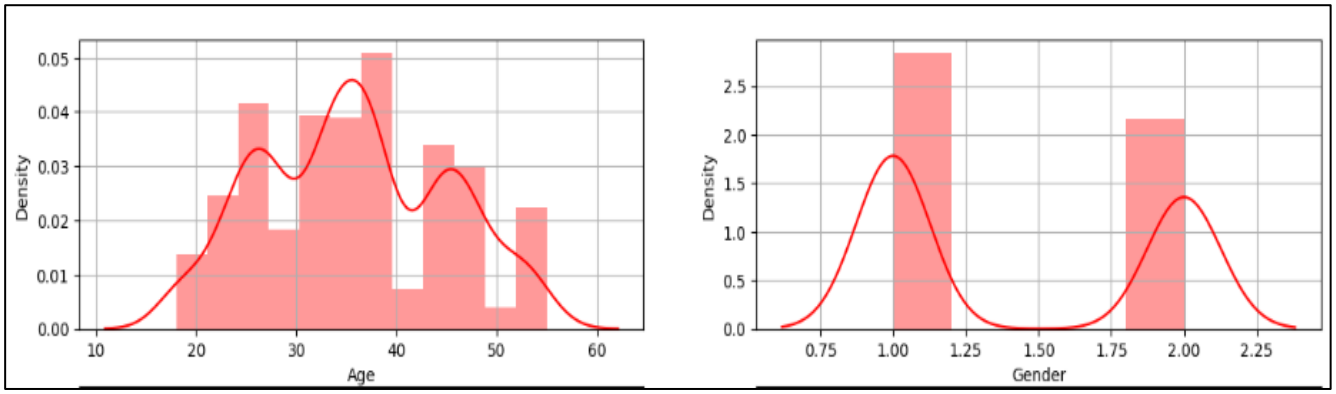

Рисунок 2.15 – Графіки розподілу параметрів «Вік» і «Стать»

На діаграмі з рисунку 2.15 спостерігається нерівномірне представлення пацієнтів за віковими групами. Найбільша кількість записів припадає на осіб віком від 30 до 40 років, тоді як категорія 40-50 років майже не представлені. Така особливість пояснюється обмеженою кількістю записів у наборі даних.

Що стосується розподілу за статтю, графік показує, що чоловіків трохи більше, ніж жінок, але різниця не є значною.

Відобразимо кількісний розподіл за такими ознаками як рівень забрудненого повітря та рівень вживання алкоголю (рис. 2.16).

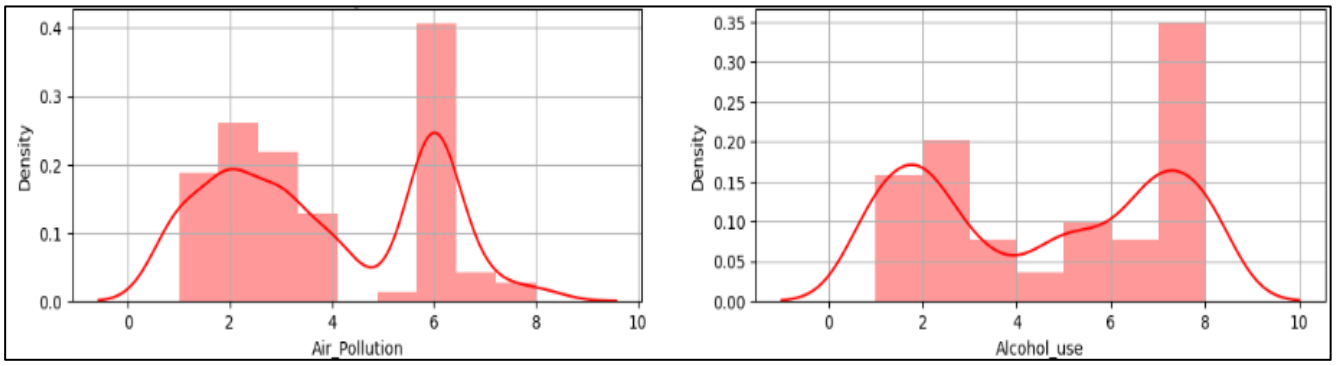

Рисунок 2.16 – Графіки розподілу параметрів «Забруднення повітря» і «Вживання алкоголю»

На рисунку 2.16 зображено, як розподіляються люди залежно від ознаки «Забруднення повітря» у їхніх районах. Найбільша частка спостережень припадає на підвищені рівні, низькі та середні значення приблизно однакові, а високі фіксуються нечасто. Що стосується алкоголю, низький рівень споживання зустрічається але не часто, а високі рівні реєструються значно частіше.

Побудуємо графіки розподілу для параметрів «Пилової алергії» і «Професійні ризики» (рис. 2.17).

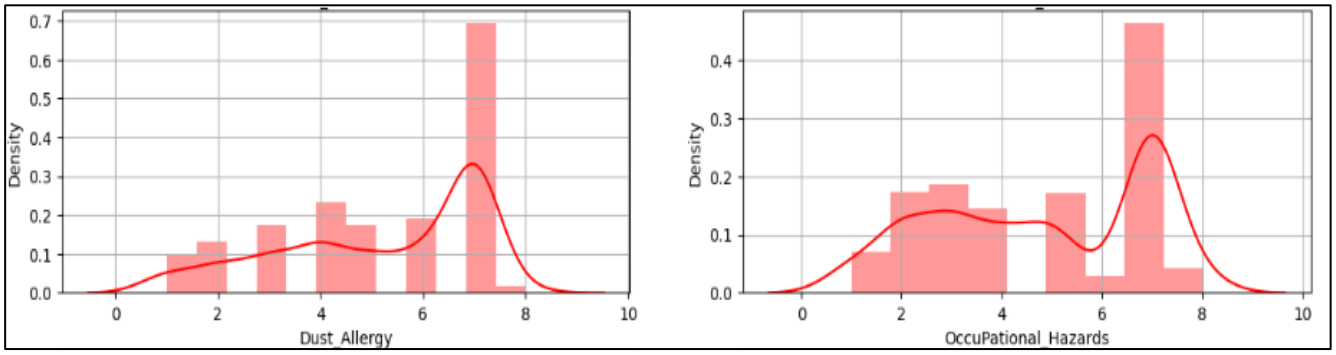

Рисунок 2.17 – Графіки розподілу параметрів «Пилової алергії» і «Професійні ризики»

Аналізуючи рисунок 2.17 відслідковується подібність цих двох графіків, яка полягає у низьких малих значеннях та різкому стрибку кількості при високих рівнях.

Далі оформимо графіки розподілу для параметрів «Генетичні ризики» і «Хронічні захворювання легень» (рис. 2.18).

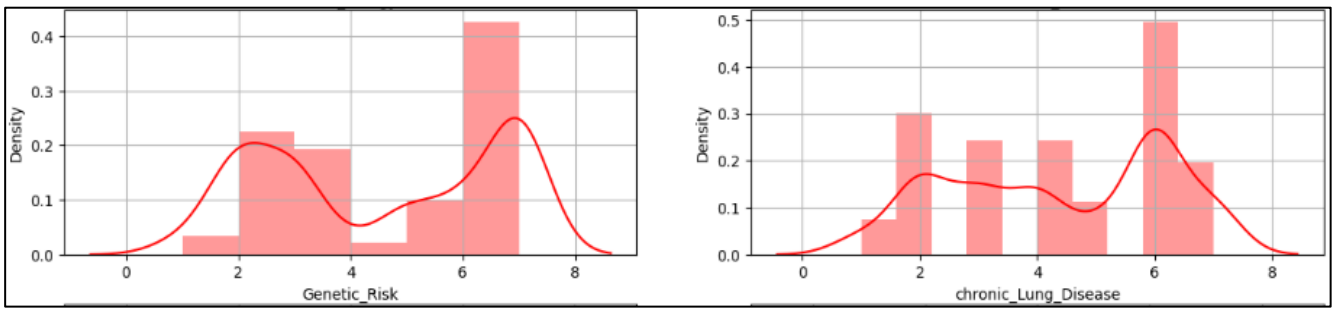

Рисунок 2.18 – Графіки розподілу параметрів «Генетичні ризики» і «Хронічні захворювання легень»

Ліва діаграма зображена на рисунку 2.18 показує перевагу у кількості людей із високим рівнем генетичних ризиків, але помірні рівні ризику також мають свою вагу.

Ознака «Хронічні захворювання легень» має рівномірний розподіл при низьких значеннях та збільшену кількість на високих значеннях.

Далі проведемо побудову графіків розподілу параметрів «Збалансована дієта» і «Ожиріння» (рис. 2.19).

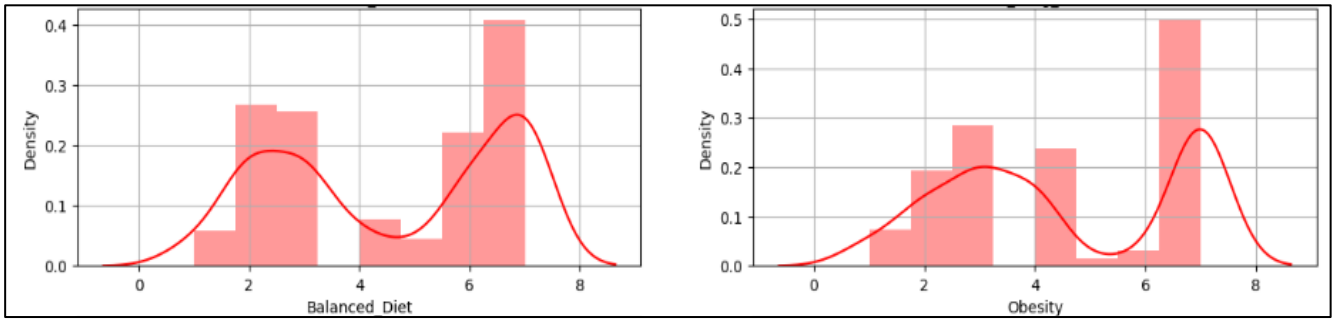

Рисунок 2.19 – Графіки розподілу параметрів «Збалансована дієта» і «Ожиріння»

Лівий графік на рисунку 2.19 демонструє значну кількість людей, які мають проблеми із харчуванням та дієтою. Також він дещо корелюється з правим графіком із ознакою «Ожиріння», тобто можна зробити висновок що значна частина записів зроблена про людей із харчовими проблемами.

Побудуємо графіки розподілу для таких параметрів як от «Куріння» та «Пасивне куріння» (рис. 2.20).

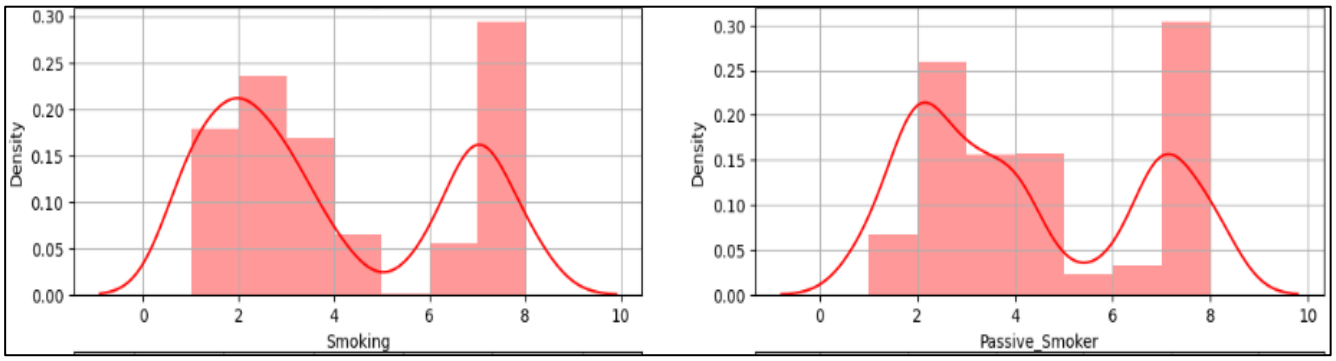

Рисунок 2.20 – Графіки розподілу параметрів «Куріння» і «Пасивне куріння»

Рисунок 2.20 демонструє діаграми на яких можна відслідкувати високий відсоток людей які не піддаються впливу куріння чи пасивного куріння, але все ж половина із цього відсотка кількості належить пацієнтам, які є затятими курцями. Правий графік дещо схожий на графік впливу звичайного куріння, але пацієнтів із низьким рівнем впливу пасивного куріння дещо менше.

Побудуємо графіки розподілу для таких параметрів як «Біль у грудях» і «Кашляння кров'ю» (рис. 2.21).

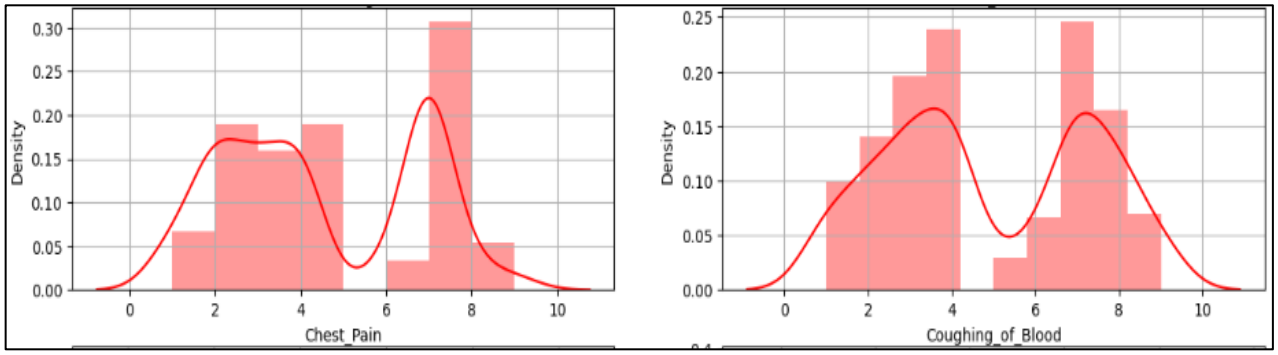

Рисунок 2.21 – Графіки розподілу параметрів «Біль у грудях» і «Кашляння кров'ю»

Проаналізувавши побудовані графіки на рисунку 2.21 можна зробити висновок що дані мають низьку наповненість для середнього рівня «Болю у грудях» та «Кашляння кров'ю», але інші рівні мають хорошу кількість значень.

Далі оформимо графіки розподілу для таких параметрів як «Втома» та «Втрата ваги» (рис. 2.22).

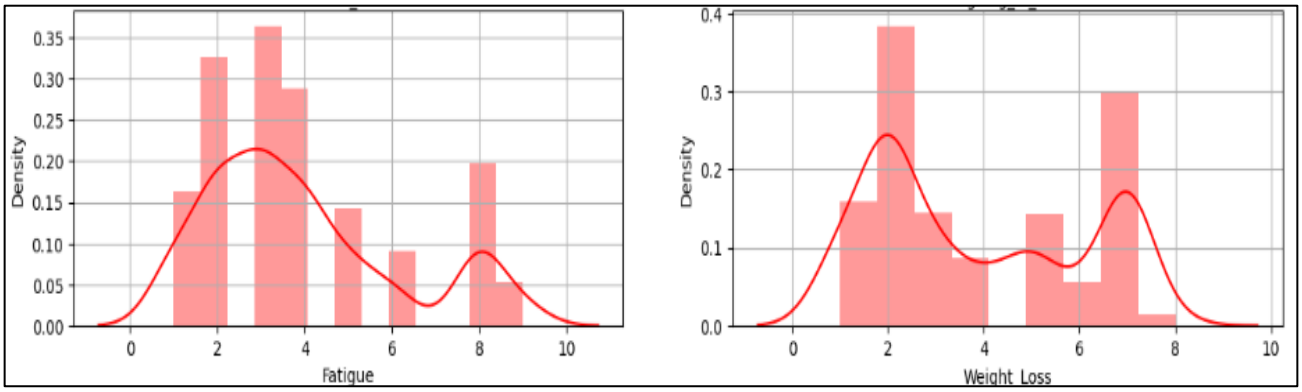

Рисунок 2.22 – Графіки розподілу параметрів «Втома» і «Втрата ваги»

Відповідно до рисунку 2.22 можна зробити аналіз, і дійти висновку що кількість людей, які перебувають на низьких показниках втоми переважає над високими рівнями.

На графіку де продемонстрований рівень втрати ваги бачимо перевагу у кількості для оцінок 2 та 7.

Побудуємо графіки розподілу опираючись на параметри «Задишки» і «Хрипіння» (рис. 2.23).

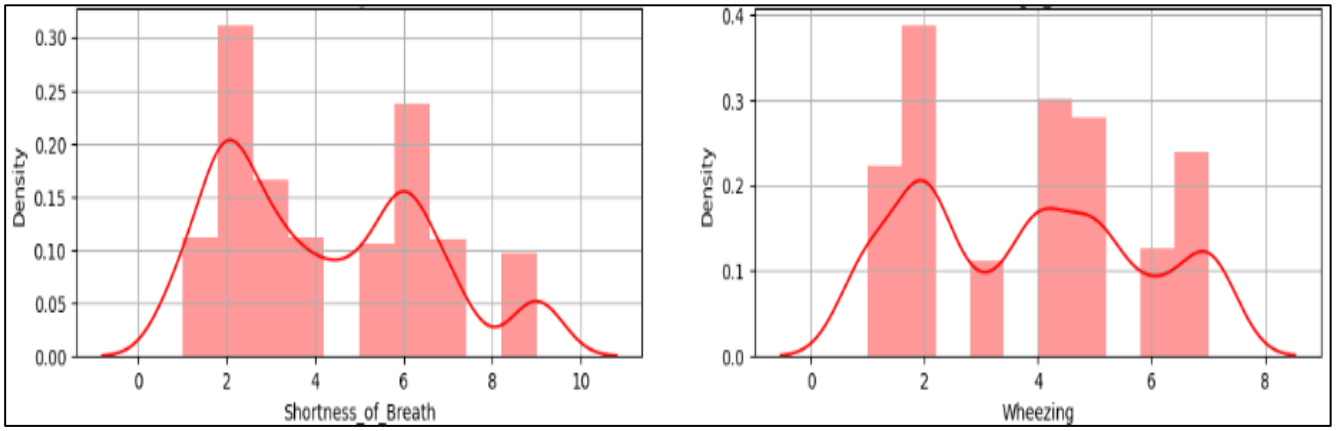

Рисунок 2.23 – Графіки розподілу параметрів «Задишка» і «Хрипіння»

Рисунок 2.23 демонструє що розподіл лівого графіку параметру «Задишка» приблизно однаковий, але між усіх оцінок вирізняються 2 та 6, яких більше порівняно із іншими. В противагу цьому правий графік параметру «Хрипіння» має хвилеподібний розподіл де на вершинах сконцентрована кількість для оцінок 2, 4 та 7.

Наступним кроком побудуємо графік кількісного розподілу спираючись на параметри «Труднощі при ковтанні» і «Булини нігтів» (рис. 2.24).

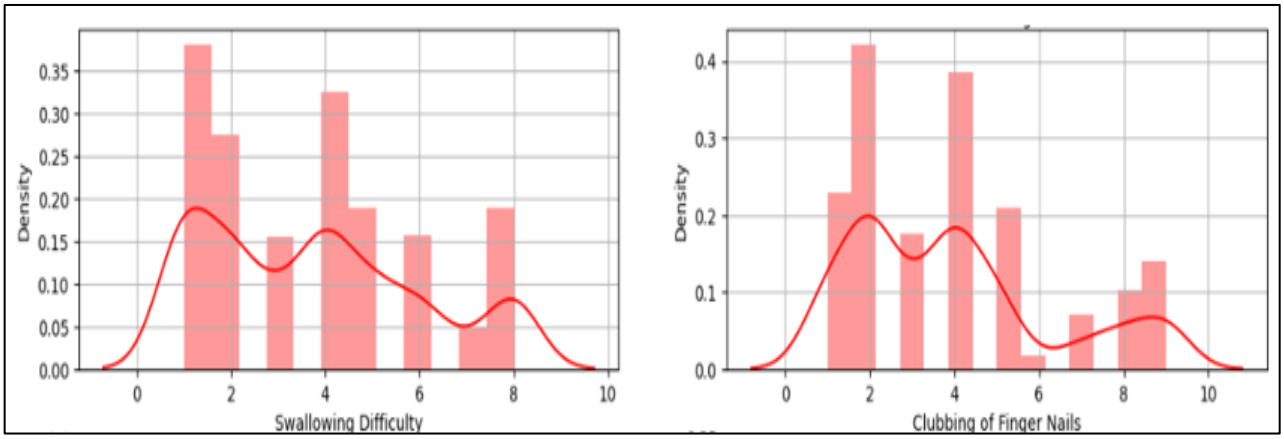

Рисунок 2.24 – Графіки розподілу параметрів «Труднощі при ковтанні» і «Булини нігтів»

Лівий та правий графік на рисунку 2.24 корелюються між собою, тобто дуже сильно схожі один на одного та повторюють тенденцію один одного. Аналізуючи варто відмітити велику кількість пацієнтів із низькими показниками цих двох ознак, та на противагу низьку кількість пацієнтів при високих оцінках впливу.

Слід звернути увагу що так звані булини нігтів це стан, при якому нігтьова пластина на руках поступово заокруглюються немов би збільшуючись в об'ємі. Високі рівні даного показника співвідносяться при дуже сильному заокругленні і збільшенні. Дана проблема є дуже поширеною серед пацієнтів, які страждають саме легеневими хворобами.

Далі оформимо графіки розподілу для параметрів «Частота захворювання грипом» і «Сухий кашель» (рис. 2.25).

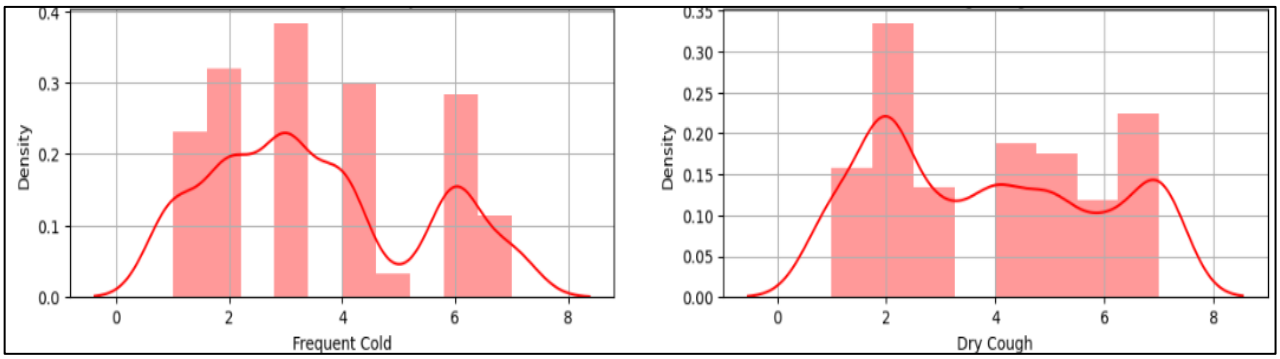

Рисунок 2.25 – Графіки розподілу параметрів «Частота захворювання грипом» і «Сухий кашель»

Провівши аналіз графіків зображених на рисунку 2.25 можемо дійти висновку про нормальний розподіл параметру «Частота захворювання грипом», тобто наскільки сильно його імунна система вразлива для грипу, із піком в центральній частині графіку та спадами на початку та в кінці.

На сусідньому графіку кількість даних між усіма оцінками розбита порівну окрім оцінки 2, яка суттєво обганяє інші по кількості.

Наступним кроком побудуємо графік кількісного розподілу спираючись на параметр «Хропіння» (рис. 2.26).

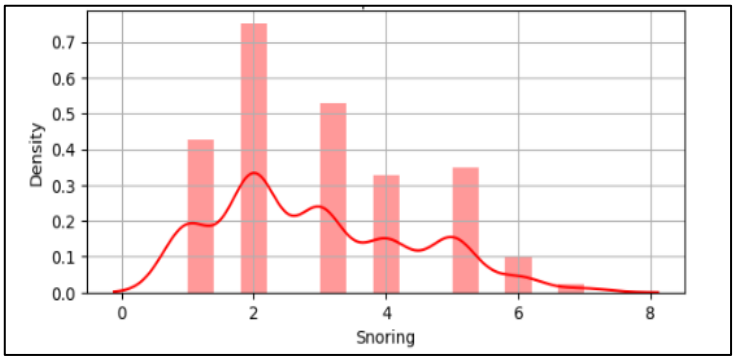

Рисунок 2.26 – Графіки розподілу параметра «Хропіння»

Аналізуючи рисунок 2.26 можна сказати що більшість людей має низький рівень хропіння, та не страждають від цього параметру.

Окремо потрібно провести детальніший аналіз параметру «Level», оскільки він є цільовою ознакою, і за сумісництвом найважливішою з усіх, адже саме його ми будемо передбачати під час побудови та тренування моделей. Відобразимо побудований розподіл за унікальними значеннями ознаки рівня ризику, а саме низький, середній та високий (рис. 2.27).

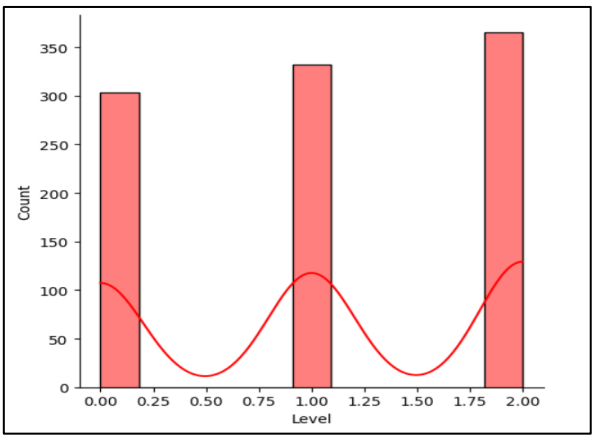

Рисунок 2.27 – Графік розподілу параметра «Level»

Далі на рисунку 2.28 побудуємо Pie Chart діаграму, із відсотковими значеннями розподілу.

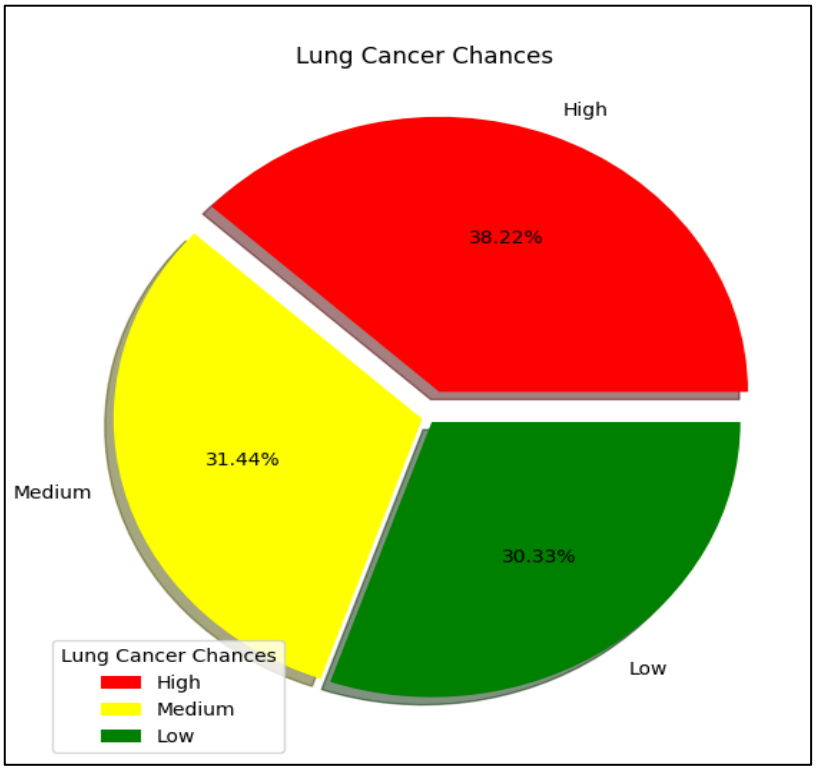

Рисунок 2.28 – Відсоткова кругова діаграма розподілу цільової ознаки «Level»

Аналізуючи дану кругову діаграму на рисунку 2.28 відслідковуємо значення у 38.2%, 31.4% та 30,3%, це означає що кількість записів для цільової ознаки є рівномірною та дозволяє користуватися такими метриками як F1_Score та R2_Score для оцінки точності моделей. Враховуючи те, що була поставлена задача саме

класифікації, основний акцент під час аналізу оцінок точності буде ставитись саме на метрику F1_Score.

Під час виконання розвідувального аналізу та побудови діаграм, було використано ноутбук користувача «Subhajeet Das», як основний приклад [21].

2.3 Конструювання ознак (FE)

Конструювання ознак (Feature engineering) — це процес, під час якого на основі знань про предметну область створюють нові характеристики даних, що допомагають алгоритмам машинного навчання працювати точніше та ефективніше. Цей етап вважається одним із найважливіших у побудові моделей, хоча він зазвичай потребує багато часу й уваги. Частину ручної роботи сьогодні можна замінити засобами автоматичного створення ознак.

Ознака це певна властивість або параметр, спільний для всіх об’єктів у наборі даних, який використовується для аналізу чи прогнозування. Фактично будь-яка характеристика може стати ознакою, якщо вона допомагає моделі краще зрозуміти задачу.

Автоматизація створення ознак стала перспективним напрямом наукових досліджень. У 2015 році група з MTI запропонувала метод Deep Feature Synthesis і продемонструвала його ефективність у змаганнях із Data Science. Алгоритм перевершив понад половину команд-учасників. На основі цього підходу була створена відкрита бібліотека Featuretools. Пізніше з’явилися й інші інструменти, зокрема OneBM від IBM та ExploreKit з Університету Берклі.

За словами дослідників IBM, подібні технології дозволяють значно скоротити час, який спеціалісти витрачають на підготовку даних, оскільки вони можуть швидко випробовувати різні ідеї. Крім того, автоматизація дає можливість навіть користувачам без глибоких знань у сфері Data Science отримувати корисну інформацію з даних із мінімальними витратами часу та ресурсів [22].

Для покращення наших моделей було вирішено застосувати автоматичне генерування нових ознак на основі тих, які корелюються між собою найбільше, а саме «Забруднення повітря», «Куріння», «Пасивне куріння», «Генетичні ризики», «Ожиріння», «Вживання алкоголю» та «Професійні ризики». Проведемо аналіз кількісного розподілу деяких із низ відповідно до унікальних значень класів цільової ознаки, для цього будемо використовувати теплові матриці, які за допомогою насиченості кольору показують найбільшу кількість значень та співпадінь (рис. 2.29-2.31).

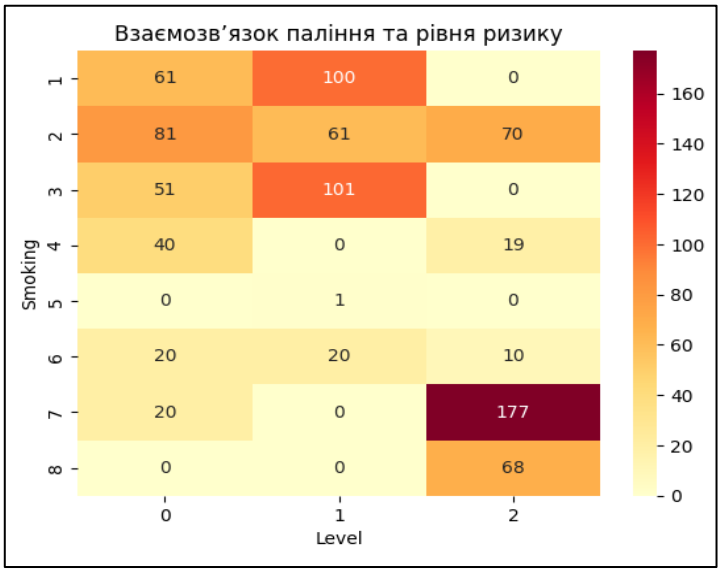

Рисунок 2.29 – Кількісний розподіл за кожною оцінкою рівня куріння відповідно до унікальних класів цільової ознаки «Level»

Проаналізувавши теплову матрицю можна зробити висновок що вибірка даних показує, що більшість даних припадає на малі оцінки впливу від 1 до 3, також значна частина даних відповідає оцінці 7 , та вони усі припадають на високий рівень ризику, для середнього ризику даних немає, а при низькому ризику даних небагато. Такий розподіл показує тенденцію де курці із великим стажем куріння з більшими шансами отримають статус високого ризику захворіти на рак легень.

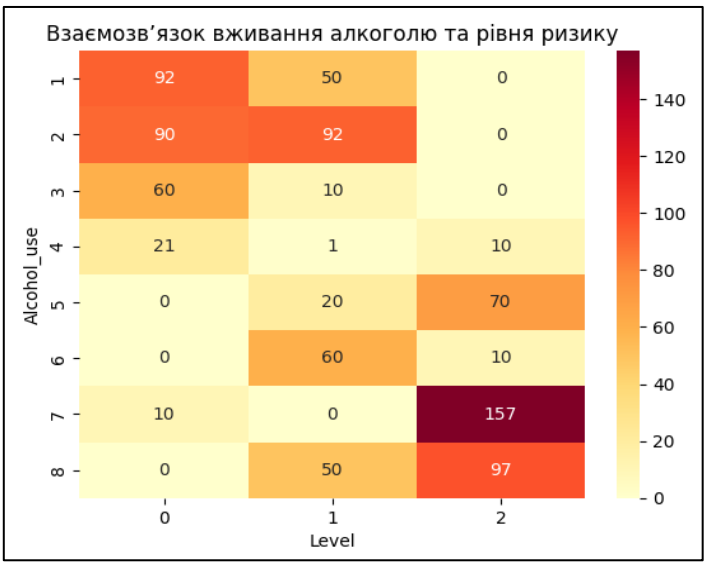

Рисунок 2.30 – Кількісний розподіл за кожною оцінкою рівня вживання алкоголю відповідно до унікальних класів цільової ознаки «Level»

На рисунку 2.30 бачимо діагональний розподіл на тепловій матриці по рівню вживання алкоголю, тобто для низьких ризиків захворіти на рак легень дані знаходяться на низьких рівнях оцінки впливу ознаки, а при високих оцінках навпаки дані уже зосередженні на високих ризиках.

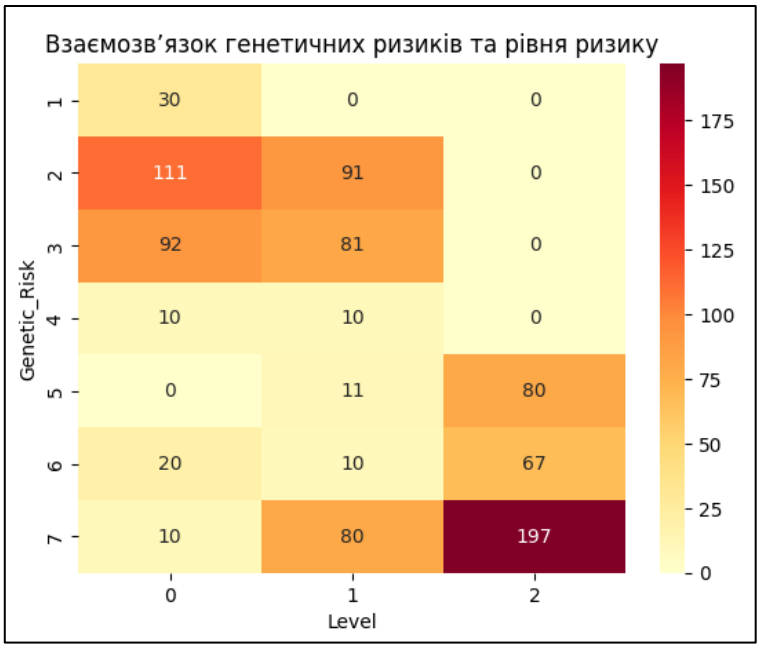

Рисунок 2.31 – Кількісний розподіл за кожною оцінкою рівня генетичних ризиків відповідно до унікальних класів цільової ознаки «Level»

Аналізуючи рисунок 2.31, можемо бачити схожу діагональну тенденцію, як і у попередніх рисунках, але бачимо що все ж середні рівні ризику розподілені більшою частиною на низьких оцінках впливу генетичних ризиків, і меншою на високих.

Для генерування ознак використаємо метод PolynomialFeatures, він використовується для підвищення якості моделей машинного навчання шляхом нелінійного розширення простору ознак. Суть методу полягає у створенні нових змінних, які є добутками або степенями існуючих ознак, що дозволяє моделі враховувати складніші взаємозв'язки між параметрами. Такий підхід є особливо ефективним у випадках, коли залежність між вхідними змінними та цільовою змінною є нелінійною, а звичайні лінійні моделі не здатні адекватно її відобразити. Перевага PolynomialFeatures полягає в тому, що він зберігає зрозумілу структуру та логіку роботи моделі, водночас суттєво підвищуючи її гнучкість. Таким чином, застосування цього алгоритму дає змогу покращити точність прогнозування та відобразити складні взаємодії між ознаками в межах простих моделей, таких як лінійна або логістична регресія та паралельно допомагає полегшити пошук закономірностей для складних моделей. На рисунку 2.32 приведений приклад застосування методу на наших ознаках та даних.

```
from sklearn.preprocessing import PolynomialFeatures

cols = ['Air_Pollution', 'Smoking',
        'Passive_Smoker', 'Genetic_Risk', 'Obesity', 'Alcohol_use', 'OccuPational_Hazard
s']
X = df1[cols]

poly = PolynomialFeatures(degree=2, interaction_only=True, include_bias=False)
X_poly = poly.fit_transform(X)

poly_features = poly.get_feature_names_out(cols)

interaction_cols = [f for f in poly_features if '_' in f]

X_poly_df = pd.DataFrame(X_poly, columns=poly_features, index=df1.index)
X_poly_df = X_poly_df[interaction_cols]
df1_poly = pd.concat([df1, X_poly_df], axis=1)
```

Рисунок 2.32 – Генерування нових ознак методом PolynomialFeatures

Після застосування методу було отримано нові ознаки у кількості двадцяти штук (рис. 2.33 – 2.34).

Passive_Smoker Genetic_Risk	Passive_Smoker Obesity	Passive_Smoker Alcohol_use	Passive_Smoker OccuPational_Hazards	Genetic_Risk Obesity	Genetic_Risk Alcohol_use	Genetic_Risk OccuPational_Haz
6.0	8.0	8.0	8.0	12.0	12.0	12.0
15.0	21.0	15.0	15.0	35.0	25.0	25.0
42.0	49.0	49.0	49.0	42.0	42.0	42.0
49.0	49.0	56.0	49.0	49.0	56.0	49.0
15.0	21.0	15.0	15.0	35.0	25.0	25.0
...	...	...	...	...	...	...
56.0	56.0	56.0	56.0	49.0	49.0	49.0
56.0	56.0	64.0	56.0	49.0	56.0	49.0
15.0	21.0	15.0	15.0	35.0	25.0	25.0
49.0	49.0	56.0	49.0	49.0	56.0	49.0
15.0	21.0	15.0	15.0	35.0	25.0	25.0

Рисунок 2.33 – Результати генерування нових ознак

```
25  Air_Pollution Smoking                900 non-null    float64
26  Air_Pollution Passive_Smoker         900 non-null    float64
27  Air_Pollution Genetic_Risk           900 non-null    float64
28  Air_Pollution Obesity                900 non-null    float64
29  Air_Pollution Alcohol_use            900 non-null    float64
30  Air_Pollution OccuPational_Hazards   900 non-null    float64
31  Smoking Passive_Smoker               900 non-null    float64
32  Smoking Genetic_Risk                 900 non-null    float64
33  Smoking Alcohol_use                  900 non-null    float64
34  Smoking OccuPational_Hazards         900 non-null    float64
35  Passive_Smoker Genetic_Risk          900 non-null    float64
36  Passive_Smoker Obesity               900 non-null    float64
37  Passive_Smoker Alcohol_use           900 non-null    float64
38  Passive_Smoker OccuPational_Hazards  900 non-null    float64
39  Genetic_Risk Obesity                 900 non-null    float64
40  Genetic_Risk Alcohol_use             900 non-null    float64
41  Genetic_Risk OccuPational_Hazards    900 non-null    float64
42  Obesity Alcohol_use                  900 non-null    float64
43  Obesity OccuPational_Hazards         900 non-null    float64
44  Alcohol_use OccuPational_Hazards     900 non-null    float64
```

Рисунок 2.34 – Результати генерування нових ознак

Отже застосування методу PolynomialFeatures дало необхідний результат, і вдалось отримати двадцять нових ознак, які будуть доповнювати уже наявні та допоможе покращити точність прогнозування моделей.

2.4 Висновки

У цьому розділі виконано процес попередньої підготовки даних, що є необхідним етапом для забезпечення коректності й ефективності подальшого моделювання. Здійснено очищення датасету шляхом видалення неінформативних полів («Індекс», «ID пацієнта»), а також перетворення категоріальних значень цільової змінної «Level» у числовий формат, що забезпечило можливість коректного застосування алгоритмів машинного навчання.

Проведено розвідувальний аналіз даних із використанням сучасних методів візуалізації, зокрема теплових карт і графічних представлень кількісних показників. Аналіз розподілу ознак підтвердив, що більшість пацієнтів з вибірки належать до середнього віку та зазнають впливу низки факторів ризику, таких як ожиріння, куріння та споживання алкоголю. Дослідження цільової ознаки показало рівномірний розподіл між усіма її унікальними класами, що є сприятливою умовою для навчання моделей без суттєвого дисбалансу.

Крім того, проаналізовано кореляційні зв’язки між ознаками з метою визначення найбільш значущих чинників впливу на цільову змінну. На основі отриманих результатів виконано створення нових похідних ознак, що сприятиме підвищенню якості функціонування моделей у подальших етапах дослідження.

3 РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

3.1 Розроблення інформаційної технології

Для розв’язання поставленої задачі, а саме розробки інформаційної технології, яка буде виконувати функцію передбачення ризиків захворіти на рак легень, вимагається сформувати сукупність методів машинного навчання, які тренуються на підготовлених даних та проходять тестування. Щоб вирішити цю задачу було обрано такі методи машинного навчання як: Random Forest, Extra Trees, Decision Tree, AdaBoost ,XGBoost, LGBM, Logistic Regression, k-NN, SVM та Multi-Layer Perceptron.

Щоб налаштувати моделі необхідно провести тюнінг, для цього було вирішено використовувати методи GridSearchCV та RandomizedSeaechCV, які зарекомендували себе як ефективні, прості та зрозумілі. Для того щоб інформаційна технологія була якісною, потрібно щоб моделі давали якомога кращі результати, за умови якщо результат моделі не буде відповідати вимогам, будуть змінюватись вхідні параметри моделей до отримання прийнятних результатів.

Наступним кроком результати по кожній моделі будуть порівнюватись, потім буде обрано найкращу з них, тобто найбільш якісну із високими результатами точності як на тренувальних такі на тестових даних. Після цього буде отримано результати дослідження шляхом аналізу впливовості ознак найліпшої моделі, які покажуть на які саме ознаки потрібно звертати увагу в першу чергу при профілактиці захворювання. Аналіз впливовості ознак буде розраховуватись за допомогою відповідних функцій де для кожної ознаки буде дано оцінку впливу на передбачення моделі.

Побудуємо блок-схему алгоритму роботи інформаційної технології передбачення ризиків захворіти на рак легень (рис. 3.1).

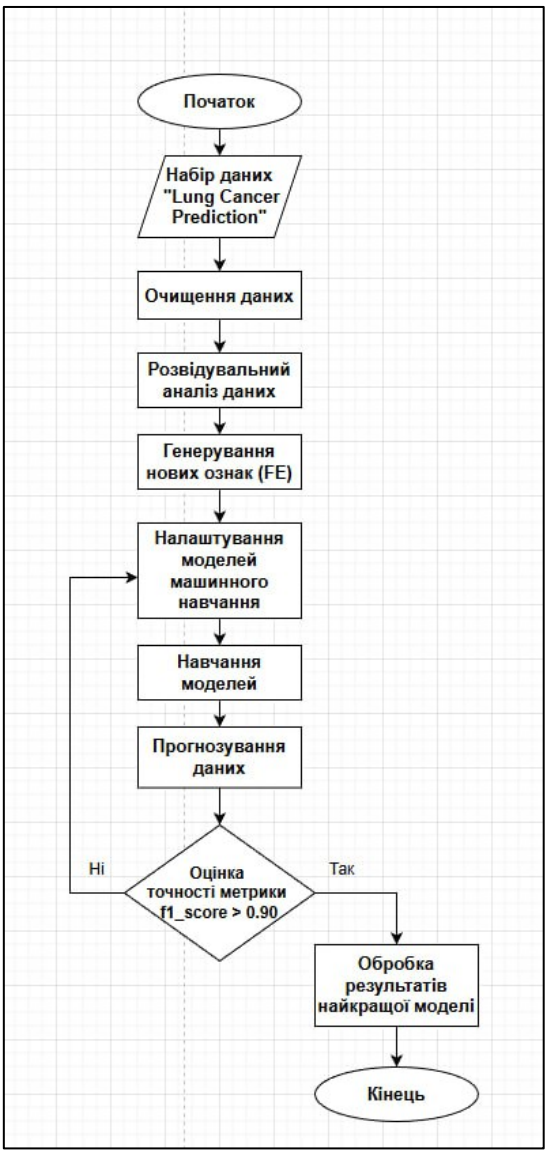

Рисунок 3.1 – Блок-схема алгоритму роботи інформаційної технології передбачення ризиків захворіти на рак легень

Зображена на рисунку 3.1 блок-схема показує як працює алгоритм, який показує процес роботи інформаційної технології. Вона описує процес оброблення даних, а саме очищення даних, розвідувальний аналіз та генерування нових ознак, які на наступному кроці віддаються для того щоб модель навчилась. Після успішного навчання моделі її результати застосовуються для прогнозування тестових даних де визначається точність кожної моделі. Після отримання оцінки точності моделі, аналізується чи менша вона за 90%, якщо так то модель повертається на доопрацювання та налаштування параметрів, тренується знову і так до моменту отримання задовільного результату. Далі обробляємо отримані

результати по усіх моделях, шляхом порівняння оцінки точності обираємо найкращу та аналізуємо які ознаки найбільше впливають на передбачення.

Для подальшого дослідження та моделювання розглянемо уніфіковану мову моделювання (UML). Мова UML є сучасним і зручним засобом для опису різних аспектів програмних систем. Вона допомагає візуалізувати структуру, логіку роботи та взаємодію елементів у майбутньому додатку або вже існуючому рішенні. UML активно використовують під час розробки програмного забезпечення, оскільки вона дає можливість побачити проєкт з різних сторін і зрозуміти, як саме повинні працювати окремі частини системи. Мова UML містить набір стандартних діаграм, які легко зчитуються навіть тими, хто не має значного досвіду у програмуванні. Завдяки цьому авторам проєктів простіше передавати свої ідеї іншим членам команди і уникати непорозумінь. UML допомагає структурувати інформацію таким чином, щоб кожен етап розробки був логічним та зрозумілим.

Переваги використання UML є досить очевидними. Основною є здатність швидко та наочно показувати складні процеси у простому графічному вигляді, що значно полегшує спільну роботу розробників, аналітиків та замовників. UML сприяє кращому плануванню, адже дозволяє заздалегідь передбачити можливі проблеми у проєкті і знайти шляхи їх вирішення ще до того, як почнеться написання коду. Крім того, UML допомагає економити час, оскільки чіткі діаграми зменшують кількість помилок та непорозумінь при спілкуванні між учасниками команди. Система UML є гнучкою та підходить як для маленьких проєктів, так і для великих комплексних систем. Вона дає можливість зосередитися на логічних аспектах і змісті майбутнього програмного продукту, а не на другорядних деталях, що робить процес проєктування більш продуктивним та зрозумілим [23].

Під час проектування інформаційної технології була розроблена діаграма Use-Case (рис. 3.2).

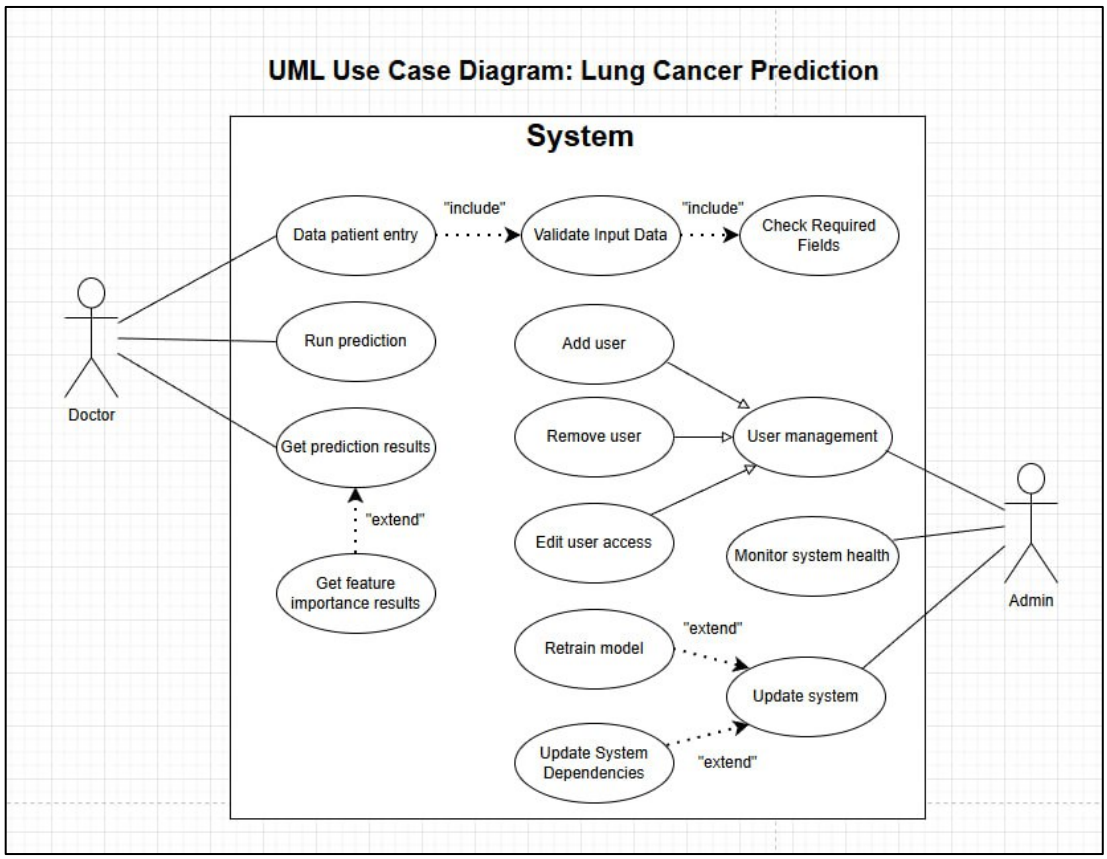

Рисунок 3.2 – UML Use Case діаграма

Рисунок 3.2 демонструє можливості використання системи передбачення раку легень. На діаграмі можемо бачити двох можливих акторів, це лікар та адміністратор системи. Основні можливості взаємодії лікаря із системою це введення даних пацієнтів, запуск передбачення та отримання результатів. Під час введення даних пацієнта обов’язково має відбуватись перевірка введених даних в тому числі перевірка заповнення усіх необхідних полів. В свою чергу основні функції взаємодії адміністратора із системою є управління користувачами, моніторинг працездатності системи та оновлення системи. Під час управління користувачами адміністратор має можливості додавання нових користувачів до системи, видалення існуючих користувачів та редагування прав доступу користувача до системи. Процес оновлення системи може супроводжуватись оновленням системних залежностей, таких як бібліотеки, які використовуються, чи безпосередньо версія компілятора, також як варіант можна перетренувати моделі, оновити їх параметри чи дані для збільшення точності прогнозування.

Далі побудуємо діаграму компонентів для відображення структури програмної системи (рис. 3.3).

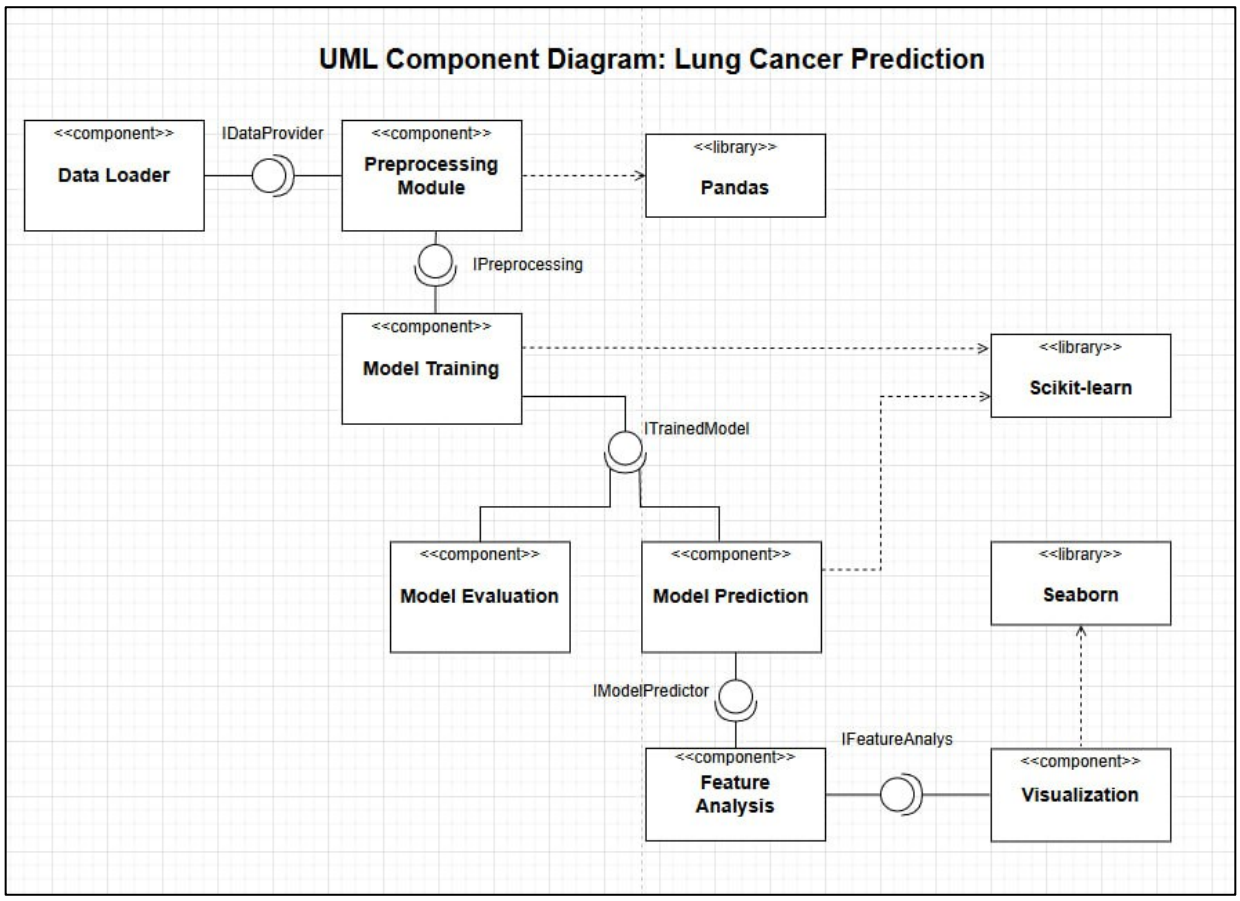

Рисунок 3.3 – UML діаграма компонентів

Діаграма компонентів зображена на рисунку 3.3 показує внутрішню будову системи. Основними компонентами системи виступають модулі завантаження даних, попередньої обробки, тренування моделі, оцінка моделі, передбачення моделі, аналіз впливовості ознак та компонент візуалізації. Компонент завантаження даних працює безпосередньо із даними, їх типом файлу і дає змогу розпочати роботу із даними за допомогою інтерфейсу «IDataProvider». Компонент попередньої обробки даних для функціонування використовує бібліотеку «Pandas» та відповідає безпосередньо за обробку даних, а саме їх очищення та генерування ознак. Компонент тренування моделі виконує функції із налаштуванням та тренуванням моделі, для цього він застосовує бібліотеку «Scikit-learn». Далі компоненти оцінки моделі та передбачення моделі, які відповідають відповідно за

оцінювання результатів тренування моделі та передбачення використовуючи результати тренування. Далі використовується компонент оцінки впливовості ознак, за результатами цих оцінок застосовується компонент візуалізації, який будує графіки для візуальної оцінки результатів, для цього компонент використовує бібліотеку «Seaborn».

Далі розробимо архітектуру технології (рис. 3.4).

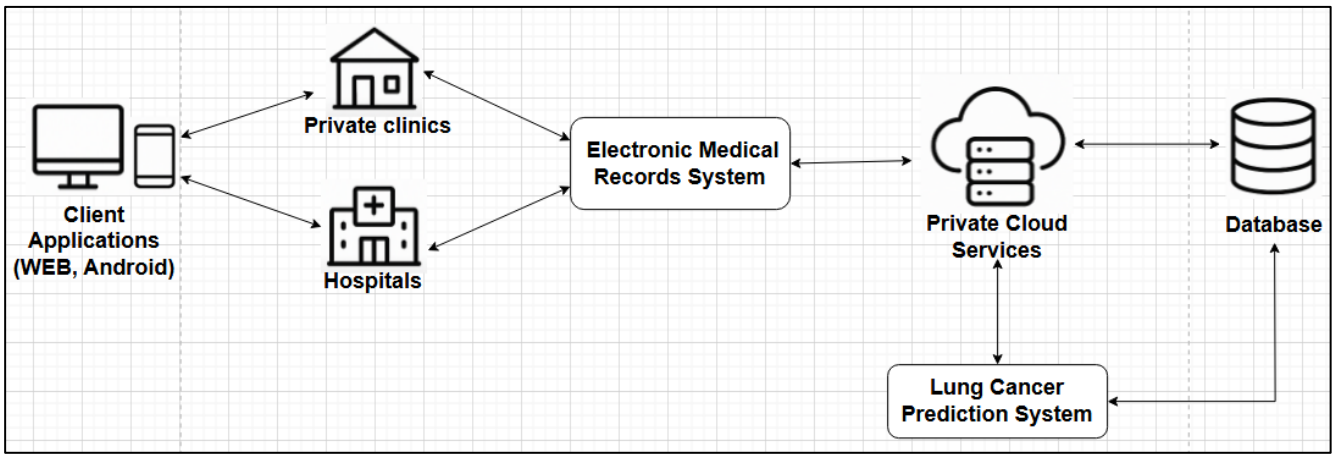

Рисунок 3.4 – Архітектура побудови інформаційної технології

Архітектура зображена на рисунку 3.4 описує топологію технології та її компонування. Розроблена технологія буде використовуватись у першу чергу в стаціонарних та приватних лікарнях, виконуючи функції додаткового інструмента діагностики при дослідженні наявності раку легень у пацієнтів. Для безпечного та зручного використання технологія буде розташовуватись у приватному хмарному сервері, доступ до якого будуть мати лише адміністратори та розробники. Для функціонування системи передбачення використовується база даних, в якій знаходяться дані про пацієнтів лікарень, на їх основі також буде вдосконалюватись система в майбутньому. База даних також знаходиться на хмарному сервісі. Для взаємодії із внутрішньою системою записів у лікарнях необхідно під’єднати її до нашого хмарного сервісу, через який і буде отримуватись доступ лікарів до функціоналу системи передбачення, а саме запис нових даних, запуск передбачення та отримання результатів. Для взаємодії із технологією за допомогою

комп’ютера передбачено реалізація web-застосунка, також можлива взаємодія за допомогою телефона використовуючи застосунок на операційній системі Android.

3.2 Налаштування моделей на основі дерев рішень

Під час роботи та побудови моделей машинного навчання було використано напрацювання у ноутбуці «Used Cars : Analysis and Prediction», який був опублікований користувачем «Vitalii Mokin» [24].

Для початку роботи та побудові моделей потрібно поділити попередньо підготовлені дані на тренувальну вибірку та тестову, із цим допоможе вбудована в бібліотеку Sklearn функція train_test_split (рис. 3.5).

```
# Data splitting
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

Рисунок 3.5 – Розділення даних на тренувальну та тестову вибірки

Рисунок 3.5 демонструє що розділення підготовлених даних відбувається у співвідношенні 20% на 80%. Це нам допомагає визначити параметр test_size=0,2, враховуючи те що кількість даних у наборі дорівнює 900, поділ відбувається у кількості 720 записів для тренувальної вибірки та 180 записів для тестової вибірки.

Наступним кроком потрібно визначитись метрикою, якою буде вимірюватись оцінка точності кожної моделі. Найбільш популярними метриками серед задач передбачення є R2_Score та F1_Score, але саме для задач класифікації найкраще підходить F1_Score, на неї і будемо орієнтуватися під час аналізу результатів, для цього створимо таблицю куди будемо передавати результати моделей та їх метрики (рис. 3.6).

```
# Results of prediction
results = pd.DataFrame(columns = ['model', 'f1_train', 'f1_test', 'r2_train', 'r2_test'])
```

Рисунок 3.6 – Таблиця для запису метрик

Для якісної побудови моделі необхідно налаштувати тюнінг, тобто для кожної моделі буде зроблений список із різними параметрами та різними значеннями кожного із цих параметрів та за допомогою методів тюнінгу будуть обрані найкращі комбінації із значень цих параметрі, що дозволить отримати найкращі можливі результати моделі з визначеного попередньо списку параметрів. Для цього будуть використовуватися методи GridSearchCV та RandomizedSearchCV.

Отже першим кроком визначимо список параметрів для тюнінгу моделі Random Forest (рис. 3.7).

```
param_RF = {
    'n_estimators': [100, 150],
    'max_depth': [3],
    'criterion':['gini', 'entropy'],
    'min_samples_split': [2, 3],
    'min_samples_leaf': [2, 3],
    'random_state' : [42],
    'max_samples': [0.2, 0.3]

}
```

```
%%time
model_tuning = GridSearchCV(estimator=RandomForestClassifier(), param_grid=param_RF
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_rf = RandomForestClassifier(**best_params)
model_rf.fit(x_train, y_train)
```

Рисунок 3.7 – Тюнінг моделі Random Forest

Після завершення тюнінгу відбувається процес навчання моделі, для якого в модель передаються найкращі обрані параметри (рис. 3.8).

```
# Train
train_predictions = model_rf.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_rf.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.8 – Навчання моделі Random Forest

За допомогою попередньо створеної таблиці для запису результатів виводимо на екран оцінки метрик натренованої моделі (рис. 3.9).

```
Train Data Metrics:
Precision :  0.9928774928774928
Recall :  0.9928774928774928
Accuracy :  0.9928774928774928
F1 Score :  0.9928774928774928
R2 Score :  0.9893813983881508


Test Data Metrics:
Precision :  0.9747474747474747
Recall :  0.9747474747474747
Accuracy :  0.9747474747474747
F1 Score :  0.9747474747474747
R2 Score :  0.9643846458250891
```

Рисунок 3.9 – Оцінки точності моделі Random Forest

Аналізуючи результати за метриками можна зробити висновок, що модель показала чудові результати, побудуємо для них матрицю плутанини (рис. 3.10).

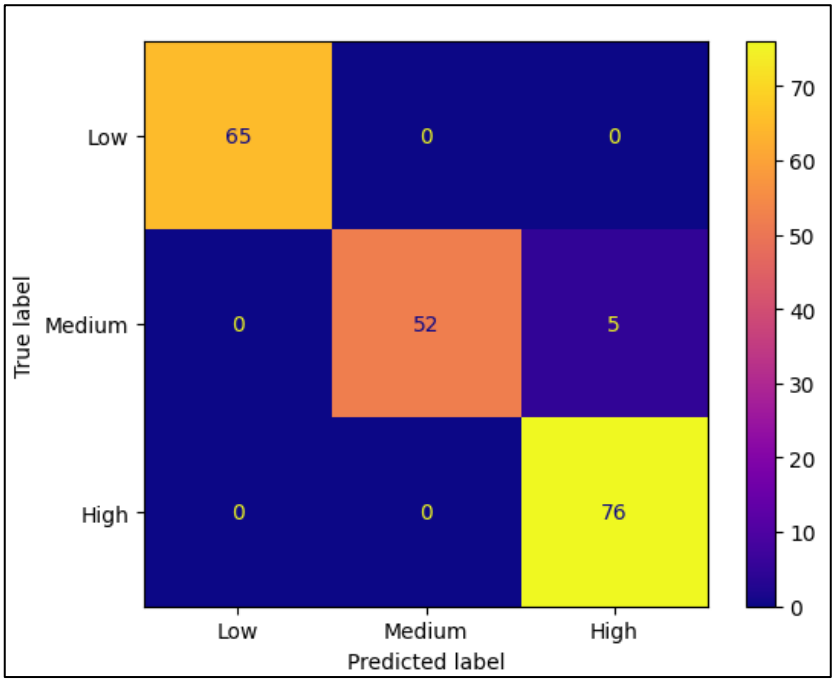

Рисунок 3.10 – Матриця плутанини моделі Random Forest

Далі проведемо налаштування моделі Extra Trees, принцип за яким відбувається тюнінг моделі ідентичний попередній моделі (рис. 3.11).

```
params_etc ={'n_estimators':[200, 300, 500],
             'max_depth': [3, 4, 5],
             'min_samples_split':  [2, 3],
             'min_samples_leaf':  [2, 3],
             'max_features': ['sqrt'],
             'ccp_alpha': [0.1],
             'random_state' : [42]}
```

```
%%time
model_tuning = GridSearchCV(estimator=ExtraTreesClassifier(), param_grid=params_etc,
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_etc = ExtraTreesClassifier(**best_params)
model_etc.fit(x_train, y_train)
```

Рисунок 3.11 – Тюнінг моделі Extra Trees

Наступним кроком після успішного тюнінгу моделі проводимо навчання на тренувальних даних, які передаються у модель без цільової ознаки (рис. 3.12).

```
# Train
train_predictions = model_etc.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_etc.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.12 – Навчання моделі Extra Trees

Далі щоб оцінити та проаналізувати якість моделі необхідно продемонструвати оцінки метрик (рис. 3.13).

```
Train Data Metrics:
Precision :  0.9245014245014245
Recall :  0.9245014245014245
Accuracy :  0.9245014245014245
F1 Score :  0.9245014245014245
R2 Score :  0.8874428229143978


Test Data Metrics:
Precision :  0.9141414141414141
Recall :  0.9141414141414141
Accuracy :  0.9141414141414141
F1 Score :  0.9141414141414141
R2 Score :  0.8789077958053028
```

Рисунок 3.13 – Оцінки точності моделі Extra Trees

Як бачимо із результатів оцінки навчання моделі, вона навчилась дуже добре, різниця між тренувальними та валідаційними даними мінімальна, що свідчить про якісне навчання.

Виведемо візуалізацію матриці плутанини для моделі Extra Trees, щоб візуально проаналізувати можливі помилки у класифікації (рис. 3.14).

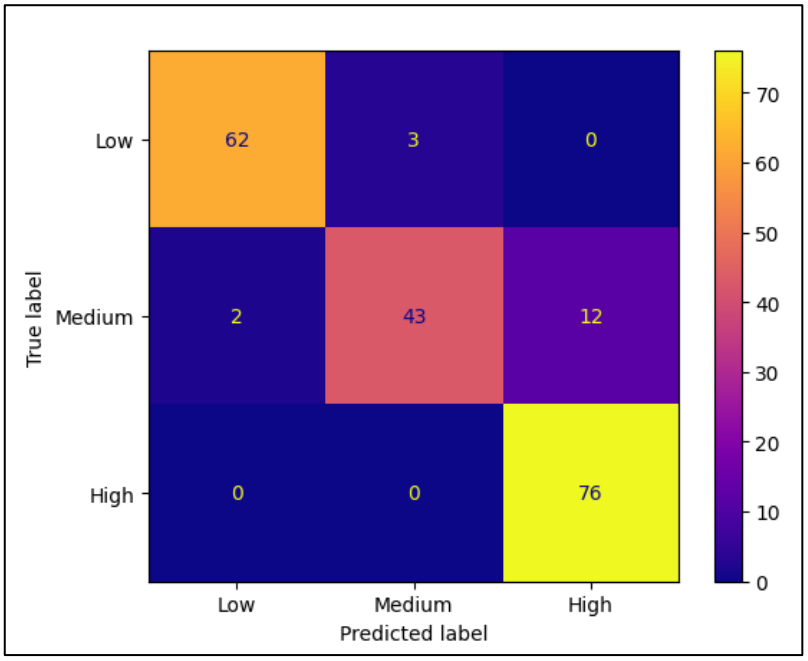

Рисунок 3.14 – Матриця плутанини моделі Extra Trees

Як бачимо із результатів матриці плутанини модель частіше помиляється на даних середнього рівня ризику.

Наступним кроком побудуємо модель Decision Tree (рис. 3.15).

```
params_DT ={'max_depth': [3,4],
            'min_samples_split': [2, 3],
            'min_samples_leaf': [1, 2],
            'max_features': ['sqrt'],
            'max_leaf_nodes': [ 5, 10, 15, 17]}
```

```
%%time
model_tuning = GridSearchCV(estimator=DecisionTreeClassifier(), param_grid=params_DT, cv=5)
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_dt = DecisionTreeClassifier(**best_params)
model_dt.fit(x_train, y_train)
```

Рисунок 3.15 – Тюнінг моделі Decision Trees

Після успішного тюнінгу навчаємо модель на найркащих параметрах, які визначили методом GridSearchCV (рис. 3.16).

```
# Train
train_predictions = model_dt.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_dt.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.16 – Навчання моделі Decision Trees

За допомогою попередньо створеної таблиці для запису результатів виводимо на екран оцінки метрик натренованої моделі (рис. 3.17).

```
Train Data Metrics:
Precision :  0.9301994301994302
Recall :  0.9301994301994302
Accuracy :  0.9301994301994302
F1 Score :  0.9301994301994302
R2 Score :  0.8959377042038772


Test Data Metrics:
Precision :  0.8888888888888888
Recall :  0.8888888888888888
Accuracy :  0.8888888888888888
F1 Score :  0.8888888888888888
R2 Score :  0.8432924416303917
```

Рисунок 3.17 – Оцінки точності моделі Decision Trees

Як бачимо із результатів оцінки навчання моделі, вона навчилась непогано, різниця між тренувальними та валідаційними даними присутня, але не критична, що свідчить про нормальне навчання.

Розглянемо результати передбачення моделі за допомогою матриці плутанини (рис. 3.18).

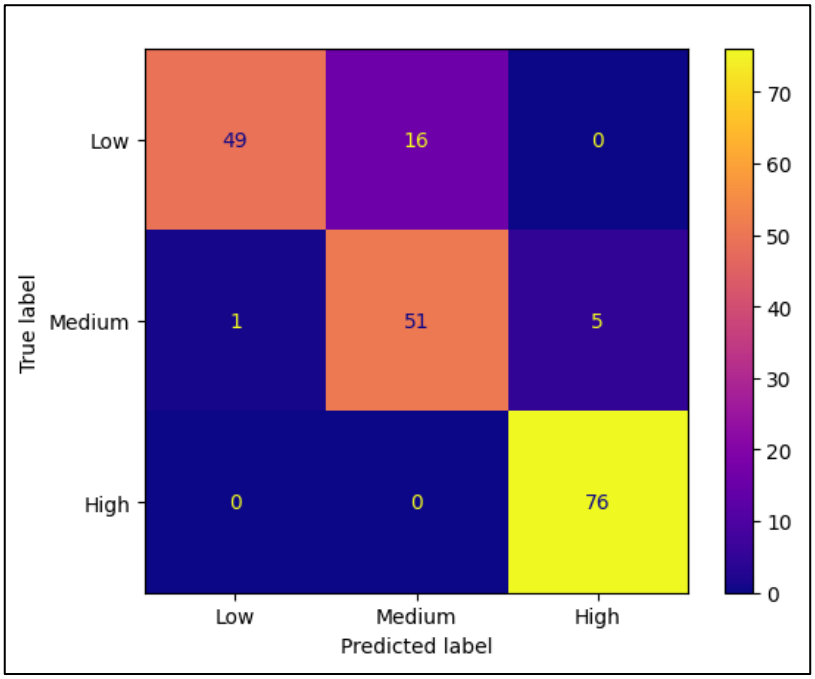

Рисунок 3.18 – Матриця плутанини моделі Decision Trees

Як бачимо із результатів матриці плутанини модель частіше помиляється на даних низького рівня ризику.

Алгоритм дерев рішень дозволяє наочно подати отримані результати, тому використаєм доступні засоби для формування графічної частини. (рис. 3.19).

Ці засоби часто застосовують для наочного подання результатів роботи моделі, адже вони легко сприймаються та не створюють труднощів у трактуванні. Крім того, графічна структура дерева дає можливість ретельно дослідити кожен його вузол і кінцеву гілку, що забезпечує повне бачення процесу переходу від початкової точки до кінцевого рішення.

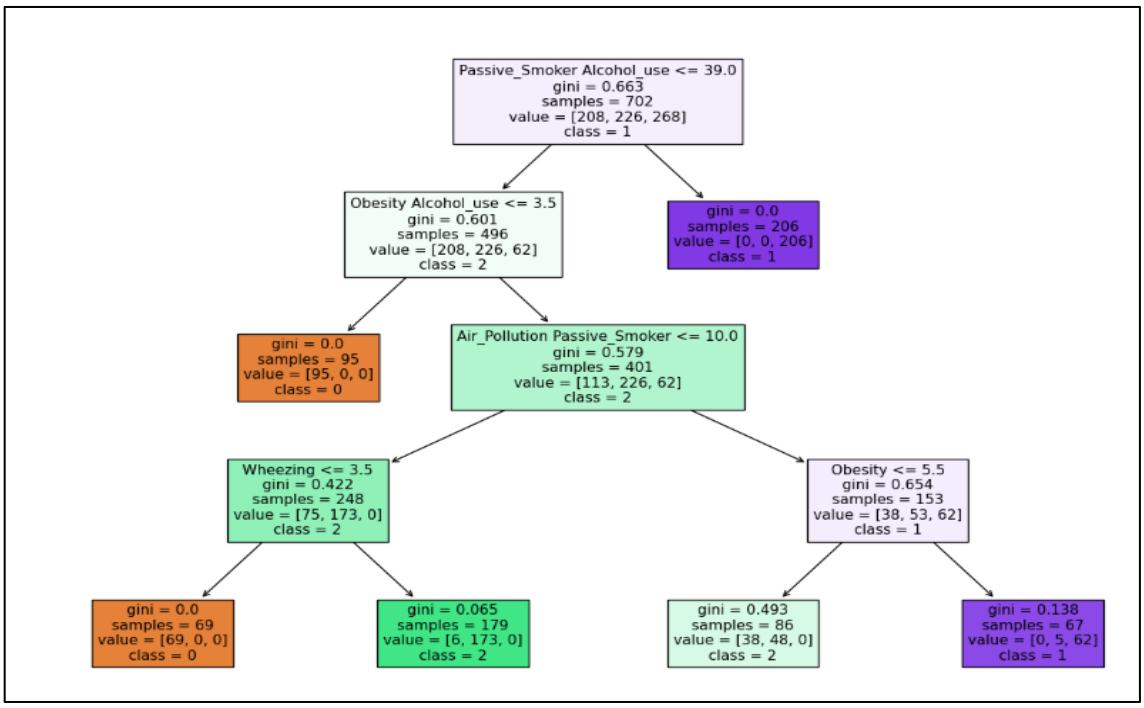

Рисунок 3.19 – Візуалізація дерева рішень

Після попереднього завантаження підключаємо бібліотеку dtreeviz, яка забезпечує розширені можливості візуалізації та дозволяє детальніше відобразити вузли і листя дерева рішень. На рисунку 3.20 подано налаштування, необхідні для створення візуального графіка за підсумками навчання моделі та подальшого збереження отриманого результату. (рис. 3.21).

```
import dtreeviz

viz_model = dtreeviz.model(model_dt,
                           X_train=x_train, y_train=y_train,
                           feature_names=feature_names,
                           target_name='Lung Cancer',
                           class_names=['Low', 'Medium', 'High'])

v = viz_model.view()     # render as SVG into internal object
v.save("Lung Cancer.svg")  # save as svg
```

Рисунок 3.20 – Параметри для побудови графіка дерева рішень dtreeviz

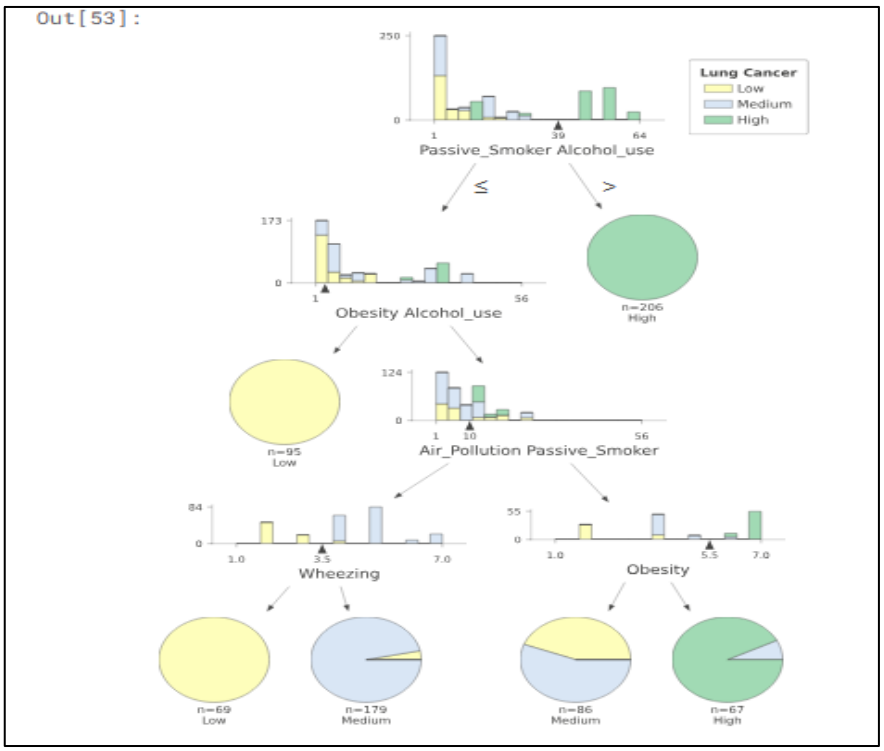

Рисунок 3.21 – Графік дерева рішень dtreeviz

3.3 Налаштування ансамблевих методів

Одним із представників ансамблевих методів є модель Ada Boost, принцип за яким відбувається тюнінг ідентичний попереднім моделям, та полягає у використанні методу GridSearchCV для списку вказаних параметрів, з яких метод обирає найкращу комбінацію, тобто таку яка дає найвищу точність (рис. 3.22).

```
param_ADA = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1, 1],
    'random_state' : [42],
    'algorithm': ['SAMME.R'],
}
```

```
%%time
model_tuning = GridSearchCV(estimator=AdaBoostClassifier(), param_grid=param_ADA, cv=5)
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_ada = AdaBoostClassifier(**best_params)
model_ada.fit(x_train, y_train)
```

Рисунок 3.22 – Тюнінг моделі Ada Boost

Далі використовуючи отриману найкращу комбінацію параметрів проводимо навчання моделі на їх основі (рис. 3.23).

```
# Train
train_predictions = model_ada.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_ada.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.23 – Навчання моделі Ada Boost

Далі щоб оцінити та проаналізувати якість моделі необхідно продемонструвати оцінки метрик (рис. 3.24).

```
Train Data Metrics:
Precision :  0.99002849002849
Recall :  0.99002849002849
Accuracy :  0.99002849002849
F1 Score :  0.99002849002849
R2 Score :  0.9851339577434111


Test Data Metrics:
Precision :  0.9848484848484849
Recall :  0.9848484848484849
Accuracy :  0.9848484848484849
F1 Score :  0.9848484848484849
R2 Score :  0.9786307874950534
```

Рисунок 3.24 – Оцінки точності моделі Ada Boost

За результатами оцінки метрик можна зробити висновок про чудову якість моделі із високими показниками точності.

Розглянемо дані результати за допомогою матриці плутанини, та проаналізуємо кількість помилок та їх характер (рис. 3.25).

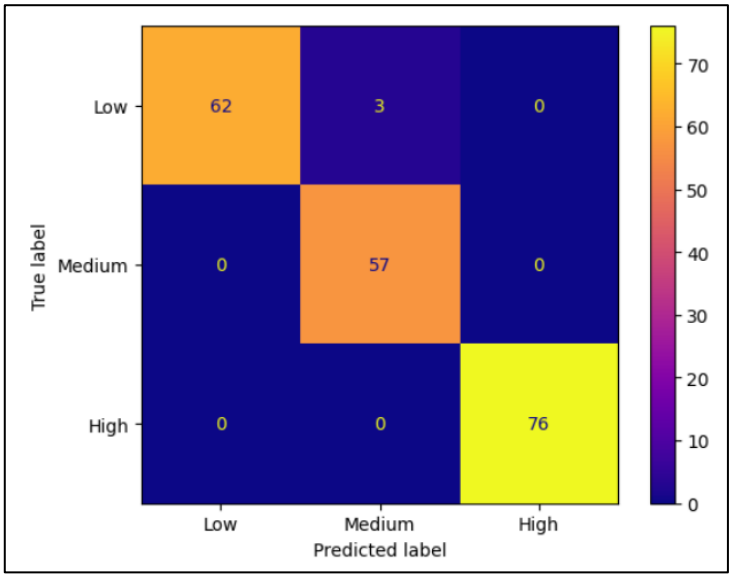

Рисунок 3.25 – Матриця плутанини моделі Ada Boost

Візуалізація матриці плутанини продемонструвала що помилки модель робить при спробі класифікувати низький рівень ризику, але кількість помилок є мінімальною тому можна вважати що модель якісна.

Наступним кроком, зображеного на рисунку 3.26, проведемо тюнінг моделі XGBoost.

```
params_XGB ={'n_estimators': [100, 200, 300],
             'num_leaves': [3, 5, 7, 10],
             'max_depth': [2, 3, 5],
             'subsample': [0.01],
             'learning_rate': [0.01, 0.05, 0.1],
             'objective': ['multi:softmax'],
             'random_state' : [42],
             'num_class': [3]}
```

```
%%time
model_tuning = GridSearchCV(estimator=XGBClassifier(), param_grid=params_XGB, cv=5)
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_xgb = XGBClassifier(**best_params)
model_xgb.fit(x_train, y_train)
```

Рисунок 3.26 – Тюнінг моделі XGBoost

На рисунку 3.26 зображено процес тюнінгу моделі, результати якого будуть використані під час тренування моделі (рис. 3.27).

```
# Train
train_predictions = model_xgb.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_xgb.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.27 – Навчання моделі XGBoost

За аналогією із попередніми моделями результати у вигляді оцінки точності за метриками такими як R2_Score та F1_Score виведемо на рисунку 3.28, а матрицю плутанини, яка демонструє кількість вірних і невірних передбачень класифікації виведем на рисунку 3.29.

```
Train Data Metrics:
Precision :  0.9643874643874644
Recall :  0.9643874643874644
Accuracy :  0.9643874643874644
F1 Score :  0.9643874643874644
R2 Score :  0.9023088651709867


Test Data Metrics:
Precision :  0.9191919191919192
Recall :  0.9191919191919192
Accuracy :  0.9191919191919192
F1 Score :  0.9191919191919192
R2 Score :  0.8219232291254452
```

Рисунок 3.28 – Оцінки точності моделі XGBoost

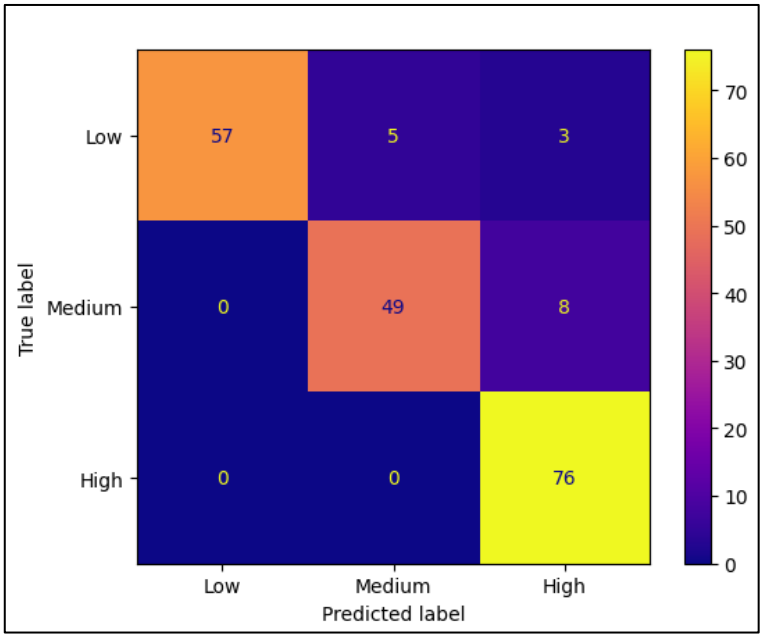

Рисунок 3.29 – Матриця плутанини моделі XGBoost

Модель показала чудові результати, мала кількість помилок, головні промахи класифікації зосереджені під час передбачення низького та середнього рівнів ризику, у той самий час як для високого рівня усі значення були класифіковані правильно.

Наступним кроком виконаємо побудову та налаштування моделі LGBM та проведемо аналіз результатів (рис. 3.30).

```
model_tuning = RandomizedSearchCV(
    estimator=LGBMClassifier(objective='multiclass',
        num_class=3,
        random_state=42,
        verbosity=-1
    ),
    param_distributions=param_lgbm,
    cv=3,
    n_iter=30,  # кількість випадкових комбінацій
    scoring='accuracy',
    random_state=42,
    verbose=0,
    n_jobs=-1
)

model_tuning.fit(x_train, y_train)


best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_lgbm = LGBMClassifier(**best_params,objective='multiclass',
        num_class=3,
        random_state=42,
        verbosity=-1 )
model_lgbm.fit(x_train, y_train)
```

Рисунок 3.30 – Тюнінг моделі LGBM

На рисунку 3.30 зображено використання методу RandomizedSearchCV, для моделі LGBM, який випадковим чином обирає комбінацію із представлених параметрів для економії часу на обчислення. Потім ці параметри використаємо під час тренування моделі (рис. 3.31).

```
# Train
train_predictions = model_lgbm.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_lgbm.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.31 – Навчання моделі LGBM

Отримані результати після навчання моделі продемонстровано рисунку 3.32, проаналізувавши які можна сказати що модель навчилась гарно, адже має високі показники за обома метриками та мінімальну різницю між тестовими та тренувальними даними.

```
Train Data Metrics:
Precision :  0.9886039886039886
Recall :  0.9886039886039886
Accuracy :  0.9886039886039886
F1 Score :  0.9886039886039886
R2 Score :  0.9830102374210412


Test Data Metrics:
Precision :  0.9848484848484849
Recall :  0.9848484848484849
Accuracy :  0.9848484848484849
F1 Score :  0.9848484848484849
R2 Score :  0.9786307874950534
```

Рисунок 3.32 – Оцінки точності моделі LGBM

Далі продемонструємо результати навчання моделі LGBM використовуючи візуалізацію матриці плутанини (рис. 3.33).

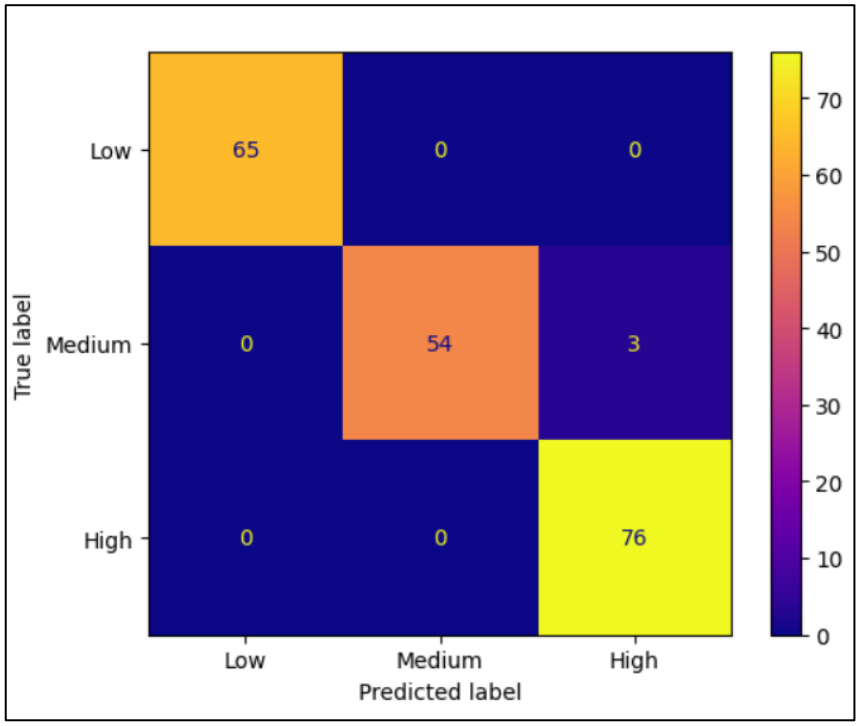

Рисунок 3.33 – Матриця плутанини моделі LGBM

Матриця плутанини для моделі LGBM показує що модель має мінімальну кількість помилок саме при класифікації середнього рівня ризику, аналізуючи результати точності можна з впевненістю сказати що модель навчилась чудово.

3.4 Налаштування класичних методів

Проведемо налаштування параметрів та проведемо тренування для моделі k-Nearest Neighbors, яка працює на основі алгоритму пошуку найближчих сусідів у багатовимірному просторі ознак. Для нового об’єкта вона знаходить кілька найбільш подібних точок із тренувального набору, так званих сусідів і на основі їхніх класів або значень приймає рішення щодо класифікації (рис. 3.34).

```
param_knn = {
    'n_neighbors': [6, 9, 12],
    'weights': ['uniform'],
    'metric': ['minkowski']
}
```

```
%%time
model_tuning = GridSearchCV(estimator=KNeighborsClassifier(), param_grid=param_knn, cv=5)
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_knn = KNeighborsClassifier(**best_params)
model_knn.fit(x_train, y_train)
```

Рисунок 3.34 – Тюнінг моделі K-Neighbors

Після успішного тюнінгу моделі, найкращі гіперпараметри використовуємо для наступного кроку, а саме навчання моделі (рис. 3.35), після чого демонструємо результати метрик після навчання (рис. 3.36).

```
# Train
train_predictions = model_knn.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_knn.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.35 – Навчання моделі K-Neighbors

```
Train Data Metrics:
Precision :  0.9957264957264957
Recall :  0.9957264957264957
Accuracy :  0.9957264957264957
F1 Score :  0.9957264957264957
R2 Score :  0.9936288390328905


Test Data Metrics:
Precision :  0.9949494949494949
Recall :  0.9949494949494949
Accuracy :  0.9949494949494949
F1 Score :  0.9949494949494949
R2 Score :  0.9928769291650178
```

Рисунок 3.36 – Оцінки точності моделі K-Neighbors

Результати навчання показують майже ідеальний результат, це можна пояснити тим, як саме працює алгоритм методу, оскільки набір даних у нас описує ознаки, які впливають на виникнення раку це означає що схожі між собою ознаки мають в кінцевому результаті подібні ризики захворюваності на рак.

Оглянемо матрицю плутанини для моделі зображену на рисунку 3.37.

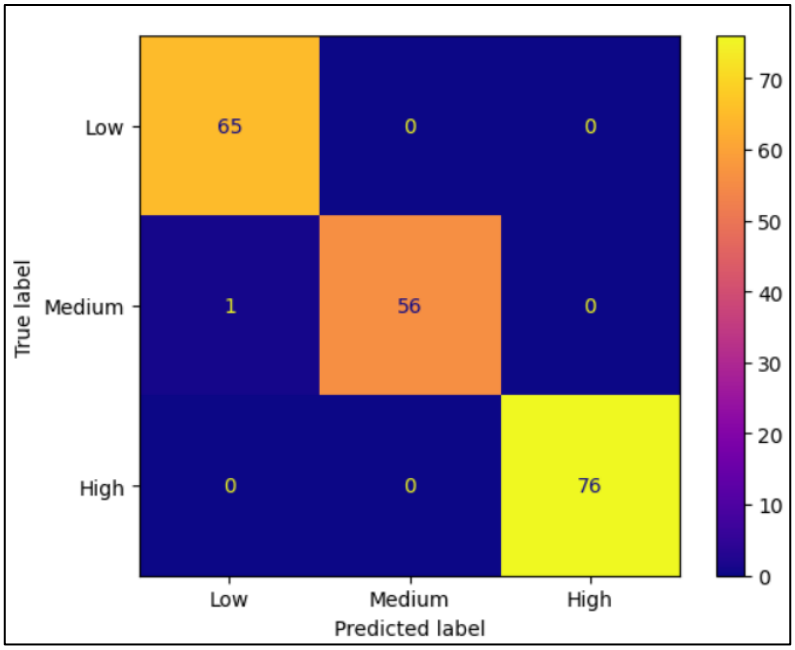

Рисунок 3.37 – Матриця плутанини моделі K-Neighbors

Аналізуючи матрицю плутанини можемо зробити висновок що модель помилилась лише при класифікації одного значення для середнього рівня ризику.

Отже далі спробуємо налаштувати модель Support Vector Machines, першим кроком виберемо найкращі гіперпараметри (рис. 3.38).

```
param_svm = {
    'C': [0.1],              # маленькі C → сильніша регуляризація
    'kernel': ['rbf'],
    'gamma': ['scale', 'auto', 0.001, 0.01, 0.1, 1]
}
```

```
%%time
model_tuning = GridSearchCV(estimator=SVC(), param_grid=param_svm,
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_svm = SVC(**best_params)
model_svm.fit(x_train, y_train)
```

Рисунок 3.38 – Тюнінг моделі Support Vector Machines

Запустимо навчання моделі після успішного тюнінгу та продемонструємо результати на рисунку 3.39.

```
# Train
train_predictions = model_svm.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_svm.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.39 – Навчання моделі Support Vector Machines

Результати навчання виводимо для тренувальних та тестових даних для порівняння точності (рис. 3.40).

```
Train Data Metrics:
Precision :  0.9985754985754985
Recall :  0.9985754985754985
Accuracy :  0.9985754985754985
F1 Score :  0.9985754985754985
R2 Score :  0.9978762796776302


Test Data Metrics:
Precision :  0.9848484848484849
Recall :  0.9848484848484849
Accuracy :  0.9848484848484849
F1 Score :  0.9848484848484849
R2 Score :  0.9358923624851603
```

Рисунок 3.40 – Оцінки точності моделі Support Vector Machines

Метрики показують досить високу оцінку точності моделі, що означає гарно навчену модель та демонструє її якість. Для перевірки тих незначних помилок, які присутні під час передбачення моделі виведемо матрицю плутанини (рис. 3.41).

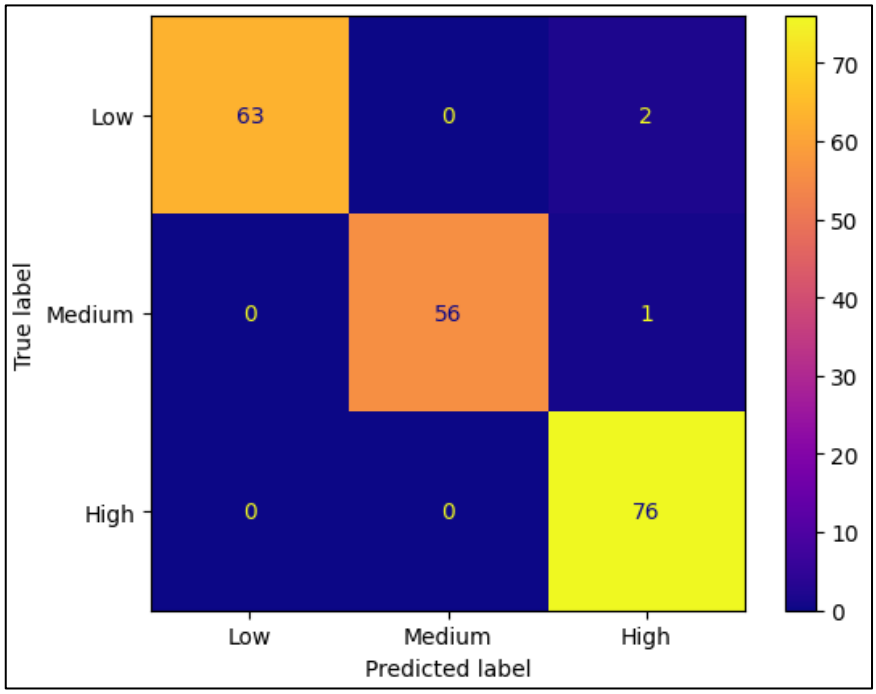

Рисунок 3.41 – Матриця плутанини моделі Support Vector Machines

Проаналізувавши рисунок 3.41, можна зробити висновок, що модель зробила малу кількість помилок при класифікації низького рівня ризику та середнього рівня ризику.

Далі проведемо налаштування моделі Logistic Regression, алгоритм налаштування ідентичний попередній моделі (рис. 3.42).

```
param_logreg = {
    'penalty': ['l1'],    # тип регуляризації
    'C': [0.01, 0.1],                    # сила регуляризації (обернена)
    'solver': ['saga', 'lbfgs'],                    # метод оптимізації
}
```

```
%%time
model_tuning = GridSearchCV(estimator=LogisticRegression(), param_grid=param_logreg,
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_lr = LogisticRegression(**best_params)
model_lr.fit(x_train, y_train)
```

Рисунок 3.42 – Тюнінг моделі Logistic Regression

Проводимо тренування моделі з використанням найліпших обраних гіперпараметрів (рис. 3.43).

```
# Train
train_predictions = model_lr.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_lr.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.43 – Навчання моделі Logistic Regression

Виводимо результати навчання та передбачення для порівняння метрик між тренувальними та тестовими даними (рис. 3.44).

```
Train Data Metrics:
Precision :  0.9715099715099715
Recall :  0.9715099715099715
Accuracy :  0.9715099715099715
F1 Score :  0.9715099715099715
R2 Score :  0.9575255935526029


Test Data Metrics:
Precision :  0.9494949494949495
Recall :  0.9494949494949495
Accuracy :  0.9494949494949495
F1 Score :  0.9494949494949495
R2 Score :  0.9287692916501781
```

Рисунок 3.44 – Оцінки точності моделі Logistic Regression

Результати демонструють добре навчену модель, різниця між тренувальними та валідаційними даними мінімальна, що свідчить про хороше та якісне навчання.

Проаналізуємо матрицю плутанини, яка демонструє результати помилок при класифікації (рис. 3.45).

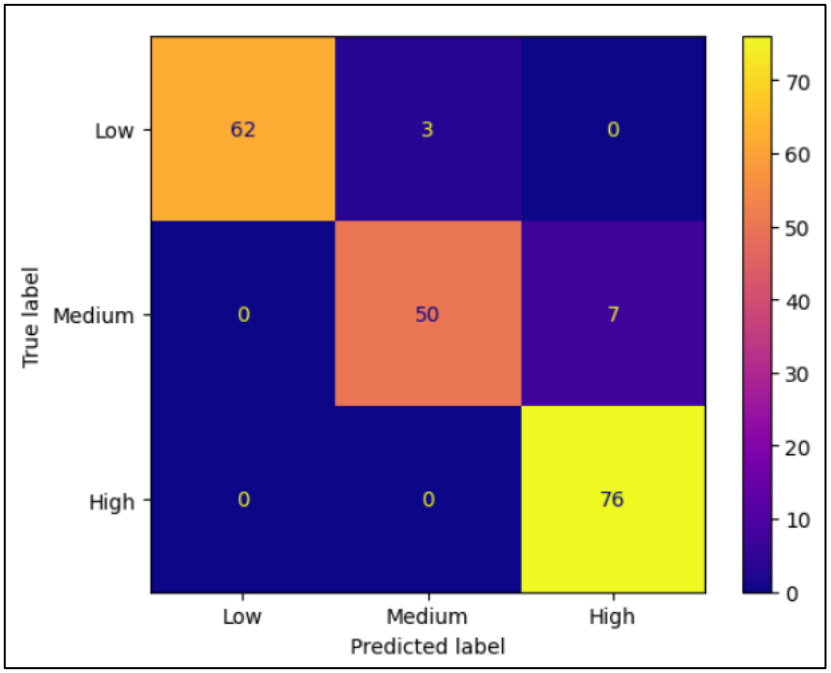

Рисунок 3.45 – Матриця плутанини моделі Logistic Regression

Аналізуючи рисунок 3.45 бачимо, що помилки при класифікації присутні, і найбільша їх кількість зосереджена при класифікації середнього рівня ризику захворіти на рак легень.

3.5 Налаштування моделей нейронних мереж

Наступним кроком проведемо моделювання за допомогою Multilayer Perceptron (рис. 3.46). Налаштування даної моделі відбувається за аналогією із попередніми, але при цьому алгоритм роботи методу відповідає нейронній мережі. MLP складається з кількох шарів штучних нейронів, вхідного, одного або кількох прихованих та вихідного шару. Завдяки такій структурі модель здатна моделювати нелінійні закономірності та показує високу гнучкість у задачах класифікації.

```
params_mlp ={'hidden_layer_sizes': [50, 75, 100],
             'learning_rate_init': [0.001],
             'max_iter': [100, 200, 300],
             'solver': ['adam'],
             'early_stopping': [True],
             'random_state' : [42]}
```

```
%%time
model_tuning = GridSearchCV(estimator=MLPClassifier(),
model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_mlp = MLPClassifier(**best_params)
model_mlp.fit(x_train, y_train)
```

Рисунок 3.46 – Тюнінг моделі MLP

Далі проводимо навчання моделі використовуючи обрані найкращі параметрами (рис. 3.47).

```
# Train
train_predictions = model_mlp.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test
test_predictions = model_mlp.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result
performTrain(train_predictions)
performTest(test_predictions)
```

Рисунок 3.47 – Навчання моделі MLP

Виводимо результати точності для порівняння між тренувальними та тестовими даними (рис. 3.48).

```
Train Data Metrics:
Precision :  0.99002849002849
Recall :  0.99002849002849
Accuracy :  0.99002849002849
F1 Score :  0.99002849002849
R2 Score :  0.9851339577434111


Test Data Metrics:
Precision :  0.9848484848484849
Recall :  0.9848484848484849
Accuracy :  0.9848484848484849
F1 Score :  0.9848484848484849
R2 Score :  0.9786307874950534
```

Рисунок 3.48 – Оцінки точності моделі MLP

Як бачимо оцінка результатів навчання моделі показує чудові результати, різниця між тренувальними та валідаційними даними мінімальна, що свідчить про якісне навчання.

Проаналізуємо матрицю плутанини, яка демонструє результати помилок при класифікації (рис. 3.49).

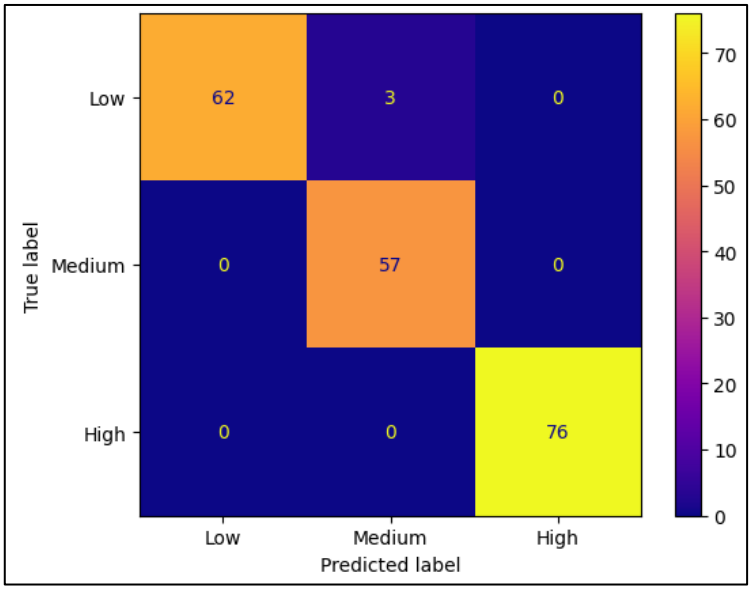

Рисунок 3.49 – Матриця плутанини моделі MLP

З рисунку 3.49 видно, що помилки при класифікації присутні, але їх мінімальна кількість та вони зосереджені при класифікації низьких рівнів ризику.

Далі побудуємо просту нейронну мережу за допомогою бібліотеки keras визначивши дня неї два основні шари Dense (рис. 3.50).

```
model_21 = tf.keras.Sequential([
    Input(shape=(x_train.shape[1],)),
    tf.keras.layers.Dense(9, activation='relu'),
    tf.keras.layers.Dense(3, activation='softmax')
])

model_21.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
model_21.fit(x_train, y_train, epochs=88, batch_size=16, validation_split=0.27, verbose = 0,
            callbacks=[])
```

Рисунок 3.50 – Побудова нейронної мережі

Як бачимо нейронна мережа має два шари Dense, перший шар має 9 нейронів та функцію активації «relu», вона допомагає знайти у даних нелінійні закономірності що допомагає мережі знайти складні залежності. Другий шар вихідний, має у собі 3 нейрони та функцію активації «softmax», це означає що наша модель виконує класифікацію на три класи, оскільки для кожного із 3 нейронів функція активації «softmax» повертає ймовірності і допомагає визначити

фінальний клас. Далі модель компілюється та навчається на вхідних даних, основні параметри мережі це «epochs», цей параметр відповідає за кількість епох, тобто скільки разів нейронна мережа буде опрацьовувати дані, «batch_size», який відповідає за оновлення ваг, у нашому випадку при проходженні кожних 16 рядків мережа буде змінювати та корегувати свої ваги, та «validation_split» цей параметр відповідає за розбиття даних на тренувальні та тестові, у нашому випадку значення 0.27 позначає що 27% даних будуть тестові а інші 73% тренувальні.

Далі виконаємо навчання нейронної мережі та відобразимо їх за допомогою графіку (рис. 3.51).

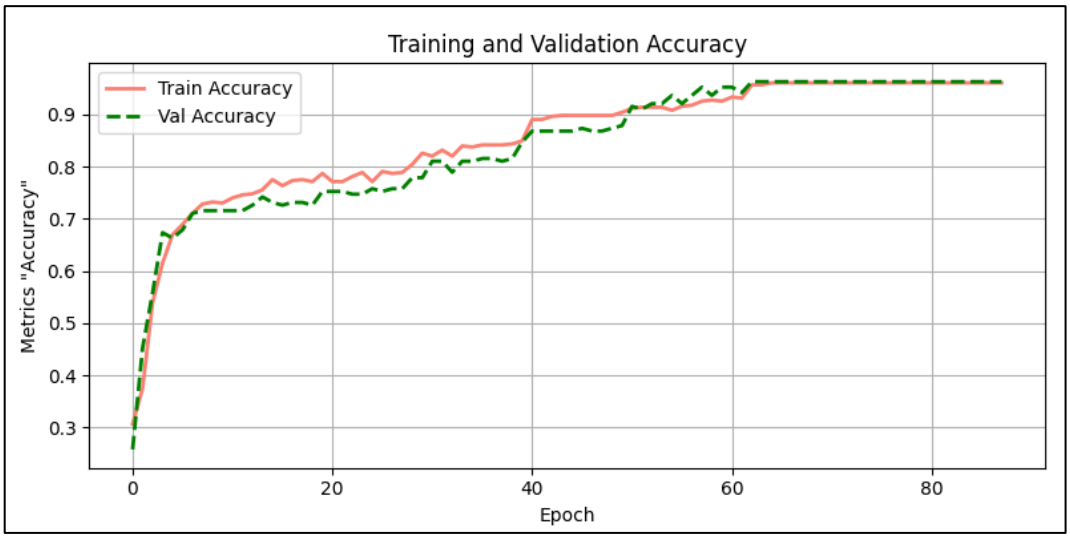

Рисунок 3.51 – Результат навчання нейронної мережі

Проаналізувавши графік, який показує точність навчання на кожній епосі, тобто демонструє прогрес навчання нейронної мережі, можна побачити, що на перших епохах точність сягає 0.7 і далі продовжує поступово зростати виходячи на своє плато після 60 епох. Значних падінь точності на графіку не спостерігається, що свідчить про те що модель не губиться у даних та чітко знаходить закономірності між ними. Не менш важливим є те що показники точності для тестових та тренувальних даних корелюються між собою та не мають суттєвих відхилень.

Далі виведемо графік показників міри помилок нейронної на тренувальних та тестових даних Train Loss та Val Loss (рис. 3.52).

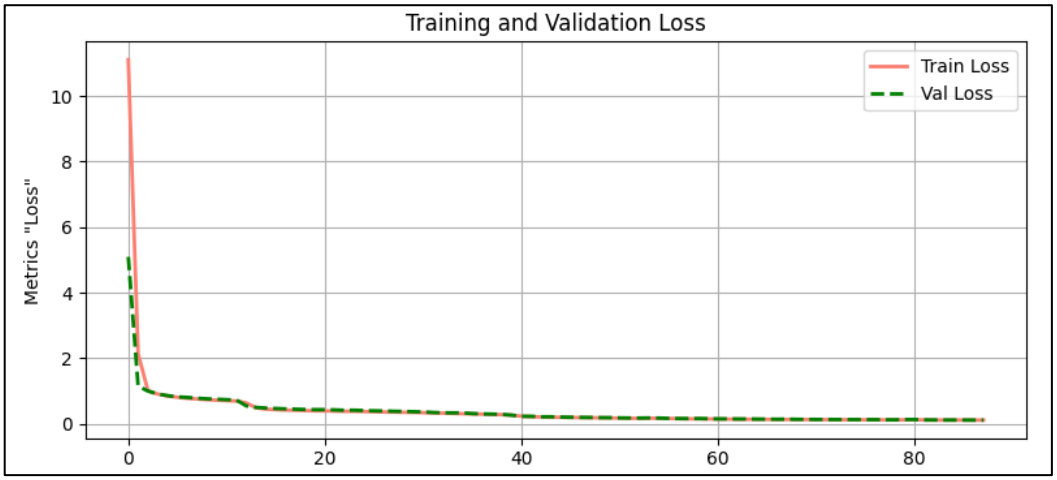

Рисунок 3.52 – Графік показника Train Loss та Val Loss нейронної мережі

Графік демонструє нам різке падіння оцінки помилок нейронної мережі, це означає що модель дуже швидко навчається базовим патернам у даних, початкові ваги були далекими від оптимальних, алгоритм працює коректно, швидко виправляючи грубі помилки. Майже ідеальна кореляція між графіками на тестових та тренувальних даних це дуже гарний показник того, що модель не перенавчається, узагальнення даних проходить стабільно. Після п’ятої епохи оцінки Loss для обох типів даних плавно та стабільно зменшуються, що свідчить про відсутність різких шумів у даних, які б могли викликати нестабільність, отже навчання проходить збалансовано.

	model	accuracy	val accuracy	loss	val loss
0	NN 2 layers	0.960938	0.963158	0.113192	0.112308

Рисунок 3.53 – Показники точності нейронної мережі на тестових та тренувальних даних

На рисунку 3.53 показники точності нейронної мережі для тренувальних та тестових даних знаходяться в межах 0.96, це хороший результат, який демонструє що проста нейронна мережа добре навчилась та чудово узагальнює наявні дані.

3.6 Дослідження впливу ознак на найкращу модель

Для оцінки змін порівняно із попереднім дослідженням виведемо його результати (рис. 3.54).

Отже були отримані результати навчання моделей, їх оцінки метрики F1_Score для тестових та тренувальних даних по усіх моделях (рис. 3.55).

	model	f1_train	f1_test	r2_train	r2_test	short
0	RandomForest Classifier	0.9675	0.98	0.905566	0.925012	RF
1	ADABoost Classifier	0.90125	0.895	0.850794	0.842526	ADA
2	ExtraTrees Classifier	0.8875	0.895	0.78469	0.797533	ExT
3	DecisionTree Classifier	0.9625	0.95	0.898011	0.880019	DT
4	XGB Classifier	0.89	0.935	0.788467	0.857523	XGB
5	MLP Classifier	0.9125	0.94	0.862126	0.910015	MLP

Рисунок 3.54 – Результати моделей при попередньому дослідженні

	model	f1_train	f1_test	r2_train	r2_test	short
0	RandomForest Classifier	0.992877	0.974747	0.989381	0.964385	RF
1	ADABoost Classifier	0.990028	0.984848	0.985134	0.978631	ADA
2	ExtraTrees Classifier	0.924501	0.914141	0.887443	0.878908	ExT
3	DecisionTree Classifier	0.930199	0.888889	0.895938	0.843292	DT
4	XGB Classifier	0.964387	0.919192	0.902309	0.821923	XGB
5	MLP Classifier	0.990028	0.984848	0.985134	0.978631	MLP
6	LR Classifier	0.97151	0.949495	0.957526	0.928769	LR
7	SVM Classifier	0.998575	0.984848	0.997876	0.935892	SVM
8	KNN Classifier	0.995726	0.994949	0.993629	0.992877	KNN
9	LGBM Classifier	0.988604	0.984848	0.98301	0.978631	LGBM

	model	accuracy	val accuracy	loss	val loss
0	NN 2 layers	0.960938	0.963158	0.113192	0.112308

Рисунок 3.55 – Результати поточного дослідження

Якщо порівняти результати раніше використаних моделей без генерування нових ознак (рис. 3.54) і ці ж самі моделі на очищених даних та з новими ознаками (рис. 3.55), то як результат можемо бачити що вдалося покращити майже усі моделі, особливо бустингові такі як AdaBoost та XGB.

Якщо ж виділяти найкращу модель з усіх представлених то це KNN, яка показує найкращі результати, оскільки має мінімальну різницю точності між тренувальними та валідаційними даними та показує точність у 0.994 за метрикою F1_score.

Наступним кроком проведемо аналіз найкращої моделі, а саме KNN, для цього скористаємось функцією permutation_importance, яка допомагає отримати оцінку впливовості ознак на нашу модель під час передбачення. Алгоритм цієї функціє вимірює наскільки погіршується якість моделі, тобто її точність передбачення, якщо зруйнувати зв’язок між однією ознакою та цільовою ознакою шляхом випадкового перемішування значень цієї ознаки. Іншими словами якщо під час виконання алгоритму функції на ознаці точність моделі суттєво погіршується, то це означає що ознака важлива, якщо ж точність моделі ніяк не змінюється,

значить ознака має мінімальний вплив на модель. Оцінки впливовості ознак за допомогою функції приведені на рисунку 3.56 та на рисунку 3.57.

	Feature	Importance
43	Alcohol_use_OccuPational_Hazards	0.066667
39	Genetic_Risk_Alcohol_use	0.056061
42	Obesity_OccuPational_Hazards	0.055051
41	Obesity_Alcohol_use	0.048485
30	Smoking_Passive_Smoker	0.042424
35	Passive_Smoker_Obesity	0.039899
0	Age	0.039394
25	Air_Pollution_Passive_Smoker	0.038384
36	Passive_Smoker_Alcohol_use	0.037374
37	Passive_Smoker_OccuPational_Hazards	0.037374
38	Genetic_Risk_Obesity	0.035354
31	Smoking_Genetic_Risk	0.032323
33	Smoking_OccuPational_Hazards	0.031313
32	Smoking_Alcohol_use	0.026768
34	Passive_Smoker_Genetic_Risk	0.025758
28	Air_Pollution_Alcohol_use	0.022222

Рисунок 3.56 – Результати оцінки впливовості ознак на модель k-NN за допомогою функції permutation_importance

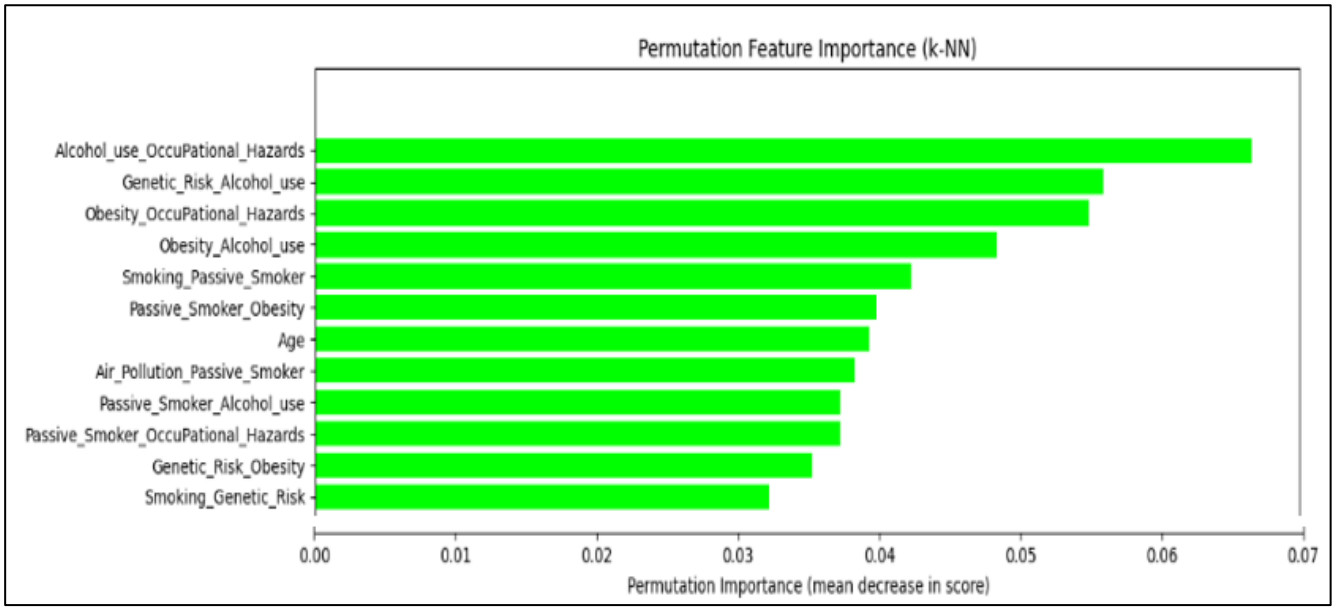

Рисунок 3.57 – Графік оцінки впливовості ознак на модель k-NN за допомогою функції permutation_importance

Як бачимо із рисунку 3.57 найбільш впливовими ознаками є нові згенеровані ознаки, які є поєднанням початкових ознак з найбільшою кореляцією між собою. Серед цих ознак найбільш впливовою виявилась комбінація рівня «Вживання алкоголю» та рівня «Професійних ризиків».

Далі проведемо аналіз впливовості ознак за допомогою бібліотеки SHAP. Бібліотека SHAP є одним з найефективніших інструментів для інтерпретації роботи прогностичних моделей штучного інтелекту, зокрема алгоритмів машинного навчання. Методологія SHAP використовує у своєму арсеналі так звану концепцію значень Шеплі, що походить із кооперативної теорії ігор, і дозволяє визначити внесок кожної окремої ознаки у формування кінцевого прогнозу моделі. Такий підхід дає можливість детально проаналізувати вплив кожної ознаки на результат передбачення, а також виявити взаємодії між ознаками, що можуть суттєво змінювати поведінку моделі. Завдяки цьому можна отримати прозоре та глибоке розуміння механізмів прийняття рішень навіть у складних моделей машинного навчання (рис. 3.58).

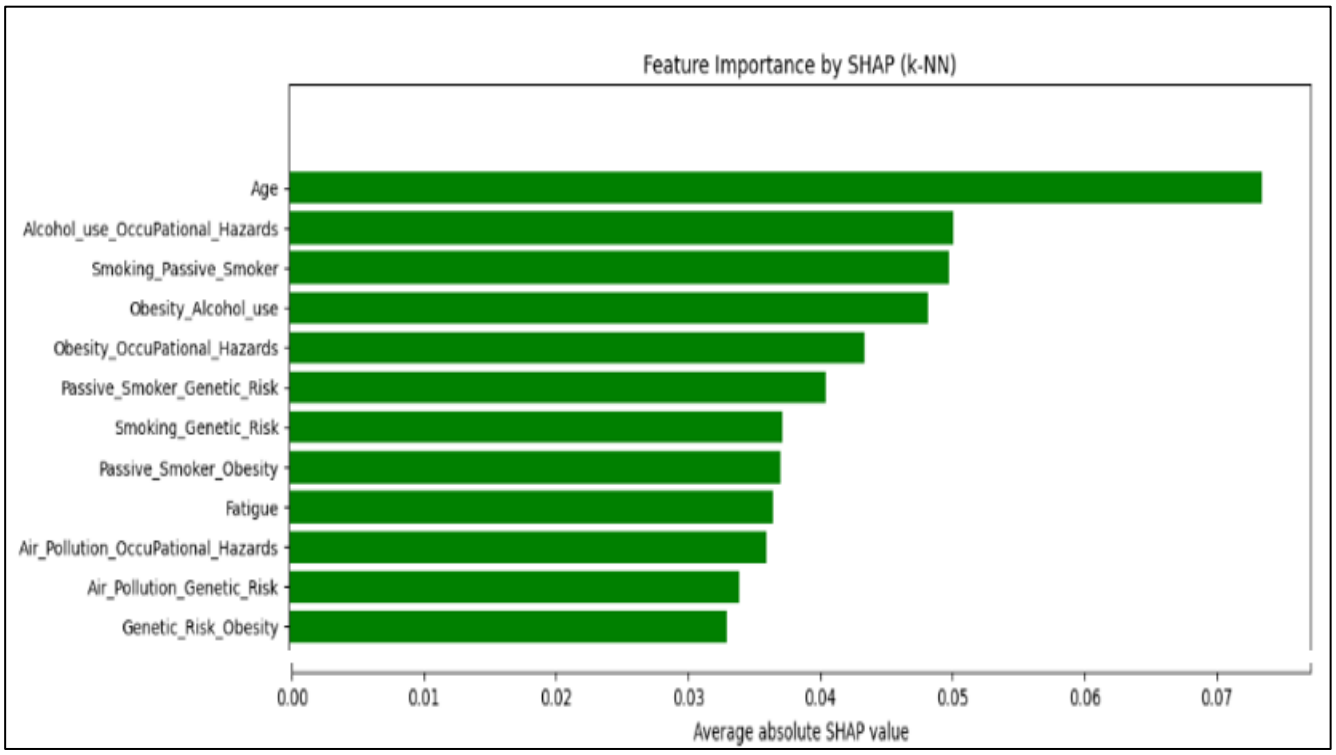

Рисунок 3.58 – Графік оцінки впливовості ознак на модель k-NN за допомогою бібліотеки SHAP

Рисунок 3.58 показує, що бібліотека SHAP також визначила найвпливовішими саме нові згенеровані ознаки, серед них збігаються із результатами permutation_importance комбінації рівня «Вживання алкоголю» та рівня «Професійних ризиків», рівня «Куріння» та «Пасивного куріння», рівень «Ожиріння» та «Професійних ризиків». Попри це найбільш впливовою ознакою за алгоритмом SHAP є саме вікова категорія пацієнтів.

3.7 Висновки

У цьому розділі здійснено розроблення та експериментальну перевірку інформаційної технології для передбачення ризику розвитку раку легень із застосуванням сучасних методів машинного навчання. Реалізовано та протестовано низку моделей, серед яких Random Forest, AdaBoost, Extra Trees, Decision Tree, XGBoost, Multi-Layer Perceptron, Logistic Regression, Support Vector Machines, K-Neighbors, LGBM та двошарова штучна нейронна мережа. Для кожного алгоритму проведено оптимізацію гіперпараметрів з метою підвищення точності прогнозування.

Виконано порівняння продуктивності моделей на основі метрики F1-Score, що дозволило оцінити збалансованість класифікації між усіма класами цільової змінної. Усі моделі продемонстрували високий рівень точності — вище 90%, що свідчить про коректність обраного підходу до підготовки даних і навчання моделей. Порівняння з результатами попередніх досліджень підтвердило підвищення ефективності більшості моделей, що є свідченням результативності застосованих етапів удосконалення інформаційної технології.

Найкращим класифікатором виявився алгоритм K-Nearest Neighbors, який досяг F1-Score на рівні 99% для тестових даних. Це дозволяє розглядати модель KNN як найбільш придатну для подальшого практичного використання в межах розробленої технології.

Крім того, проведено оцінювання важливості ознак для прогнозування цільової змінної з використанням методів permutation importance та аналізу SHAP-значень. Результати інтерпретації моделей показали, що найбільший внесок у формування прогнозу роблять згенеровані ознаки, зокрема комбінований показник, що поєднує рівень споживання алкоголю та професійних ризиків. Це підтверджує доцільність створення похідних ознак як інструменту підвищення якісних характеристик моделі.

4. ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу так і в плані економіки. Тому для науково-дослідної роботи необхідно провести аналіз та оцінку економічної ефективності результатів проведеної роботи.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання» є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями [25].

Таблиця 4.1 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	4	5	4
2. Ринкові переваги (наявність аналогів)	4	4	4
3. Ринкові переваги (ціна продукту)	3	4	4
4. Ринкові переваги (технічні властивості)	4	3	4
5. Ринкові переваги (експлуатаційні витрати)	3	4	4

Продовження таблиці 4.1

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
6. Ринкові перспективи (розмір ринку)	2	3	1
7. Ринкові перспективи (конкуренція)	4	3	2
8. Практична здійсненність (наявність фахівців)	4	5	4
9. Практична здійсненність (наявність фінансів)	2	3	2
10. Практична здійсненність (необхідність нових матеріалів)	3	4	3
11. Практична здійсненність (термін реалізації)	5	4	4
12. Практична здійсненність (розробка документів)	3	3	4
Сума балів	41	45	*40*
Середньоарифметична сума балів $СБ_с$	42		

За результатами розрахунків, наведених в таблиці 4.1, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки.

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання» становить 42 бали, що, відповідно до [25], свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки високий).

4.2 Розрахунок узагальненого коефіцієнта якості розробки

Узагальнений коефіцієнт якості ($B_н$) для нового технічного рішення розрахуємо за формулою [26]:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i \quad , \tag{4.1}$$

де k – кількість найбільш важливих технічних показників, які впливають на якість нового технічного рішення;

α_i – коефіцієнт, який враховує питому вагу *i-го* технічного показника в загальній якості розробки. Коефіцієнт α_i визначається експертним шляхом і при цьому має виконуватись умова $\sum_{i=1}^{k} \alpha_i = 1$;

β_i – відносне значення *i-го* технічного показника якості нової розробки.

Результати порівняння зведемо до таблиці 4.2.

Таблиця 4.2 – Порівняння основних параметрів розробки та аналога.

Показники (параметри)	Одиниця вимірю-вання	Аналог	Проектований продукт	Відношення параметрів нової розробки до аналога	Питома вага показника
1. Кількість використаних моделей машинного навчання	од.	6	11	1,8	0,25
2. Попередня обробка та очистка даних	од.	1	3	3	0,15
3. Точність передбачення	%	96	99	1,03	0,3
4. Кількість графіків розвідувального аналізу	од.	15	22	1,46	0,15
5. Кількість багатошарових нейронних мереж	од.	1	2	2	0,15

Узагальнений коефіцієнт якості (Вн) для нового технічного рішення складе:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i = 1{,}8 \cdot 0{,}25 + 3 \cdot 0{,}15 + 1{,}03 \cdot 0{,}3 + 1{,}46 \cdot 0{,}15 + 2 \cdot 0{,}15 = 1{,}72.$$

Отже за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 1,72 рази.

4.3 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи на тему «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

4.3.1 Витрати на оплату праці

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [25]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \quad (4.2)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=20 день.

$$З_о = 24700{,}00 \cdot 10 / 20 = 12350{,}00 \; грн.$$

Проведені розрахунки зведемо до таблиці 4.3.

Таблиця 4.3 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту (проєктний менеджер)	24700,00	1235,00	10	12350,00
Консультант (лікар онколог вищої категорії)	22300,00	1115,00	6	6690,00

Продовження таблиці 4.3

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Інженер-програміст	21000,00	1050,00	25	26250,00
Інженер-аналітик (системний аналіз)	20600,00	1030,00	10	10300,00
Консультант-аналітик цифрових обчислюваних систем	23900,00	1195,00	5	5975,00
Всього				61565,00

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання» розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i, \qquad (4.3)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_р \cdot t_{зм}}, \qquad (4.4)$$

де M_M – розмір мінімальної місячної заробітної плати, приймемо $Мм$=8000,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення (табл. Б.2, додаток Б) [25];

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок;

T_p – середнє число робочих днів в місяці, приблизно T_p = 20 дн;

$t_{зм}$ – тривалість зміни, год.

$$C_1 = 8000{,}00 \cdot 1{,}10 \cdot 1{,}15 / (20 \cdot 8) = 63{,}25 \text{ грн.}$$

$$З_{р1} = 63{,}25 \cdot 8{,}00 = 506{,}00 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці 4.4.

Таблиця 4.4 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Монтаж робочого місця розробника системи прогнозування	10,00	2	1,10	63,25	632,00
Інсталяція програмного забезпечення	6,00	5	1,70	97,75	586,50
Встановлення цифрових обчислювальних систем	3,00	4	1,50	86,25	258,75
Налаштування модулів	8,00	5	1,70	97,75	782,00
Тренування цифрової експериментальної моделі	4,00	4	1,50	86,25	345,00
Формування бази даних прогнозного аналізу	14,00	3	1,35	77,63	1086,82
Інші допоміжні роботи	15,00	3	1,35	77,63	1164,45
Монтаж серверного обладнання	5	4	1,50	86,25	431,25
Всього					5286,77

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (4.5)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 10%.

$З_{дод}$ = (61565,00 + 5286,77) · 11 / 100% = 7353,70 грн.

4.3.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (4.6)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (61565,00 + 5286,77 + 7353,70) · 22 / 100% = 14707,39 грн.

4.3.3 Сировина та матеріали

Витрати на матеріали (*М*), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{в\,j}, \quad (4.7)$$

де $Н_j$ – норма витрат матеріалу *j*-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу *j*-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*);

$В_j$ – маса відходів *j*-го найменування, кг;

$Ц_{вj}$ – вартість відходів *j*-го найменування, грн/кг.

$М_1$ = 3,000 · 163,00 · 1,05 - 0 · 0 = 513,45 грн.

Проведені розрахунки зведемо до таблиці 4.5.

Таблиця 4.5 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 од, грн	Норма витрат, од.	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Папір канцелярський офісний Maestro Standart (A4-500)	163,00	3,000	0	0	513,45
Начиння канцелярське Buromax	210,00	3,000	0	0	661,50
Органайзер офісний Buromax	228,00	3,000	0	0	718,20
Картридж для принтера Canon CLI-471Bk	1349,00	2,000	0	0	2832,90
Диск оптичний Sony (CD-R)	30,00	4,000	0	0	126,00
Диск оптичний Sony (CD-RW)	35,00	4,000	0	0	147,00
FLASH-пам'ять Kingstar (32 ГБ) Class 10	220,00	1,000	0	0	231,00
FLASH-пам'ять Kingstar (64 ГБ) Class 10 A	350,00	1,000	0	0	367,50
Всього					5597,55

4.3.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Інформаційна технологія аналізу та передбачення раку легень методами

машинного навчання», розраховуємо, згідно з їхньою номенклатурою, за формулою:

$$К_в = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j \quad (4.8)$$

де $Н_j$ – кількість комплектуючих j-го виду, шт.;

$Ц_j$ – покупна ціна комплектуючих j-го виду, грн;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 … 1,15*).

$К_в = 2 \cdot 220,00 \cdot 1,05 = 462,00$ *грн.*

Проведені розрахунки зведемо до таблиці 4.6.

Таблиця 4.6 – Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Atcom USB to RS232 0.85m	2	220,00	462,00
Концентратор Defender SEPTIMA SLIM USB 2.0	1	675,00	708,75
Пам'ять (SSD диск) Kingston NV3 M.2 2280 NVMe PCIe 4.0 (SNV3S/1000G)	4	3389,00	14 233,80
Всього			15404,55

4.3.5 Спецустаткування для наукових (експериментальних) робіт

Балансову вартість спецустаткування розраховуємо за формулою:

$$В_{спец} = \sum_{i=1}^{k} Ц_i \cdot С_{пр.i} \cdot К_i \ , \quad (4.9)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$С_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

$К_i$ – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, ($К_i$ = *1,10…1,12);*

k – кількість найменувань устаткування.

$B_{спец} = 23765,00 \cdot 1 \cdot 1,05 = 24953,25$ *грн.*

Отримані результати зведемо до таблиці 4.7.

Таблиця 4.7 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Серверне обладнання на основі ПК AMD Ryzen 5 3500U + Radeon Vega Mobile Gfx 2GB	1	23765,00	24953,25
Маршрутизатор WiFi4 TP-Link TL-WR840N V6	1	730,00	766,50
ПК AMD Ryzen 7 8700F + GeForce RTX 5070 12GB	1	58700	61635,00
Всього			87354,75

4.3.6 Програмне забезпечення для наукових (експериментальних) робіт

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.10)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань програмних засобів.

$B_{прг} = 8910,00 \cdot 1 \cdot 1,05 = 9355,50$ *грн.*

Отримані результати зведемо до таблиці 4.8.

Таблиця 4.8 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Середовище мови програмування Python	1	8910,00	9355,50
Середовище розробки програмного забезпечення PyCharm	1	7600,00	7980,00
Програмне забезпечення розробки Kaggle	1	1099,00	1153,95
Високошвидкісний доступ до мережі Internet (міс)	2	455,00	955,50
Бібліотеки для машинного навчання Scikit-Learn, PyTorch, TensorFlow	1	6350,00	6667,50
Всього			26112,45

4.3.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.11)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (40799{,}00 \cdot 2) / (3 \cdot 12) = 2266{,}61$ *грн.*

Проведені розрахунки зведемо до таблиці 4.9.

Таблиця 4.9 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Програмно-аналітичний комплекс ПК AMD Ryzen 5/ RAM 16ГБ / SSD 1ТБ / nVidia GeForce RTX 5080 12ГБ	40799,00	3	2	2266,61
Персональний комп'ютер Ноутбук HP Laptop 15-fc0142ua (B9PF5EA)	15999,00	3	2	888,83
Спеціалізоване робоче місце розробника інформаційної технології	6700,00	5	2	223,30
Пристрій виводу текстової інформації Принтер Canon PIXMA MG3640S	4499,00	3	2	249,94
Оргтехніка Samsung Office	12850,00	5	2	428,30
ОС Windows 11 Pro	10999,00	3	2	611,05
Прикладний пакет Microsoft Office 2024 Home & Business	9875,00	3	2	548,61
Мережеве обладнання передачі цифрових даних	5700,00	4	2	237,50
Всього				5454,14

4.3.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (4.12)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; приймемо Цe = 12,56 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,58 \cdot 240,0 \cdot 12,56 \cdot 0,95 / 0,97 = 1712,30$ *грн.*

Проведені розрахунки зведемо до таблиці 4.10.

Таблиця 4.10 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Програмно-аналітичний комплекс ПК AMD Ryzen 5/ RAM 16ГБ / SSD 1ТБ / nVidia GeForce RTX 5080 12ГБ	0,58	240,0	1712,30
Персональний комп'ютер Ноутбук HP Laptop 15-fc0142ua (B9PF5EA)	0,05	240,0	147,61
Спеціалізоване робоче місце розробника інформаційної технології	0,07	240,0	206,66
Пристрій виводу текстової інформації Принтер Canon PIXMA MG3640S	0,02	4	0,98
Мережеве обладнання передачі цифрових даних	0,05	240,0	147,61
Серверне обладнання на основі ПК AMD Ryzen 5 3500U + Radeon Vega Mob 2GB	0,02	240,0	59,04
Всього			2274,20

4.3.9 Службові відрядження

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{Н_{св}}{100\%}, \quad (4.13)$$

де $Н_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 20%.5286,77

$В_{св}$ = (61565,00 + 5286,77) · 20 / 100% = 13370,35 грн.

4.3.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (4.14)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$В_{сп}$ = (61565,00 + 5286,77) · 30 / 100% = 20055,53 грн.

4.3.11 Інші витрати

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.15)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$І_в$ = (61565,00 + 5286,77) · 50 / 100% = 33425,88 грн.

4.3.12 Накладні (загальновиробничі) витрати

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.16)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 110%.

$В_{нзв}$ = (61565,00 + 5286,77) · 110 / 100% = 73536,95 грн.

Витрати на проведення науково-дослідної роботи на тему «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (4.17)$$

$В_{заг}$= 61565,00 +5286,77 + 7353,70 + 14707,39 + 5597,55 + 15404,55 + 87354,75 + 26112,45 + 5454,14 + 2274,20 + 13370,35 + 20055,53 + 33425,88 + 73536,95 = 371499,21 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (4.18)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,9.

ЗВ = 371499,21 / 0,9 = 412776,90 грн.

4.4 Розрахунок економічної ефективності науково-технічної розробки при її можливій комерціалізації потенційним інвестором

Результати дослідження проведені за темою «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання» передбачають комерціалізацію протягом 4-х років реалізації на ринку.

В цьому випадку основу майбутнього економічного ефекту будуть формувати:

ΔN – збільшення кількості споживачів яким надається відповідна інформаційна послуга у періоди часу, що аналізуються;

Кількість споживачів приведена в таблиці 4.11.

Таблиця 4.11 – Кількість споживачів

Показник	1-й рік	2-й рік	3-й рік	4-й рік
Збільшення кількості споживачів, осіб	300	500	800	400

N – кількість споживачів яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки, приймемо 2300 осіб;

$Ц_б$ – вартість послуги у році до впровадження інформаційної системи, приймемо 7800,00 грн;

$\pm\Delta Ц_o$ – зміна вартості послуги від впровадження результатів, приймемо 758,00 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 4-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [25]:

$$\Delta П_i = (\pm \Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (4.19)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2025 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо $\rho = 39\%$;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2025 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1$ =(758,00·2300,00+8558,00·300)·0,83·0,39·(1-0,18/100%)= 1412959,82 грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2$ =(758,00·2300,00+8558,00·800)·0,83·0,39·(1-0,18/100%)= 2795578,91 грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3$ =(758,00·2300,00+8558,00·1600)·0,83·0,39·(1-0,18/100%)= 5007769,47 грн.

Збільшення чистого прибутку 4-го року:

$\Delta П_4$ =(758,00·2300,00+8558,00·2000)·0,83·0,39·(1-0,18/100%)= 6113864,75 грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.20)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=0,1;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = 1412959,82 /(1+0,1)^1+2795578,91 /(1+0,1)^2+5007769,47 /(1+0,1)^3+$$
$$+6113864,75/(1+0,1)^4=1284508,93+2310395,79+3762411,32+4175851,89=$$
$$=11533167,93 \text{ грн.}$$

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \quad (4.21)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=2;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 413746,87 грн.

$PV = k_{інв} \cdot ЗВ = 2 \cdot 412776,90 = 825553,80$ грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \quad (4.22)$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 11533167,93 грн;

PV – теперішня вартість початкових інвестицій, 825553,80 грн.

$E_{абс} = ПП - PV = 11533167,93 - 825553,80 = 10707614,13$ грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.23)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 10707614,13 грн;

PV – теперішня вартість початкових інвестицій, 825553,80 грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 4 роки.

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+10707614,13/825553,80)^{1/4} - 1 = 0,93.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \quad (4.24)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2025 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,35.

$\tau_{мін}$= 0,1+0,25 = 0,45 < 0,93 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \quad (4.25)$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$$T_{ок} = 1 / 0{,}93 = 1{,}07 \text{ р.}$$

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

4.5 Висновки

Результати дослідження проведені за темою «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання» показали що рівень комерційного потенціалу розробки 42 бали, що свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки високий).

При оцінюванні за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 1,72 рази.

Також термін окупності становить 1,07 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання».

ВИСНОВКИ

В процесі виконання магістерської кваліфікаційної роботи було здійснено ґрунтовний аналіз проблематики діагностики та лікування раку легень у сучасних умовах, що дозволило визначити ключові недоліки наявних підходів. Встановлено, що традиційні методи діагностування часто забезпечують виявлення захворювання на пізніх стадіях, що зумовлює високий рівень смертності та потребує впровадження інноваційних технологій для раннього прогнозування ризиків розвитку легеневої онкології.

Проведено аналіз основних факторів, що впливають на ймовірність виникнення раку легень, із виокремленням найбільш критичних для подальшого моделювання. Додатково опрацьовано інструментарій, необхідний для реалізації інформаційної технології, зокрема програмну мову Python та платформу Kaggle, що забезпечують широкі можливості для проведення аналізу даних і побудови моделей машинного навчання. Також розглянуто та оцінено релевантні методи машинного навчання, які доцільно застосувати для створення інформаційної технології прогнозування ризику раку легень.

Було виконано процес попередньої підготовки даних, а саме здійснено очищення датасету шляхом видалення неінформативних полів, а також перетворення категоріальних значень цільової змінної «Level» у числовий формат, що забезпечило можливість коректного застосування алгоритмів машинного навчання. Проведено розвідувальний аналіз даних із використанням сучасних методів візуалізації, зокрема теплових карт і графічних представлень кількісних показників. Аналіз розподілу ознак підтвердив, що більшість пацієнтів з вибірки належать до середнього віку та зазнають впливу низки факторів ризику, таких як ожиріння, куріння та споживання алкоголю. Дослідження цільової ознаки показало рівномірний розподіл між усіма її унікальними класами. Крім того, проаналізовано кореляційні зв’язки між ознаками з метою визначення найбільш значущих чинників впливу на цільову змінну. На основі отриманих результатів виконано створення нових похідних ознак.

Здійснено розроблення та експериментальну перевірку інформаційної технології для передбачення ризику розвитку раку легень із застосуванням сучасних методів машинного навчання. Реалізовано та протестовано низку моделей, серед яких Random Forest, AdaBoost, Extra Trees, Decision Tree, XGBoost, Multi-Layer Perceptron, Logistic Regression, Support Vector Machines, K-Neighbors, LGBM та двошарова штучна нейронна мережа. Для кожного алгоритму проведено оптимізацію гіперпараметрів з метою підвищення точності прогнозування [27].

Виконано порівняння продуктивності моделей на основі метрики F1-Score, що дозволило оцінити збалансованість класифікації між усіма класами цільової змінної. Усі моделі продемонстрували високий рівень точності — вище 90%, що свідчить про коректність обраного підходу до підготовки даних і навчання моделей. Порівняння з результатами попередніх досліджень підтвердило підвищення ефективності більшості моделей, що є свідченням результативності застосованих етапів удосконалення інформаційної технології.

Найкращим класифікатором виявився алгоритм K-Nearest Neighbors, який досяг F1-Score на рівні 99% для тестових даних. Це дозволяє розглядати модель KNN як найбільш придатну для подальшого практичного використання в межах розробленої технології.

Крім того, проведено оцінювання важливості ознак для прогнозування цільової змінної з використанням методів permutation importance та аналізу SHAP-значень. Результати інтерпретації моделей показали, що найбільший внесок у формування прогнозу роблять згенеровані ознаки, зокрема комбінований показник, що поєднує рівень споживання алкоголю та професійних ризиків. Це підтверджує доцільність створення похідних ознак як інструменту підвищення якісних характеристик моделі.

За результатами роботи було опубліковано тези на LV Всеукраїнській науково-технічній конференції підрозділів Вінницького національного технічного університету (м. Вінниця, 2025-2026 рр.).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Неволя С. Д., Жуков С. О. Тренування моделей для інформаційної технології аналізу та передбачення раку легень методами машинного навчання. . *LV Всеукраїнська науково-технічна конференція підрозділів Вінницького національного технічного університету (2025-2026)*, Вінниця 2025. URL: https://conferences.vntu.edu.ua/index.php/all-fksa/all-fksa-2026/paper/view/26557. (дата звернення: 03.12.2025).
2. Прогнозування захворюваності на рак, ВООЗ. URL: https://phc.org.ua/news/vooz-prognozue-zrostannya-zakhvoryuvanosti-na-rak-yak-zniziti-rizik-rozvitku-onkonedugi. (дата звернення: 15.10.2025).
3. Що важливо знати про рак легень. URL: https://www.phc.org.ua/news/scho-vazhlivo-znati-pro-rak-legen. (дата звернення: 17.10.2025).
4. Путівник мовою програмування Python. URL: https://pythonguide.rozh2sch.org.ua/#_короткий_опис. (дата звернення: 19.10.2025).
5. Популярність Python. URL: http://www.itschool.vn.ua/the-popularity-of-python/. (дата звернення: 21.10.2025).
6. Scikit-learn Machine Learning in Python. URL: https://scikit-learn.org/stable/index.html. (дата звернення: 22.10.2025).
7. Kaggle Wiki. URL: https://uk.wikipedia.org/wiki/Kaggle. (дата звернення: 22.10.2025).
8. Dey, D., Haque, M.S., Islam, M.M. *et al.* The proper application of logistic regression model in complex survey data: a systematic review. *BMC Med Res Methodol.* 2025, 25p. URL: https://doi.org/10.1186/s12874-024-02454-5. (дата звернення: 29.10.2025).
9. Xiaochuan Gao, Weiting Bai, Qianlong Dang, Shuai Yang, Guanghui Zhang. Learnable self-supervised support vector machine based individual selection strategy for multimodal multi-objective optimization. *Elsevier,* 2025, 63-73p. URL: https://doi.org/10.1016/j.ins.2024.121553. (дата звернення: 29.10.2025).

10. Bouke, M.A., Alramli, O.I. & Abdullah, A. XAIRF-WFP: a novel XAI-based random forest classifier for advanced email spam detection. *Int. J. Inf. Secur.* 2025, 24-25p. URL: https://doi.org/10.1007/s10207-024-00920-1. (дата звернення: 29.10.2025).
11. Elshewey, A.M., Selem, E. & Abed, A.H. Improved CKD classification based on explainable artificial intelligence with extra trees and BBFS. *Sci Rep* 2025, 15p. URL: https://doi.org/10.1038/s41598-025-02355-7. (дата звернення: 29.10.2025).
12. Hui Chen, Zelong Lin, Zhongming Chen, Junkang Jian, Chang Liu. Adaptive DWA algorithm with decision tree classifier for dynamic planning in USV navigation. *Elsevier,* 2025*,* 94p. URL: https://doi.org/10.1016/j.oceaneng.2025.120328. (дата звернення: 29.10.2025).
13. Zhang, X., Yu, L. & Yin, H. An Ensemble Resampling Based Transfer AdaBoost Algorithm for Small Sample Credit Classification with Class Imbalance. *Comput Econ* 2025, 65p. URL: https://doi.org/10.1007/s10614-024-10690-6. (дата звернення: 29.10.2025).
14. Mengyuan He, Hong Liu, Shan Zhou, Yan Yao, Risto Kosonen, Yuxin Wu, Baizhan Li. Machine learning-based assessment of thermal comfort for the elderly in warm environments: Combining the XGBoost algorithm and human body exergy analysis. *Elsevier,* 2025*,* 209p. URL: https://doi.org/10.1016/j.ijthermalsci.2024.109519. (дата звернення: 29.10.2025).
15. J. Vijaya, Nagaraju Jajam, Debashish Padhy. Fine-Tuning Multilayer Perceptron Classifiers for Enhanced Heart Disease Prediction. *IEEEl,* 2025, 51p. URL: https://doi.org/10.1109/IATMSI64286.2025.10984498. (дата звернення: 29.10.2025).
16. Мокін В. Б., Дратований М. В. Наука про дані: машинне навчання та інтелектуальний аналіз даних. Вінниця: ВНТУ, 2024. – 258с. URL: https://docs.vntu.edu.ua/card.php?id=8163. (дата звернення: 03.11.2025).
17. Liu, C., Yang, W. Transformer fault diagnosis using machine learning: a method combining SHAP feature selection and intelligent optimization of LGBM. *Energy*

Inform 2025, 52p. URL: https://doi.org/10.1186/s42162-025-00519-3. (дата звернення: 06.11.2025).

18. Changjun Wang, Fengxia You, Yu Wang. Text intelligent classification model based on improved KNN algorithm in library service transformation. *Elsevier,* 2025*,* 7p. URL: https://doi.org/10.1016/j.sasc.2025.200313. (дата звернення: 15.11.2025).
19. Lung Cancer Prediction Dataset. Kaggle, 2023. URL: https://www.kaggle.com/datasets/thedevastator/cancer-patients-and-air-pollution-a-new-link. (дата звернення: 15.11.2025).
20. Кореляційна матриця Пірсона та Спірмена. URL: https://www.guru99.com/uk/r-pearson-spearman-correlation.html . (дата звернення: 16.11.2025).
21. Lung Cancer Prediction by SUBHAJEET DAS. Kaggle, 2023. URL: https://www.kaggle.com/code/subhajeetdas/lung-cancer-prediction . (дата звернення: 17.11.2025).
22. Feature engineering. URL: https://uk.wikipedia.org/wiki/Конструювання_ознак. (дата звернення: 17.11.2025).
23. UML Diagram Types Guide: Learn About All Types of UML Diagrams with Examples. URL: https://creately.com/blog/diagrams/uml-diagram-types-examples/. (дата звернення: 20.11.2025).
24. Used Cars : Analysis and Prediction by Vitalii Mokin. Kaggle. URL: https://www.kaggle.com/code/vbmokin/used-cars-analysis-and-prediction. (дата звернення: 21.11.2025).
25. В. О. Козловський, О. Й. Лесько, В. В. Кавецький. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт. Вінниця : ВНТУ, 2021. – 42 с. URL: https://epvm.vntu. edu.ua/wp-content/uploads/2021/09/Ekonom_magister_tehn_2021.pdf. (дата звернення: 24.11.2025).

26. В. В. Кавецький, В. О. Козловський, І. В. Причепа**.** Економічне обґрунтування інноваційних рішень: практикум. Вінниця: ВНТУ, 2016. – 113с. URL: https://epvm.vntu.edu.ua/wp-content/uploads/2018/01/EOIP_prakt_kavetsky.pdf. (дата звернення: 24.11.2025).
27. Lung Cancer Prediction 3.0 by SergiiN. Kaggle, 2025. URL: https://www.kaggle.com/sergiin/lung-cancer-prediction-3-0. (дата звернення: 16.11.2025).

Додаток А

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ЗАТВЕРДЖУЮ
Завідувач кафедри САІТ
____ д.т.н., проф. Віталій МОКІН
«__» _____2025 року

ТЕХНІЧНЕ ЗАВДАННЯ

на магістерську кваліфікаційну роботу

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ АНАЛІЗУ ТА ПЕРЕДБАЧЕННЯ РАКУ ЛЕГЕНЬ МЕТОДАМИ МАШИННОГО НАВЧАННЯ»

08-34.МКР.002.02.000 ТЗ

Керівник: к.т.н., доц. каф. САІТ
_____ Сергій ЖУКОВ
«__» _____ 2025 р.

Розробив студент гр. 2ІСТ-24м
_____Сергій НЕВОЛЯ
«__» _____ 2025 р.

Вінниця 2025

1. Підстава для проведення робіт.
Підставою для виконання роботи є наказ №__ по ВНТУ від «__» _____2025р., та індивідуальне завдання на МКР, затверджене протоколом №__ засідання кафедри САІТ від «__» _____ 2025р.
2. Джерела розробки:
1) Мокін В. Б., Дратований М. В. Наука про дані: машинне навчання та інтелектуальний аналіз даних. Вінниця: ВНТУ, 2025. 258 с. URL: https://docs.vntu.edu.ua/card.php?id=8163.
2) Lung Cancer Prediction Dataset. Kaggle, 2023. URL: https://www.kaggle.com/datasets/thedevastator/cancer-patients-and-air-pollution-a-new-link.
3. Мета і призначення роботи.
Метою дослідження є підвищення точності передбачення ризику захворіти на рак легень шляхом розроблення інформаційної технології.
4. Вихідні дані для проведення робіт.
Набір даних «Lung Cancer Prediction» з відкритої бібліотеки платформи Kaggle.
5. Методи дослідження.
Аналіз і синтез наукових джерел, методи машинного навчання, статистичний аналіз даних, розвідувальний аналіз даних (EDA), методи візуалізації даних, оптимізація гіперпараметрів, порівняльний аналіз моделей, оцінювання ефективності за якісними метриками.
6. Етапи роботи і терміни їх виконання:
а) Аналіз предметної області передбачення раку легень _____ – _____
б) Розвідувальний аналіз та обробка даних _____ – _____
в) Розробка інформаційної технології та аналіз результатів _____ – _____
г) Економічна частина _____ – _____
д) Оформлення матеріалів до захисту МКР _____ – _____
7. Очікувані результати та порядок реалізації.
Очікуваним результатом є створення інформаційної технології, здатної з високою точністю прогнозувати ризик розвитку раку легень на основі медичних та поведінкових даних пацієнтів.
8. Вимоги до розробленої документації.
Текстова та ілюстративна частини роботи оформлені у відповідності до вимог «Методичних вказівок до виконання магістерських кваліфікаційних робіт для студентів спеціальності 126 «Інформаційні системи та технології» (освітня програма «Інформаційні технології аналізу даних та зображень»).
9. Порядок приймання роботи.
Публічний захист «___» _____ 2025 р.
Початок розробки «___» _____ 2025 р.
Граничні терміни виконання МКР «___» _____ 2025 р.

Розробив студент групи 2ІСТ-24м _____ Сергій НЕВОЛЯ

Додаток Б

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: « Інформаційна технологія аналізу та передбачення раку легень методами машинного навчання»

Тип роботи: магістерська кваліфікаційна робота

Підрозділ: кафедра САІТ, ФІІТА, гр. 2ІСТ-24м

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism 5,11 %

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне):

- ■ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту
- □ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.
- □ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Віталій МОКІН, зав. каф. САІТ _____ (підпис)

Сергій ЖУКОВ, доц. каф. САІТ _____ (підпис)

Особа, відповідальна за перевірку _____ (підпис) Сергій ЖУКОВ

З висновком експертної комісії ознайомлений(-на)

Керівник _____ (підпис) Сергій ЖУКОВ, к.т.н., доц. каф. САІТ

Здобувач _____ (підпис) Сергій НЕВОЛЯ

Додаток В

Лістинг програми

```
!pip install dtreeviz

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import xgboost as xgb

import matplotlib.gridspec as gridspec

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier,
ExtraTreesClassifier

from sklearn.tree import DecisionTreeClassifier

from sklearn.neural_network import MLPClassifier

from xgboost import XGBClassifier

from sklearn.metrics import precision_score, recall_score, accuracy_score, r2_score,
f1_score, confusion_matrix, ConfusionMatrixDisplay, classification_report

from sklearn import preprocessing

from sklearn.model_selection import GridSearchCV

import shap

import warnings

warnings.filterwarnings("ignore")

df = pd.read_csv("/kaggle/input/cancer-patients-and-air-pollution-a-new-link/cancer
patient data sets.csv")

df.drop(columns=['index', 'Patient Id'], axis=1, inplace=True)

df.describe()

# Replace "level" with Integer

print('Cancer Levels: ', df['Level'].unique())
```

```
df["Level"].replace({'High': 2, 'Medium': 1, 'Low': 0}, inplace=True)
print('Cancer Levels: ', df['Level'].unique())
df.info()
EDA
plt.figure(figsize=(10,4))
plt.boxplot(df['Age'], vert=False, patch_artist=True,
            boxprops=dict(facecolor='skyblue', linewidth=2),
            whiskerprops=dict(color='green',linewidth=3),
            medianprops=dict(color='red',linewidth=2)
           )
plt.title('Розподіл даних по віку пацієнтів')
plt.xticks(np.arange(10, max(df['Age'])+1, 3))
plt.show()
import statsmodels.api as sm
data = data = df['Age'].dropna()
# Побудова QQ-графіку
sm.qqplot(data, line='s')
plt.title("Розподіл даних для стовбця 'Age'")
plt.show()
df_corr = df.corr()
df_corr
plt.title("Correlation Matrix")
sns.heatmap(df_corr, cmap='viridis')
print('\n')
plt.figure(figsize=(20,15))
sns.heatmap(df.corr(), annot=True, cmap=plt.cm.PuBu)
plt.show()
print('\n')
```

```
for i in range(24):
    plt.subplot(16, 2, i+1)
    sns.distplot(df.iloc[:, i], color = 'red')
    plt.grid()
plt.figure(figsize = (15,7))
colors = ['red', 'yellow', 'green']
plt.title("Lung Cancer Chances ")
plt.pie(df['Level'].value_counts(), explode = (0.1, 0.02, 0.02), labels = ['High',
'Medium', 'Low'], autopct = "%1.2f%%", shadow = True, colors = colors)
plt.legend(title = "Lung Cancer Chances", loc = "lower left")
sns.displot(df['Level'], kde=True, color = 'red')
dfviz = df.copy()
y = df.pop('Level')
x = df
Train & Test Splitting the Data
# Data splitting
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
# Results of prediction
results = pd.DataFrame(columns = ['model', 'f1_train', 'f1_test', 'r2_train', 'r2_test'])
Random Forest
from sklearn.model_selection import GridSearchCV
param_RF = {
    'n_estimators': [50, 100, 150],
    'max_depth': [2],
    'criterion':['gini'],
    'min_samples_split': [2, 3],
    'min_samples_leaf': [2, 3],
    'random_state' : [42],
    'max_samples': [0.4]}
```

```
%%time
model_tuning = GridSearchCV(estimator=RandomForestClassifier(), param_grid=param_RF, cv=5)
model_tuning.fit(x_train, y_train)
best_params = model_tuning.best_params_
print("Best Parameters:", best_params)
model_rf = RandomForestClassifier(**best_params)
model_rf.fit(x_train, y_train)
# Train
train_predictions = model_rf.predict(x_train)
r2_train = r2_score(y_train, train_predictions)
f1_train = f1_score(y_train, train_predictions, average = 'micro')
# Test
test_predictions = model_rf.predict(x_test)
r2_test = r2_score(y_test, test_predictions)
f1_test = f1_score(y_test, test_predictions, average = 'micro')
#Result
performTrain(train_predictions)
performTest(test_predictions)
score_model_rf = model_rf.score(x_test, y_test)
ADABoost Classifier
param_ADA = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1, 1],
    'random_state' : [42],
    'algorithm': ['SAMME.R'],  }
%%time
model_tuning = GridSearchCV(estimator=AdaBoostClassifier(), param_grid=param_ADA, cv=5)
model_tuning.fit(x_train, y_train)
```

```
best_params = model_tuning.best_params_

print("Best Parameters:", best_params)

model_ada = AdaBoostClassifier(**best_params)

model_ada.fit(x_train, y_train)

# Train

train_predictions = model_ada.predict(x_train)

r2_train = r2_score(y_train, train_predictions)

f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test

test_predictions = model_ada.predict(x_test)

r2_test = r2_score(y_test, test_predictions)

f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result

performTrain(train_predictions)

performTest(test_predictions)

score_model_ada = model_ada.score(x_test, y_test)

Extra Trees Classifier

params_etc ={'n_estimators':[200, 300, 500],

             'max_depth': [3, 4, 5],

             'min_samples_split':  [2, 3],

             'min_samples_leaf':  [2, 3],

             'max_features': ['sqrt'],

             'ccp_alpha': [0.1],

             'random_state' : [42]}

%%time

model_tuning = GridSearchCV(estimator=ExtraTreesClassifier(), param_grid=params_etc, cv=5)

model_tuning.fit(x_train, y_train)

best_params = model_tuning.best_params_
```

```
print("Best Parameters:", best_params)

model_etc = ExtraTreesClassifier(**best_params)

model_etc.fit(x_train, y_train)

# Train

train_predictions = model_etc.predict(x_train)

r2_train = r2_score(y_train, train_predictions)

f1_train = f1_score(y_train, train_predictions, average = 'micro')

# Test

test_predictions = model_etc.predict(x_test)

r2_test = r2_score(y_test, test_predictions)

f1_test = f1_score(y_test, test_predictions, average = 'micro')

#Result

performTrain(train_predictions)

performTest(test_predictions)

score_model_etc = model_etc.score(x_test, y_test)

Decision Tree

params_DT ={'max_depth': [3, 4],

            'min_samples_split': [1, 2, 3],

            'min_samples_leaf': [0, 1, 2],

            'max_features': ['sqrt', 'log2', 'auto'],

            'max_leaf_nodes': [ 5, 10, 15, 17],

            'random_state' : [42]}
```

Додаток Г

(обов’язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПЕРЕДБАЧЕННЯ РАКУ ЛЕГЕНЬ МЕТОДАМИ МАШИННОГО НАВЧАННЯ

Нормоконтроль: к.т.н., доцент

_____ Сергій ЖУКОВ

«___» _____ 2025 р.

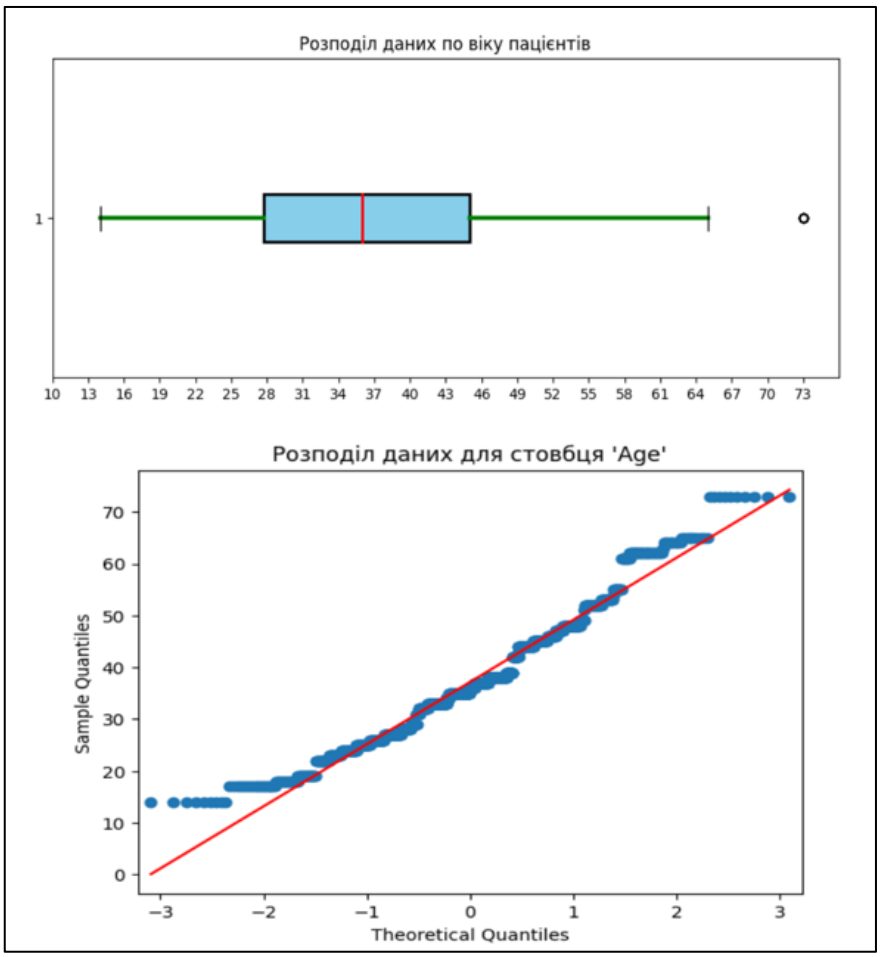

Рисунок Г.1 – Розподіл ознаки «Вік»

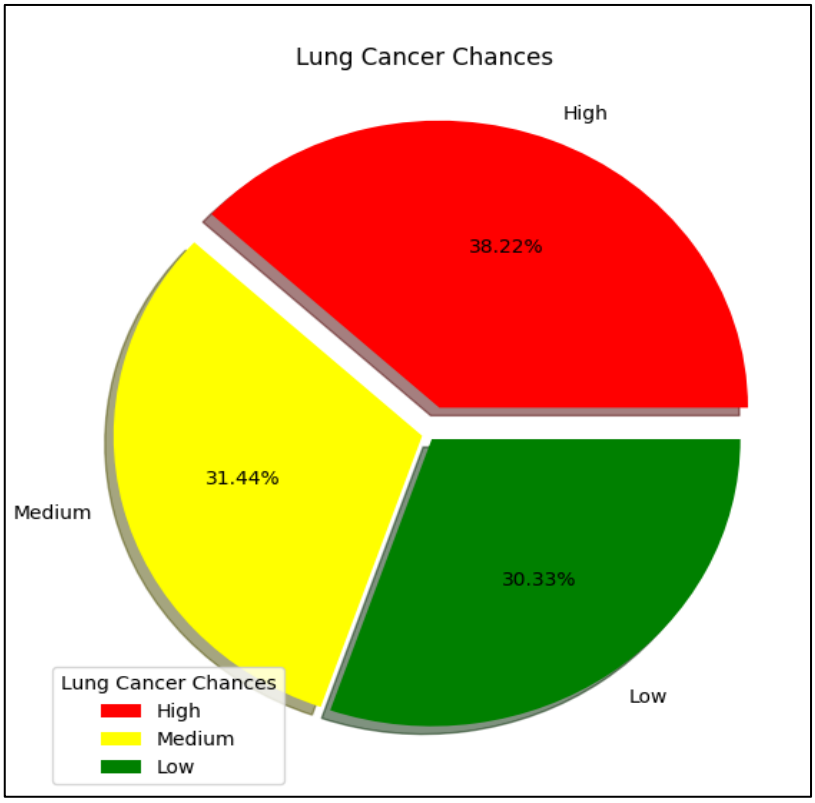

Рисунок Г.2 – Кругова діаграма розподілу цільової ознаки «Level»

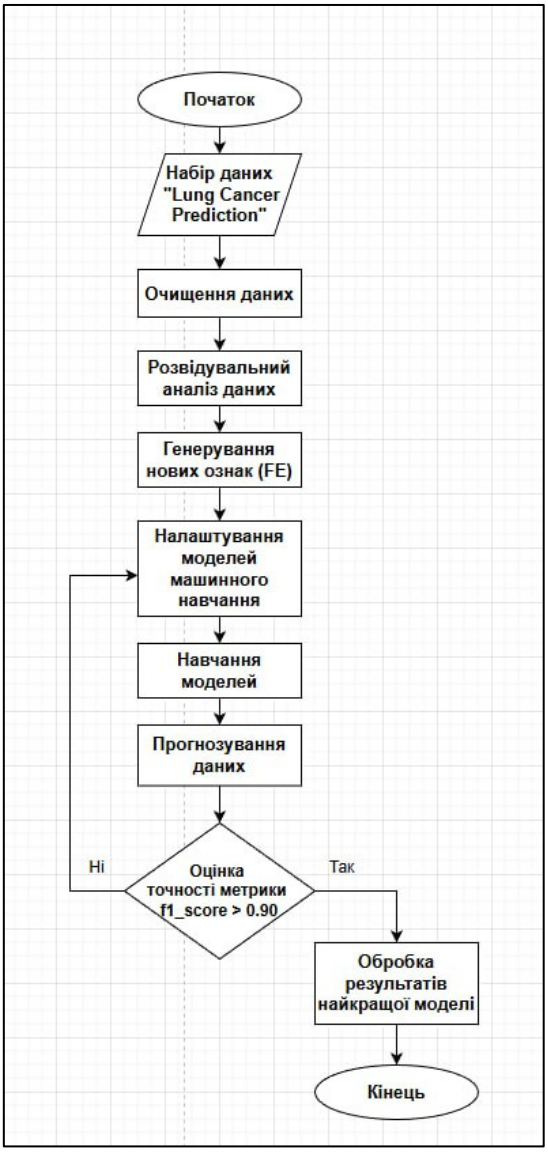

Рисунок Г.3 – Блок-схема алгоритму роботи інформаційної технології

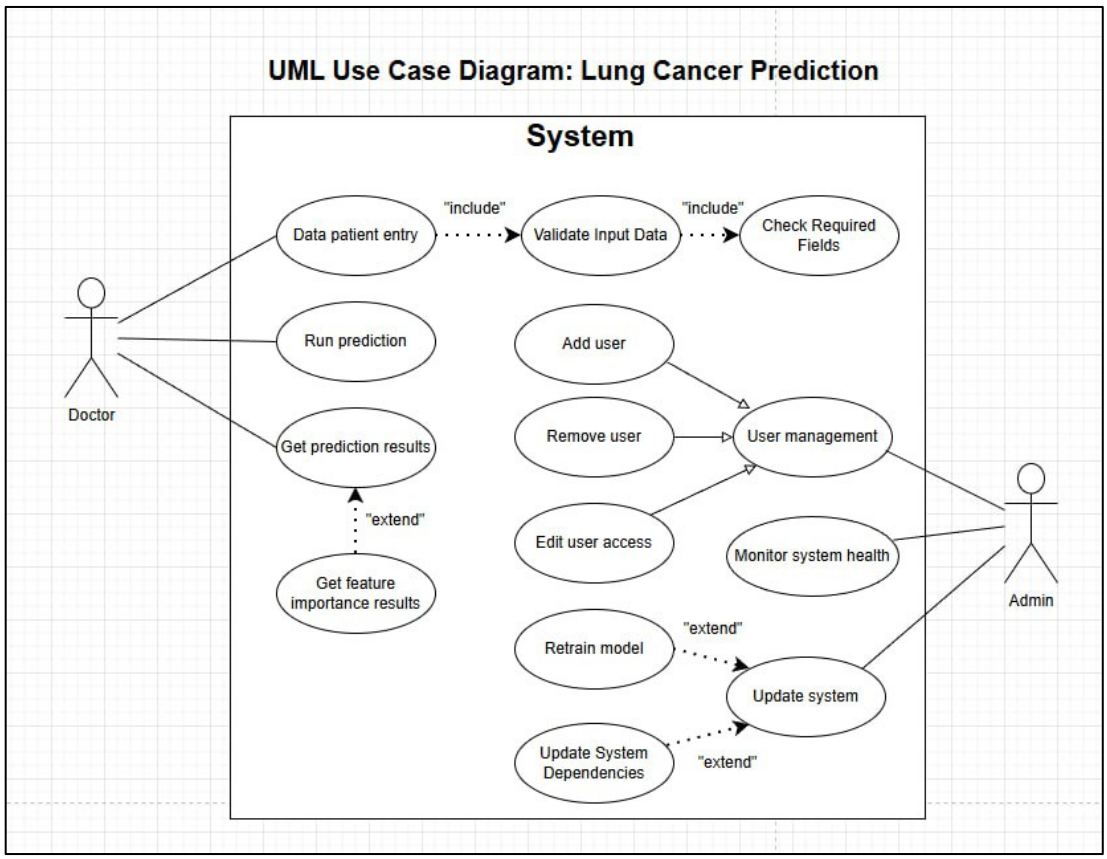

Рисунок Г.4 – UML Use Case діаграма

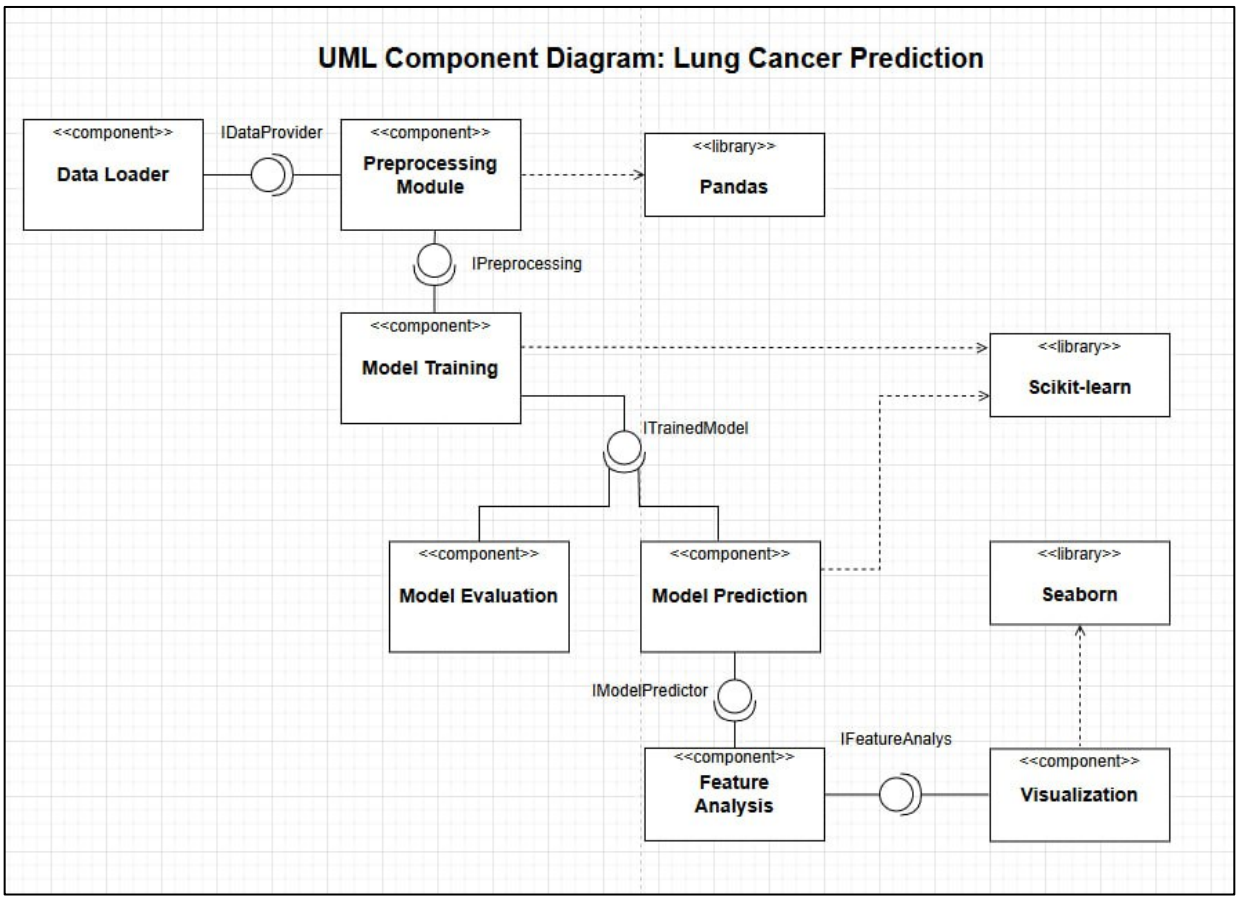

Рисунок Г.5 – UML діаграма компонентів

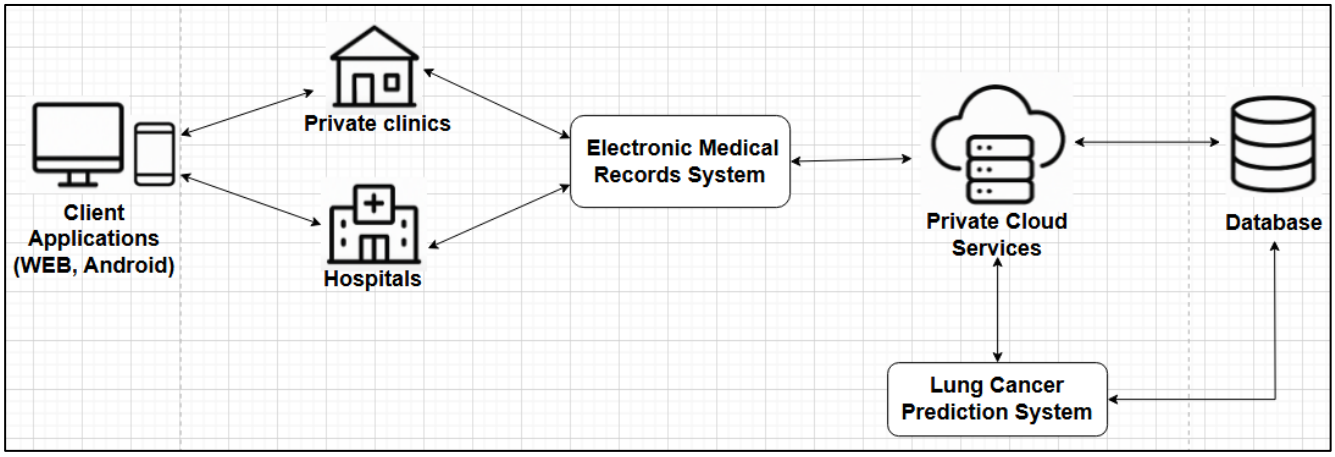

Рисунок Г.6 – Архітектура побудови інформаційної технології

	model	f1_train	f1_test	r2_train	r2_test	short
0	RandomForest Classifier	0.992877	0.974747	0.989381	0.964385	RF
1	ADABoost Classifier	0.990028	0.984848	0.985134	0.978631	ADA
2	ExtraTrees Classifier	0.924501	0.914141	0.887443	0.878908	ExT
3	DecisionTree Classifier	0.930199	0.888889	0.895938	0.843292	DT
4	XGB Classifier	0.964387	0.919192	0.902309	0.821923	XGB
5	MLP Classifier	0.990028	0.984848	0.985134	0.978631	MLP
6	LR Classifier	0.97151	0.949495	0.957526	0.928769	LR
7	SVM Classifier	0.998575	0.984848	0.997876	0.935892	SVM
8	KNN Classifier	0.995726	0.994949	0.993629	0.992877	KNN
9	LGBM Classifier	0.988604	0.984848	0.98301	0.978631	LGBM

	model	accuracy	val accuracy	loss	val loss
0	NN 2 layers	0.960938	0.963158	0.113192	0.112308

Рисунок Г.7 – Результати навчання усіх моделей

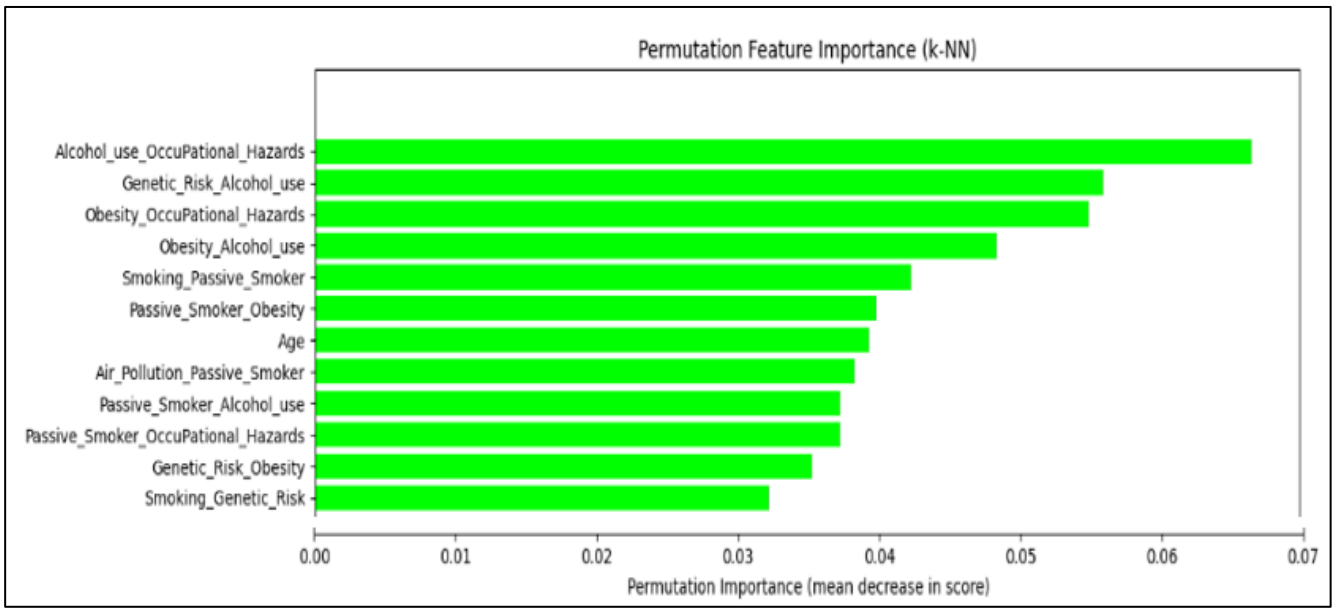

Рисунок Г.8 – Графік важливості ознак для моделі KNN за допомогою функції permutation_importance

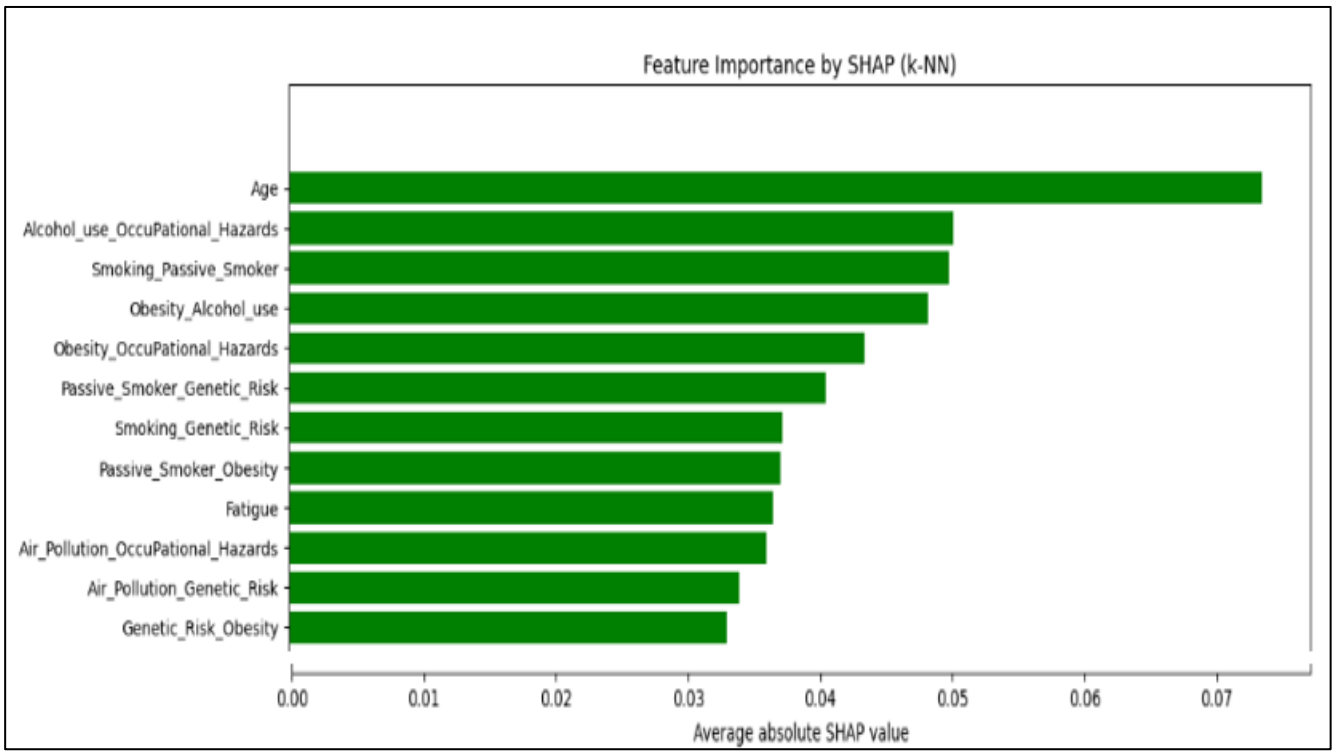

Рисунок Г.9 – Графік важливості ознак для моделі KNN за допомогою бібліотеки SHAP