КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем

Кваліфікаційна робота

на здобуття освітнього рівня магістра
за спеціальністю 121 Інженерія програмного забезпечення
на тему:

РОЗРОБКА СЕРВІСУ ДЛЯ ВИЗНАЧЕННЯ АРИТМІЇ ЗА ДОПОМОГОЮ ЕКГ

Виконав студент 2-го курсу магістратури
Семен ЧЕЛНОКОВ _____
(підпис)

Науковий керівник:
професор, доктор фіз.-мат. наук
Анатолій ПАШКО _____
(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри інтелектуальних програмних систем
«___» _____ 2023 р.,
протокол № ___
Завідувач кафедри
Олександр ПРОВОТАР _____
(підпис)

Київ – 2023

РЕФЕРАТ

Обсяг роботи 50 сторінок, 16 ілюстрацій, 11 джерело посилань, 0 додатків.

ВИЗНАЧЕННЯ АРИТМІЇ ЗА ЕКГ, НЕЙРОМЕРЕЖА, PYTHON, ANACONDA, TENSORFLOW, KERAS, SKLEAN, DATA PROCESSING, TABLEAU, TABPY SERVER.

Об’єктом дослідження є визначення аритмії серця за допомгою екг, предметною областью є застосування нейромереж для визначення виду аритмії серця.

Метою роборти є створення сервісу для визначення виду аритмії серця на основі досліджень ЕКГ. В якості засобу створення нейромережі було обрано середовище програмування Google Colab та мову програмування Python 3.9. Для створення нейромережі було обрано бібліотеке tensorflow та для створення сервісу з використанням нейромережі було використано віртуальне середовище з python 3.9 за допомогою застосунку Anaconda на якому було встановлено сервер вкиористовуючи бібліотеку TabPy.

Результат кваліфікаційної роботи: розглянуто та проаналізовано аналогічні сервіси, також були досліджені різні методи обробки даних, використовувані у медицині, та їх ефективність для вирішення задачі класифікації аритмій; проаналізовано нейромережі для визначення багатокласової класифікації та реалізовано послідовну модель з використанням прихованих шарів; запущено віртуальне середовище з сервером та додано реалізовану модель на цей сервер для подальшого користування; реалізовано дашборд з інформацією про використаний набір даних та в цьому ж дашборді реалізовано можливість визначення виду аритмії за наданими парметрами екг-дослідження;

Сфера застосування: захворювання серця, на жаль поширена проблема і в Україні, тому використання такого сервісу може бути корисним для визначення попереднього діагнозу лікарем, щоб зменшити час при визначенні кінцевого діагнозу. Також даний сервіс може бути використаний, як допоміжний інструмент при визначені виду аритмії захворюваного під час проходження електрокардіографії.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

ЕКГ – електрокардіографія

НМ – нейромережа

SB – Sinus Bradycardia

SR – Sinus Rhythm

AFIB – Atrial Fibrillation

ST – Atrial Fibrillation

AF – Atrial Flutter

SI – Sinus Irregularity

SVT – Supraventricular Tachycardia

AT – Atrial Tachycardia

AVNRT – Atrioventricular Node Reentrant Tachycardia

AVRT – Atrioventricular Reentrant Tachycardia

SAAWR – Sinus Atrium to Atrial Wandering Rhythm

ВСТУП

У сучасному світі серцево-судинні захворювання є однією з найбільш поширених та небезпечних проблем для людства. Аритмія, як одна з найбільш поширених форм серцево-судинних захворювань, вимагає оперативної діагностики та негайного лікування. У даний час наявні багато методів для діагностики аритмій, однак, використання методів машинного навчання може значно покращити якість та точність діагностики.

Актуальність роботи. У зв'язку з тим, що електрокардіографія є одним з основних методів діагностики серцевих захворювань, розробка сервісу для визначення виду аритмій за результатами електрокардіографії є вельми актуальною задачею. Застосування нейромереж для аналізу результатів електрокардіографії дозволяє з високою точністю визначити вид аритмії за досить швидкий час на основі якої лікар зможе запропонувати відповідну стратегію лікування.

Мета й завдання роботи. Метою кваліфікаційної роботи є розробка сервісу для автоматичної класифікації виду аритмій за допомогою нейромереж та застосування отриманої моделі на практиці для діагностики та лікування серцево-судинних захворювань. Для досягнення даної мети робота передбачає вирішення таких задач:

- збір та обробки даних для навчання нейромережі;
- вибір оптимальної архітектури нейромережі, настройку параметрів моделі та оцінку отриманих результатів;
- побудова серверу для користування нейромережою віддалено;
- побудова сервісу для користування кінцевим користувачем.

Отже, результат даної дипломної роботи можна використати у практичній медицині для вдосконалення методів діагностики та лікування серцево-судинних захворювань, навчання практикантів, лаборантів та студентів медичних вищів. А також для збереження здоров'я та життя пацієнтів.

РОЗДІЛ 1. АРИТМІЯ СЕРЦЯ ТА ДОСЛІДЖЕННЯ ЕКГ

1.1 Електрокардіограма

ЕКГ (електрокардіограма) – це метод дослідження, який використовується для запису електричної активності серця та встановлення його ритму. [4]

ЕКГ є одним із найпростіших та найбільш доступних методів для дослідження серця Під час ЕКГ дослідження, електроди, прикріплені до грудної клітки, рук і ніг пацієнта, реєструють електричну активність серця, яка відображається на електрокардіограмі (рис. 1.1).

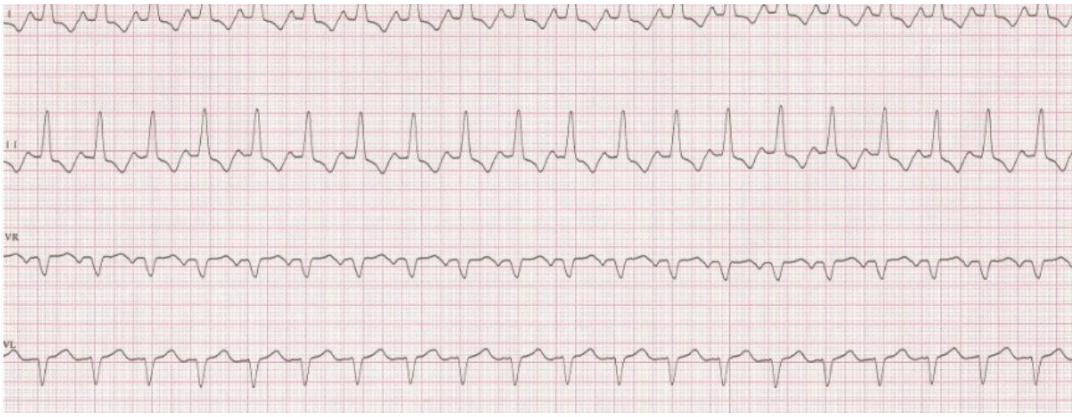

Рисунок 1.1 – Приклад електрокардіограми

ЕКГ вимірює потенціал, який генерують серцеві м'язи під час їх скорочення. Цей процес передачі електричного стимулу відбувається через спеціальні клітини серцевого м'яза – пацемакери та провідні системи. ЕКГ записує ці електричні хвилі за допомогою електродів, які прикріплені різних ділянок тіла.

Під час забору даних, проведення електрокардіографії проводиться очищення ЕКГ сигналів, яке зазвичай включає в себе фільтрацію, що включає як високочастотні, так і низькочастотні фільтри, які допомагають видалити шуми та інші артефакти, що виникають під час запису ЕКГ-сигналу.

Фільтрація низької частоти дозволяє видалити шуми, які виникають внаслідок рухів дихання та м'язів, а також шуми, що виникають під час запису ЕКГ від нестабільної контактної поверхні електродів.

Фільтрація високої частоти дозволяє видалити шуми, які виникають внаслідок рухів пацієнта та шуми від електронного обладнання.

Після очищення та фільтрації сигналів, можна приступити до аналізу ЕКГ даних.

Отримана за допомогою ЕКГ інформація дає змогу діагностувати різноманітні порушення серцево-судинної системи, такі як порушення ритму серця, патології клапанів серця, ішемічну хворобу серця, інфаркт міокарда та інші.

ЕКГ є невід'ємною частиною медичних досліджень та популярним методом рутинних профілактичних обстежень серцево-судинної системи.

1.2 Аритмія серця

Аритмії серця є досить поширеною проблемою, яка може виникати в будь-якому віці. Аритмії – це зміна ритму серця, коли воно б'ється занадто швидко, повільно або нерегулярно. Зазвичай, серце б'ється регулярно з частотою від 60 до 100 ударів на хвилину у дорослих людей. Аритмії можуть призвести до серйозних ускладнень, таких як інсульт, серцеві недостатності та інші захворювання серця.

Існує багато видів аритмій, які можуть мати різні причини. Найбільш поширені:

– фібриляція передсердь (нерегулярний і швидкий серцевий ритм), тахікардія (швидкий серцевий ритм більше 100 ударів за хвилину);
– брадікардія (повільний серцевий ритм менше 60 ударів за хвилину);
– блокади серцевого провідника (зупинка передачі електричного сигналу у серці);
– додаткові серцеві скорочення (екстрасистоли).

Кожен тип аритмії має свої характеристики та може бути пов'язаний з різними факторами ризику, такими як вік, стать, наявність захворювань серця, гіпертонія, діабет, зловживання алкоголем та інші.

Для діагностики та лікування аритмій використовуються різні методи, включаючи ЕКГ (електрокардіограму), Холтерів моніторинг, ехокардіографію та інші дослідження. Розробка сервісу, який може автоматично визначати тип аритмій за результатами ЕКГ, може значно полегшити та прискорити діагностику та лікування цієї проблеми.

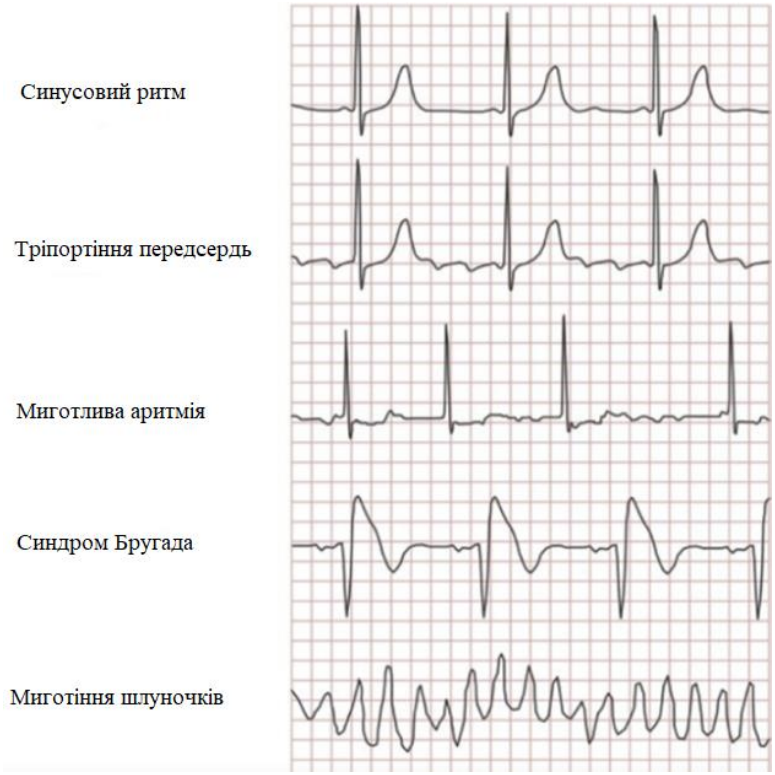

Рисунок 1.2 – Електрокардіограми пацієнтів з різними видами аритмій

Інші можливі причини аритмій включають серцеву недугу – порушення електролітного балансу, коронарну хворобу, інфекцію, травму або отруєння. Деякі аритмії можуть бути спадковими. Лікування аритмій зазвичай залежить від відповідної форми аритмії та її причини. Як правило, лікарі використовують ліки, щоб зменшити частоту аритмії або змінити серцевий ритм, або інші методи, такі як кардіовертер-дефібрилятор, щоб воротити серцевий ритм до нормального. В найскладніших випадках, може бути рекомендована хірургічна процедура для поліпшення серцевого ритму або встановлення кардіостимулятора.

1.3 Сучасні системи визначення аритмій

Сучасні сервіси для визначення аритмій за результатами ЕКГ – це багатофункціональні програми, які забезпечують високоточний аналіз ЕКГ і визначення різних типів аритмій.

На сьогоднішній день існує багато сучасних сервісів для визначення типу аритмії за даними ЕКГ. [5] Вони використовують різні методи машинного навчання і не тільки, щоб аналізувати дані ЕКГ та класифікувати їх на різні типи аритмій.

Один з таких сервісів – Cardiologs. Він використовує навчання з вчителем для класифікації різних типів аритмій на основі даних ЕКГ. Cardiologs базується на архітектурі нейронних мереж, що використовують згорткові шари (convolutional layers) та шари рекурентних нейронів (recurrent layers).

Ще один сервіс – AliveCor KardiaMobile 6L – використовує збір і аналіз даних ЕКГ з використанням мобільного пристрою. Користувачі можуть записувати своє ЕКГ на пристрої з електродами і передавати ці дані до облікового запису AliveCor, де вони аналізуються з використанням алгоритмів машинного навчання для виявлення різних типів аритмій.

Також варто відзначити інші сервіси, такі як Zio XT Patch, який використовує бездротовий моніторинг ЕКГ протягом декількох днів, та BardyDx CAM Patch, який дозволяє моніторити ЕКГ протягом навіть кількох тижнів.

Всі ці сервіси використовують алгоритми машинного навчання для аналізу даних ЕКГ та виявлення різних типів аритмій. Вони мають великий потенціал для ранньої діагностики та виявлення серцевих захворювань, також

можуть бути корисні для здорових людей, які бажають знати більше про свій стан здоров'я. Проте для того, щоб скористатись цими сервісами необхідно купляти додаткове обладнання та підписки додатків.

РОЗДІЛ 2. ОПИС ТА АРХІТЕКТУРА ПРОЕКТУ

2.1 Tableau

Tableau – це програмний продукт, який дозволяє візуалізувати та аналізувати дані за допомогою інтерактивних та динамічних графіків, дашбордів, звітів та карт.[2] Його можна використовувати для збору, зберігання та обробки даних, а також для розповсюдження звітності в організації. Tableau підтримує багато джерел даних, таких як Excel, SQL-бази даних, Hadoop, Amazon Web Services та інші.

Один з ключових елементів Tableau – це інтерфейс для візуалізації даних. Він дозволяє користувачам легко перетворювати дані в графіки, діаграми, таблиці та картографічні зображення. Це забезпечує швидкий та ефективний процес розуміння та аналізу даних.

Tableau також дозволяє користувачам співпрацювати та обмінюватися даними з іншими користувачами, що забезпечує швидкий та ефективний процес обміну даними та співпраці.

Крім візуалізації даних, Tableau надає також можливості аналізу та прогнозування даних. Завдяки інтеграції зі статистичними пакетами та іншими інструментами машинного навчання, Tableau дозволяє використовувати аналітичні методи для отримання більш детальної інформації з даних.

Tableau використовують у багатьох сферах, таких як бізнес-аналітика, фінанси, медицина, освіта та багато інших. Він дозволяє компаніям та організаціям отримувати більш глибоке розуміння даних та приймати кращі рішення. В свою чергу, інтеграція Tableau з машинним навчанням дозволяє

створювати прогнози, моделі та класифікатори на основі даних, що робить його потужним інструментом для розробки сервісів, які працюють з даними.

2.2 TabPy сервер

TabPy сервер може бути використаний для створення та впровадження моделей машинного навчання безпосередньо в Tableau. Це дозволяє користувачам Tableau використовувати власні алгоритми та моделі для аналізу даних, що робить Tableau ще більш потужним інструментом для аналізу даних.

TabPy сервер дозволяє використовувати різні мови програмування для створення моделей машинного навчання, але переважно використовується Python. Python має багатий стек бібліотек машинного навчання, таких як NumPy, SciPy та TensorFlow, які дозволяють створювати складні моделі машинного навчання. [1] Крім того, Python має простий та доступний синтаксис, що робить його досить зрозумілим для більшості розробників.

TabPy сервер використовує відкритий протокол REST API для забезпечення взаємодії між Tableau та сервером. Це означає, що моделі машинного навчання можуть бути використані безпосередньо в Tableau, і їх можна використовувати для створення нових полів, розрахунків та прогнозів.

Загалом, TabPy сервер – це потужний інструмент для використання моделей машинного навчання в Tableau. Він дозволяє створювати та використовувати власні моделі машинного навчання безпосередньо в Tableau, що дозволяє користувачам Tableau аналізувати дані ще більш ефективно та точно.

2.3 Взаємодія Tableau та TabPy-сервера

TabPy сервер та Tableau взаємодіють між собою за допомогою REST API. REST API – це протокол обміну даними, який використовує HTTP-запити для взаємодії між різними компонентами системи. В даному випадку, Tableau та TabPy використовують REST API для передачі даних одне одному.

Коли користувач створює запит на створення нового обчислення у Tableau, Tableau відправляє запит на TabPy за допомогою REST API, передаючи необхідні дані, такі як параметри запиту та дані для обробки. TabPy отримує запит, аналізує його та оброблює дані, використовуючи Python-скрипт. Після цього, результат обчислення повертається до Tableau за допомогою REST API, що дозволяє відображати результати візуалізації у Tableau.

У випадку, коли користувач створює запит на отримання даних з TabPy в Tableau, Tableau відправляє запит до TabPy за допомогою REST API, передаючи необхідні параметри запиту. TabPy отримує запит, оброблює дані та повертає результат у форматі, який може бути візуалізований у Tableau. Після цього, дані передаються з TabPy до Tableau за допомогою REST API, і візуалізація створюється у Tableau на основі отриманих даних.

Таким чином, використання REST API дозволяє Tableau та TabPy взаємодіяти між собою та обмінюватися даними у реальному часі. API забезпечує гнучкість та розширюваність взаємодії між системами, що дозволяє використовувати Tableau та TabPy для вирішення різноманітних завдань у різних галузях.

2.4 Опис архітектури проекту

Для розробки сервісу з визначення типу аритмії за реузльтатами ЕКГ було використано мову Python, а саме – для підготовки нейромережі, яка буде визначати тип аритмії.

Також для розробки сервісу було використано програмне забезпечення Tableau, яка дозволяє працювати з великими об’ємами даних та проводити їх аналіз. Ця система надає можливість використовувати підготовленні моделі машиного навчання та довзволяє розміщувати свої дашборди в клауді (Tableau Public, Tableau Cloud та Tableau Server), що збільшує автономність системи та спрощує роботи для кінцевого користувача.

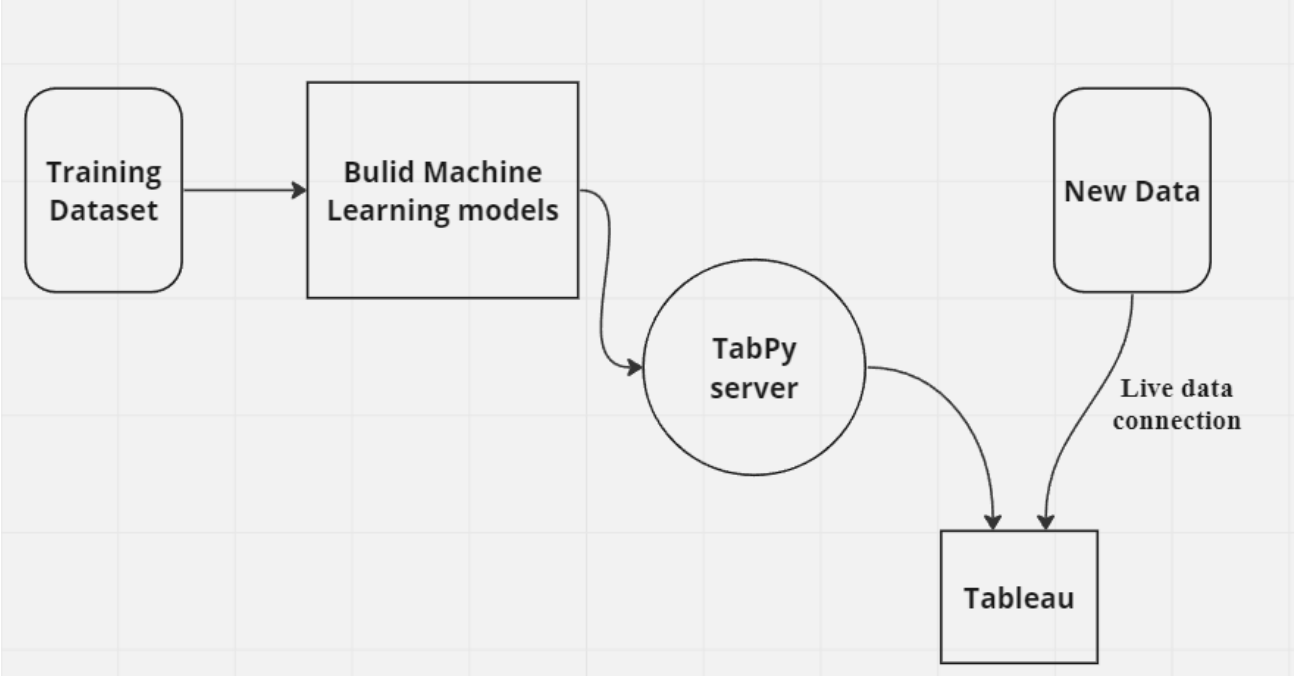

Рисунок 2.1 – Архітектура проекту

На рисунку 2.1 видно архітектуру даного проекту, спочатку було побудовано модель машинного навчання та навченно на наданому наборі даних в середовищі Python, використовуючи бібліотеку TensorFlow. Після навчання модель було збережено та експортовано та згодом за допомогою tabpy_tools.client було завантажнно на сервер TabPy.

За допомогою Anaconda, створено віртуальне середовище на якому запускається TabPy сервер, який працює за моделью клієнт-сервер – це модель в якій взаємодія між клієнтом та сервером відбувається за допомогою HTTP-запитів, які відповідають принципам архітектури REST.

Клієнт – це модуль, який взаємодіє з сервером, зазвичай за допомогою HTTP-запитів. У випадку використання TabPy для моделей машинного навчання, клієнтом може бути, наприклад, Tableau Desktop, Tableau Cloud або Tableau Server.

Сервер – це модуль, який приймає запити та надає доступ до даних за допомогою HTTP-запитів. В контексті використання TabPy для моделей машинного навчання, сервером є сам TabPy. Він прослуховує запити на певному порту і відопідає на запити, які йому надсилає клієнт. Наприклад, при створені поля в tableau desktop з використанням python – сервер отримає запит на використання тої чи іншої функції.

Згодом було створено дашборд в Tableau Desktop використовуючи поля(calculated fields), що зв'язані з моделлю на сервері TabPy. Поля передають параметри, необхідні для визначення типу аритмії – вік та стать пацієнта, інтервали P-Q, комплекс QRS та інші дані, зібрані під час ЕКГ. Після отримання відповіді з сервера, результат передається в це ж саме поле та відображається на робочій області.

Така архітектура дозволяє швидко та ефективно інтегрувати моделі машинного навчання в дашборди Tableau, що забезпечує зручний та інтерактивний спосіб аналізу даних та прийняття рішень.

РОЗДІЛ 3. АНАЛІЗ ТА ПОБУДОВА НЕЙРОМЕРЕЖІ

3.1 Аналіз нейромереж

Один з перших успішних прикладів використання нейронних мереж був створений Френком Розенблаттом у 1958 році і був названий перцептроном. Перцептрон був здатний класифікувати прості об'єкти, такі як лінії та круги, і його можна було навчити розрізняти між двома категоріями, використовуючи метод навчання з вчителем.

Основою роботи нейронних мереж є математичні операції з ваговими коефіцієнтами, які навчаються в процесі тренування на великій кількості даних. Тренування нейронної мережі полягає у пошуку найкращих значень вагових коефіцієнтів, які дозволяють моделі робити точні прогнози на тестових даних. Цей процес зазвичай використовує метод зворотного поширення помилок, де модель намагається мінімізувати функцію втрат, що вимірює різницю між прогнозованими значеннями та правильними відповідями. [8]

Після тренування, нейронна мережа може бути використана для прогнозування нових даних.

Нейронні мережі — це алгоритму машинного навчання, натхненний структурою та функціями людського мозку. Вони складаються із взаємопов’язаних вузлів або штучних нейронів, які обробляють і передають інформацію через серію шарів.

У нейронній мережі вхідні дані подаються на вхідний рівень, який пропускає дані через один або кілька прихованих шарів нейронів, які виконують обчислення над вхідними даними. Потім вихідний рівень створює

прогноз або класифікацію на основі обчислень, виконаних прихованими шарами

Нейронні мережі використовувалися в різноманітних програмах, включаючи розпізнавання зображень і мови, обробку природної мови, комп'ютерний зір і навіть такі ігри, як го та шахи. Вони також використовуються в таких галузях, як охорона здоров'я, фінанси та маркетинг для таких завдань, як виявлення шахрайства, оцінка ризиків і сегментація клієнтів.

Нейронні мережі можна використовувати для класифікаційних завдань, навчаючи їх на позначеному наборі даних, де кожен приклад пов'язаний з міткою або категорією. Нейронна мережа вчиться пов'язувати шаблони у вхідних даних з відповідними мітками, а потім може використовувати ці знання для класифікації нових, небачених прикладів.

Процес навчання нейронної мережі для класифікації включає кілька етапів:

1) підготовка даних (data preparation):
    a) набір даних розділяють на навчальний набір, набір перевірки та тестовий набір. Навчальний набір використовується для навчання мережі, тоді як набір перевірки використовується для моніторингу її продуктивності під час навчання та коригування її параметрів. Тестовий набір використовується для оцінки кінцевої продуктивності мережі на невидимих даних;
2) проектування архітектури мережі:
    a) етап заключається у створенні структури нейронної мережі, включаючи кількість шарів, кількість нейронів на кожному

шарі та функції активації, які використовуються на кожному шарі;

3) навчання:

а) мережа навчається на навчальному наборі за допомогою алгоритму оптимізації, такого як стохастичний градієнтний спуск. Під час навчання мережа коригує свої ваги та зміщення, щоб мінімізувати різницю між прогнозованими результатами та справжніми мітками;

4) перевірка:

а) продуктивність мережі відстежується на наборі перевірки, щоб запобігти перенавчанню, яке відбувається, коли мережа добре працює на навчальному наборі, але погано на нових даних;

5) тестування:

а) кінцева продуктивність мережі оцінюється на тестовому наборі для оцінки її здатності до узагальнення.

Коли нейронна мережа навчена, її можна використовувати для класифікації нових, небачених прикладів. Вхідні дані подаються в мережу, яка створює розподіл ймовірностей за можливими класами. Потім клас із найвищою ймовірністю вибирається, як прогнозований результат або мережа повертає вектор ймовірностій приналежності об’єкту до кожного з класів.

Загалом, нейронні мережі можуть бути потужним інструментом для класифікаційних завдань, особливо при роботі зі складними або великовимірними даними. Завдяки належній підготовці та налагодженню вони можуть досягти найсучаснішої продуктивності в різноманітних класифікаційних завданнях.

3.2 Задача класифікації

Задача класифікація полягає у визначенні класу певного об’єкту на основі його вхідних параметрів або аттрибутів. Приклади вхідних даних зазвичай представлені у вигляді векторів ознак, які можуть бути числовими, категоріальними або комбінацією (числовими і категоріальними).

У завданні класифікації надається набір даних, який складається з позначених прикладів, де кожен приклад пов’язаний із відомим класом або категорією. Мета класифікатора — навчитися зіставляти вхідні функції та відповідні мітки класу, щоб він міг точно класифікувати нові, невідомі приклади.

Наприклад, завдання класифікації може включати визначення того, чи є електронний лист спамом чи ні, на основі його вмісту та інших характеристик, або визначення виду грипу у пацієнта на основі його симптомів та інших факторів.

Існує кілька типів завдань класифікації, включаючи бінарну класифікацію (де результатом є один із двох можливих класів), багатокласову класифікацію (де результатом є один із кількох можливих класів) і класифікацію з кількома мітками (де прикладом може бути пов’язані з кількома мітками/класами). Завдання класифікації можна вирішувати за допомогою різноманітних алгоритмів машинного навчання, включаючи дерева рішень, опорні векторні м ашини та нейронні мережі.

3.3 Проектування нейромережі для багато-класової класифікації

При проектуванні нейронної мережі для багатокласової класифікації існує кілька рекомендацій, які слід розглянути для досягнення оптимальної продуктивності:

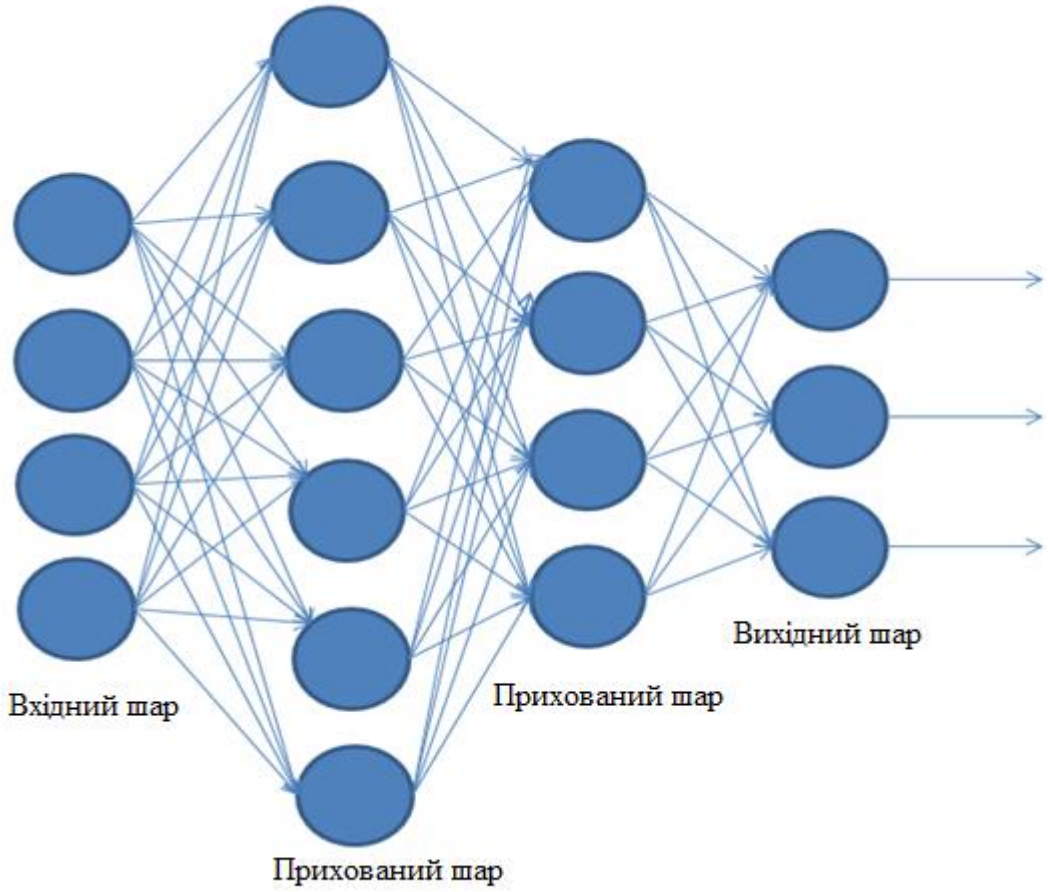

Рисунок 3.1 – Приклад нейромережі для багатокласової класифікації

На рисунку 3.1 якраз можна бачити приклад багатокласової класифікації, де на вході мережа приймає 4 вхідні значення на основі яких має порахувати ймовірності належності об’кту до кожного з трьох класів.

Функція активації вихідного рівня: функція активації вихідного рівня залежить від кількості класів у наборі даних. Для бінарної класифікації зазвичай використовується sigmoid функція активації. Для багатокласової класифікації виокристовується функція активіації softmax для перетворення вихідних даних кожного нейрона в розподіл ймовірностей за класами. [7]

Функція втрат: вибір функції втрат залежить від типу задачі багатокласової класифікації. Наприклад, cross-entropy loss зазвичай використовується для задач багатокласової класифікації, де кожен приклад належить лише до одного класу. Для задач класифікації з кількома мітками, де приклад може належати до кількох класів, можна binary cross-entropy loss або hinge loss.

Архітектура мережі: архітектуру нейронної мережі слід вибирати на основі складності вхідних даних і розміру набору даних. Загальною архітектурою для багатокласової класифікації є нейронна мережа прямого зв’язку з кількома прихованими шарами. Однак інші архітектури, такі як згорткові нейронні мережі (convolution neural networks) або рекурентні нейронні мережі (recurrent neural networks), можуть використовуватися залежно від характеру вхідних даних.

Регуляризація: методи регулярізації, такі як відсівання або регулярізація рівня L2, можуть бути використані для запобігання перенавчанню нейронної мережі до навчальних даних.

Алгоритм оптимізації: вибір алгоритму оптимізації залежить від розміру набору даних і складності мережі. Загальні алгоритми оптимізації для багатокласової класифікації включають стохастичний градієнтний спуск, Адам і RMSprop.

Загалом, розробка ефективної нейронної мережі для багатокласової класифікації передбачає збалансування складності моделі та продуктивності

узагальнення, а також експериментування з різними варіантами дизайну та гіперпараметрами для досягнення найкращих результатів. Використання нейромережі для даної задачі є доволі гарним рішенням, адже :

- нейромережі здатні для вирішення складних задач, де необхідно обробити велику кількість класів та значний обсяг даних;
- вони можуть бути налаштовані під різні типи даних, що розширює їх спектр застосування;
- можливість використання глибокої структури з великою кількістю параметрів.

Проте, вони мають і недоліки, а саме:

- нейромережі можуть бути вразливі до перенавчання, коли вони навчаються на основі вхідних даних поточного даних та не можуть ефективно працювати на нових даних;
- вони можуть бути важкими для інтерпретації, оскільки їхні внутрішні процеси часто є нелінійними та складними;
- значна вартість та час навчання, особливо якщо це глибокі архітектурні мережі та великі обсяги даних;
- дуже залежать від якості та повноти даних, тому неточні або неповні/нерівномірні(класи по об’єму даних не рівні між собою) знчно зменшують ефективність нейромереж.

РОЗДІЛ 4. ПІДГОТОВКА ДАНИХ ДЛЯ НЕЙРОМЕРЕЖІ

4.1 Перевірка даних

Дані отримані з відкритого ресурсу physionet.org, в якому були сформовані файли сигналів для кожного з пацієнтів, словник до видів аритмії та уже оброблений файл з очищеними та відфілтрованими даними по сигналам, для кожного з пацієнтів. [6] [10] Тому, для реалізації НМ використав файл з уже обробленими сигналами.

Дані мають наступний вигляд:

1) Інформація про пацієнта:
    a) FileName – назва файлу, пов’язана з файлам сигналу, на основі якого зроблено цей запис;
    b) Rhythm – вид аритмії наявний у пацієнта;
    c) Beat – характеристика серцебиття;
    d) PatientAge – вік пацєнта;
    e) DateofBirth – дата народження;
    f) Gender – стать;
2) Параметри електро-кардіаграми:
    a) VentricularRate – шлуночкова частота;
    b) AtrialRate – частота предсердь;
    c) QRSDuration – тривалість QRS;
    d) QTInterval – інтервал QT;
    e) QTCorrected – QT виправлення;
    f) RAxis – R вісь;
    g) Taxis – T вісь;
    h) QRSCount – кількість QRS;

i) QOnset – Q початок;
j) QOffset – Q зсув;
k) TOffset – T зсув.

За наявними даними переглянув розподіл пацієнтів на типи аритмії, щоб зрозмуміти об’єми даних для кожного з класу, для яких робитиму класифікацію.

Розподіл пацієнтів за видами аритмії такий:

– SB – 3889;
– SR – 1826;
– AFIB – 1780;
– ST – 1568;
– SVT – 587;
– AF – 445;
– SA – 399;
– AT – 121;
– AVNRT – 16;
– AVRT – 8;
– SAAWR – 7.

Для розроблення нейромержі залишив аритмії типу SB, SR, AFIB та ST, так як вони мають достатній об’єм даних та приблизно однаково розподілені (згодом змешнив кількість записів синусової брадикардії, щоб уникнути перенавчання на одному класі).

Загальний розподіл даних за інформацією про пацієнта подано у рисунку 4.1.

Розподіл пацієнтів

Gender	Rhythm	0-10 years	10-20 years	20-30 years	30-40 years	40-50 years	50-60 years	60-70 years	70-80 years	80+ years
FEMALE	AF			2	7	11	15	52	55	46
	AFIB			2	3	19	59	168	255	233
	AT	2	2	3	2		8	12	20	8
	AVNRT			2	1		6	2		1
	AVRT							2	1	
	SA	28	30	40	18	15	16	15	10	4
	SAAWR		2	1					1	2
	SB		9	37	88	252	452	361	143	66
	SR	2	13	90	102	189	262	220	101	45
	ST	29	30	76	75	95	146	145	102	71
	SVT	1	9	29	35	62	53	63	25	31
MALE	AF			1	3	14	36	56	81	66
	AFIB				5	36	104	265	334	297
	AT	2		2	1	4	5	16	19	15
	AVNRT							2		2
	AVRT			1		2		1	1	
	SA	42	43	24	28	26	16	17	23	4
	SAAWR					1				
	SB	2	29	98	141	346	589	798	362	116
	SR	2	19	51	92	137	163	183	109	46
	ST	25	33	62	64	107	126	168	117	97
	SVT	1	4	20	25	52	54	50	40	33

Рисунок 4.1 – розподіл пацєнтів з вхідного набору даних

4.2 Обробка та кодування даних

Наступним кроком, вибрав параметри, які будуть використовуватись нейромережею для навчання – це будуть всі параметри, наявні у файлі, окрім – DateofBirth та Beat, адже замість дати народження ми вже маємо вік пацієнта, а вид серце-биття можна отримати з наявних інших параметрів.

Щоб підготувати дані для використання НМ потрібно закодувати параметри, які збережені у форматі тексту – це Rhytm та Gender, використовуючи бібліотеку sklearn іморптував метод LabelEncoder() – який довзоляє перетворювати категоріальні дані у числові мітки або індекси. Тобто, він дозволяє легко перетворити рядки з текстовими значеннями на цілі числа. LabelEncoder діє на основі методу "one-hot encoding", який замінює кожне унікальне значення категорії на свій індекс або мітку.

Після застосувань кодування маємо вид даних поданих на рисунку 4.2.

Arytmia Dataset

index	Rhythm	AtrialRate	Gender	PatientAge	QOffset	QOnset	QRSCount	QRSDuration	QTCorrected	QTInterval	RAxis	Rhythm	TAxis	TOffset	Ventricular..
0	0	234	1	85	265	208	19	114	496	356	81	0	-27	386	117
1	1	52	0	59	261	215	8	92	401	432	76	1	42	431	52
3	1	53	1	66	267	219	9	96	427	456	34	1	3	447	53
5	1	57	0	46	260	225	9	70	393	404	38	1	24	427	57
6	0	86	0	80	252	215	17	74	459	360	69	0	83	395	98
7	2	63	1	46	266	221	11	90	384	376	24	2	38	409	63
8	1	59	1	45	260	218	10	84	386	390	78	1	68	413	59
9	1	58	0	47	252	212	10	80	412	420	80	1	48	422	58
10	1	55	1	63	263	223	9	80	417	436	74	1	74	441	55
11	3	120	1	77	260	225	20	70	432	306	39	3	28	378	120
12	1	58	1	54	262	211	10	102	424	432	2	1	60	427	58

Рисунок 4.2 – Готовий датасет для навчання нейромережі

РОЗДІЛ 5. РОЗРОБКА НЕЙРОМЕРЕЖІ

5.1 TensorFlow

Для розробки нейромережі було використанно бібліотеку tensorflow, оскільки – це відкрита платформа з відкритим вихідним кодом для розробки та навчання моделей машинного навчання та глибинного навчання. TensorFlow пропонує ряд інструментів та бібліотек для розробки та навчання різноманітних типів моделей машинного навчання. Основою TensorFlow є структура графа обчислень, де кожен вузол графа представляє математичну операцію, а ребра – дані, що передаються між операціями. Завдяки цій структурі, TensorFlow може розподіляти обчислення між багатьма процесорами та пристроями, що дозволяє ефективно виконувати обчислення над великими обсягами даних.

Для створення моделі в TensorFlow було використона високо-рівневий інтерфейс – Keras. Крім того, TensorFlow має широкий спектр інструментів для обробки даних, включаючи вбудовані функції для завантаження та передобробки даних, валідації моделей та оцінки їх продуктивності. TensorFlow пропонує можливості для розробки та навчання моделей на різних пристроях, таких як CPU, GPU та TPU.

5.2 Створення нейромережі

Для створення мережі розділив отриманий набір даних на вхідні параметри – вектор x (набір даних дослідження-екг), та вихідний параметр – значення y (тип аритмії поле Rhytm).

На основі моделі послідовної мережі (Sequential) формую нейромережу з чотирьох шарів Dense – повнозв’язні шар, який має зв’язок між всіма нейронами двох прилеглих шарів. В слоях використовую функцію активіації relu – це функція активації є нелінійною функцією, яка повертає 0 для вхідних значень, менших за 0, і повертає саме це значення для вхідних значень, більших або рівних 0 – формула виглядає так: f(x) = max(0,x). Ця функція має кілька переваг при використанні в нейронних мережах. По-перше, вона є простою і швидко виконується. По-друге, вона дозволяє нейронній мережі "вимкнути" деякі нейрони, що допомагає уникнути перенавчання.

В останньому шарі додав 4 нейрони, з функцією активіації softmax, яка забезпечує ймовірносну класифікацію на кілька класів.

Після створення шарів, компілюю модель з функцією втрат sparse_categorical_crossentropy, яка підходить для задач класифікації з багатьма класами та оптимізатором Adam.

Модель має наступну архітектуру:

– вхідний шар (input layer) з 13 нейронів, що відповідають за кількість ознак у вхідних даних;
– перший прихований шар (hidden layer) з 240 нейронів, що використовує функцію активації relu;
– другий прихований шар (hidden layer) з 36 нейронів, що використовує функцію активації relu;
– третій прихований шар (hidden layer) з 12 нейронів, що використовує функцію активації relu;
– вихідний шар (output layer) з 4 нейронів, що використовує функцію активації softmax, яка перетворює вихідні значення на імовірності належності до кожного з 4 класів.

5.3 Розподілення набору даних

Для навчання підготував та розподілив набори даних у такій пропорції: 80% для тренування, 20% для тестування. В процесі тренування також поділив тренувальний набір даних на набір для тренування та валідації, це допомже уникнути проблеми перенавчання (‘overfitting’).

5.4 Проблема перенавчання та недовчання нейромережі

Перенавчання (overfitting) – це проблема, коли модель машинного навчання занадто точно "запам'ятовує" тренувальні дані, в результаті чого вона недостатньо узагальнює свої знання і не може ефективно працювати з новими, раніше не баченими даними.

Це може статися, коли модель має надто велику кількість параметрів та використовується недостатня кількість тренувальних даних, або коли тренування триває дуже довго, що може призвести до того, що модель перестане виявляти нові закономірності в даних і буде тільки запам'ятовувати вже відомі.

Недонавчання (underfitting) – це коли модель недостатньо "запам'ятовує" тренувальні дані і не може ефективно працювати ні з тренувальними, ні з тестовими даними.

Щоб уникнути цих проблем, можна використовувати різноманітні методи, такі як регуляризація, додавання шару – dropout (що випадковим чином прибирає певні нейрони), зменшення кількості параметрів, збільшення кількості тренувальних даних, зменшення часу тренування та інші. [11] Також можна використовувати перевірку з раннім зупиненням (early

stopping), яка зупиняє тренування моделі, якщо вона перестає показувати покращення на тестових даних.

5.4 Алгоритм зворотнього поширення помилки

Алгоритм зворотнього поширення помилки (backpropagation) є основним методом навчання нейронних. Цей алгоритм дозволяє знаходити такі ваги нейронів мережі, які мінімізують функцію втрат (loss function).

Принцип роботи данного алгоритму полягає у зворотньому поширенні помилки через мережу, після кожного ітерації роботи (визначення класу, прогнозування значення, тощо). Якщо прогноз мережі не відповідає правильній відповіді, то похибка обчислюється та розповсюджується назад через всю мережу по кожному з сигналів.

На початку навчання, ваги нейронів мережі ініціалізуються випадковими значеннями, і модель здійснює передачу вперед через мережу, обчислює прогноз та порівнює його з правильною відповіддю. На основі різниці між прогнозованим значення та правильною відповіддю, обчислюється значення втрат.

Згодом, значення втрат використовуються для визначення градієнту (часткової похідної функції втрат відносно кожної ваги) у кожному нейроні мережі. Цей градієнт потім використовується для оновлення ваг кожного нейрона зворотнім шляхом через мережу, що дозволяє зменшити втрати.

Процес навчання повторюється багато разів, і з кожним циклом навчання (епохою навачання) ваги оновлюються таким чином, щоб зменшити втрати та підвищити точність прогнозів. Зазвичай, процес навчання

закінчується, коли досягається задана точність прогнозування або коли підвищення точності перестає бути помітним.

5.4 Тренування нейромережі

Для навчання, виділив валідаційний набір даних – 25% від тренувалульного наборду даних. Та виставив параметр “epochs” рівним 100 – що пройде сто епох тренування мережі, і кожен раз коли мережа натренується їй буде видано валідаційний набір даних і вона порівнюватиме результати своїх обчислень з фактичними і за допомогою алгоритму зворотнього поширення помилки (backpropagation) будуть оновлені ваги мережі для покращення точності визначення класу об’єкта.

Після навчання отримав показник точності (графік зображений на рисунку 5.1):

– на валідаційному наборі = 93.8%;
– на тестувальному наборі = 93.3%.

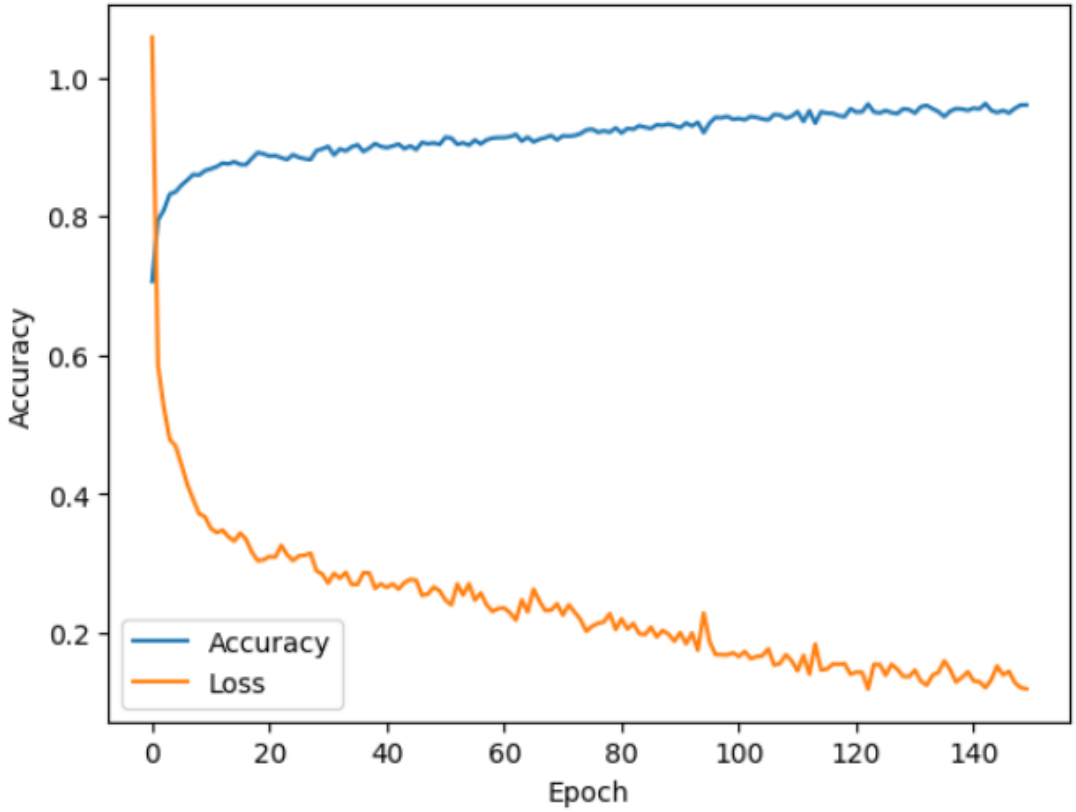

Рисунок 5.1 – Результати навчання нейромережі

РОЗДІЛ 6. РОЗГОРТАННЯ НЕЙРОМЕРЕЖІ НА СЕРВЕРІ TabPy

TabPy (Tableau Python Server) – це сервер, який дозволяє виконувати Python скрипти в Tableau. Завдяки TabPy можна розширити можливості Tableau, додавши до візуалізацій додаткові аналітичні функції, які реалізовані на Python.

TabPy можна запустити як сервер, що виконується на комп'ютері або на сервері. Після запуску TabPy стає доступним для використання в Tableau. TabPy також дозволяє створювати власні моделі машинного навчання на Python та використовувати їх в Tableau для прогнозування та класифікації даних.

6.1 Розгортання TabPy сервера

Для запуску цього сервера, спочатку необхідно встановити python 3.9, та завантажити пакет tabpy, як бібліотеку. Щоб запустити сервер необхідно просто виконати команду tabpy у терміналі і сервер буде запущено на цій машині, для роботи сервер використовує 9004 порт. Після успішного запуску, можна перевірити роботу сервера – треба перейти в браузер та увести адресу – http://localhost:9004/.

Рисунок 6.1 – Приклад успішного запуску tabpy сервера

Для роботи з сервером було створено віртуальне середовище за допомогою Anaconda, що дозволить розробити та налаштувати роботу сервера віртуально, а потім перенести образ системи на іншу машину і бути певним, що все буде працювати без додаткових налаштувань.

6.2 Завантаження нейромережі на сервер

Для завантаження нейромережі на сервер необхідно написати код, який має відкрити готову мережу, та написати функцію, яка буде приймати на вхід параметри, що необхідні для роботи моделі, та повертатиме результат роботи моделі, тобто, прогнозований клас, якому належить об’єкт, приклад зображено на рисунку 6.2.

```
def arytmia_classification(_arg1, _arg2, _arg3, _arg4, _arg5, _arg6, _arg7, _arg8, _arg9, _arg10, arg11, _arg12, _arg13):
  input_data = np.column_stack([_arg1, _arg2, _arg3, _arg4, _arg5, _arg6, _arg7, _arg8, _arg9, _arg10, arg11, _arg12, _arg13])
  x = pd.DataFrame(input_data, columns=['PatientAge',
                                        'Gender',
                                        'VentricularRate',
                                        'AtrialRate',
                                        'QRSDuration',
                                        'QTInterval',
                                        'QTCorrected',
                                        'RAxis',
                                        'TAxis',
                                        'QRSCount',
                                        'QOnset',
                                        'QOffset',
                                        'TOffset'])
  y = sq_model.predict(x)
  prob = np.amax(y)
  result = np.argmax(y)
  rhythm = ''
  if result == 0:
    rhythm = 'AFIB'
  elif result == 1:
    rhythm ='SB'
  elif result == 2:
    rhythm ='SR'
  else: rhythm ='ST'

  return rhythm
```

Рисунок 6.2 – Код функції для сервера TabPy

Яку згодом необхідно залити на сервер використовуючи метод deploy() наявний у бібліотеці tabpy.tabpy_tools. Після успішного виконання, вона буде відображена на сервері (рис. 6.3)

TabPy Server Info:


```
{
    "description": "",
    "creation_time": "0",
    "state_path": "C:\\Users\\sam\\anaconda3\\envs\\my-tabpy-env\\Lib\\site-packages\\tabpy\\tabpy_server",
    "server_version": "2.7.0",
    "name": "TabPy Server",
    "versions": {
        "v1": {
            "features": {
                "evaluate_enabled": true,
                "gzip_enabled": true,
                "arrow_enabled": false
            }
        }
    }
}
```


Deployed Models:


```
{
    "summm": {
        "description": "descript",
        "type": "model",
        "version": 1,
        "dependencies": [],
        "target": null,
        "creation_time": 1683572243,
        "last_modified_time": 1683572243,
        "schema": null,
        "docstring": "-- no docstring found in query function --"
    },
    "arytmia_classification": {
        "description": "Returns type of arytmia.",
        "type": "model",
        "version": 6,
        "dependencies": [],
        "target": null,
        "creation_time": 1683572307,
        "last_modified_time": 1683574550,
        "schema": null,
        "docstring": "-- no docstring found in query function --"
    },
    "arytmia_classification_v2": {
        "description": "Returns type of arytmia. Version with CategoricalAccuracy",
        "type": "model",
        "version": 1,
```


Рисунок 33 – Успішне завантаження моделі arytmia_classification

РОЗДІЛ 7. РОЗРОБКА ДАШБОРДУ

Для використання побудови сервісу, створив дашборд на основі якого буде працювати сервіс для визначення виду аритмії за вказаними параметрами дослідження-екг.

Для використання побудованої нейромережі необхідно підключитись до раніше створеного серверу TabPy, на якому було запущено нейромережу (рис. 7.1 та рис. 7.2).

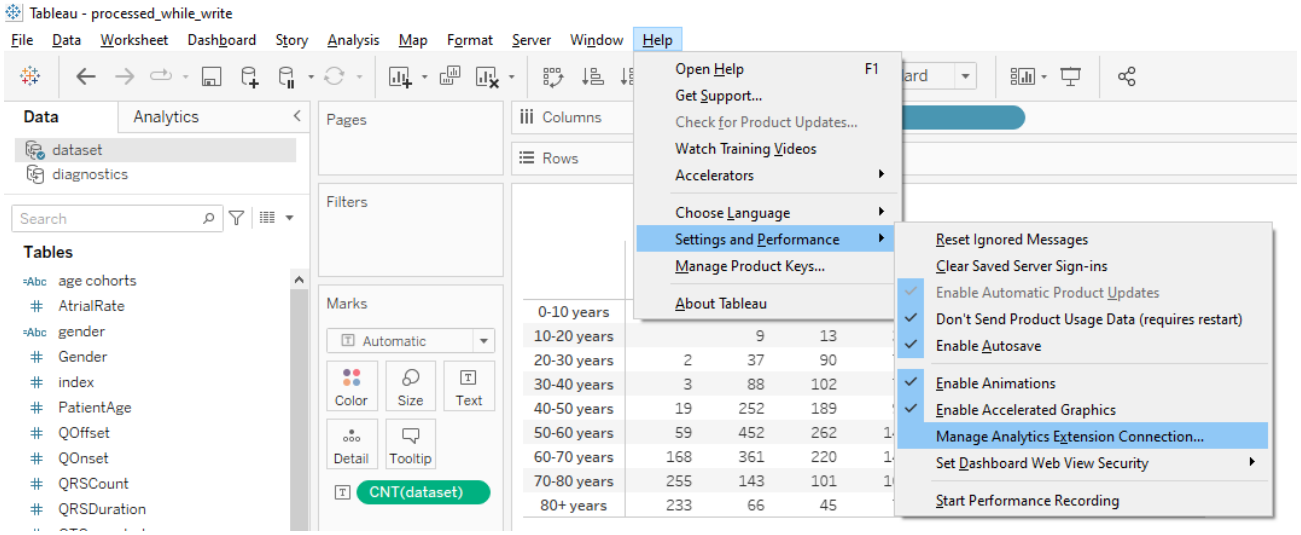

Рисунок 4 – Підключення до серверу TabPy

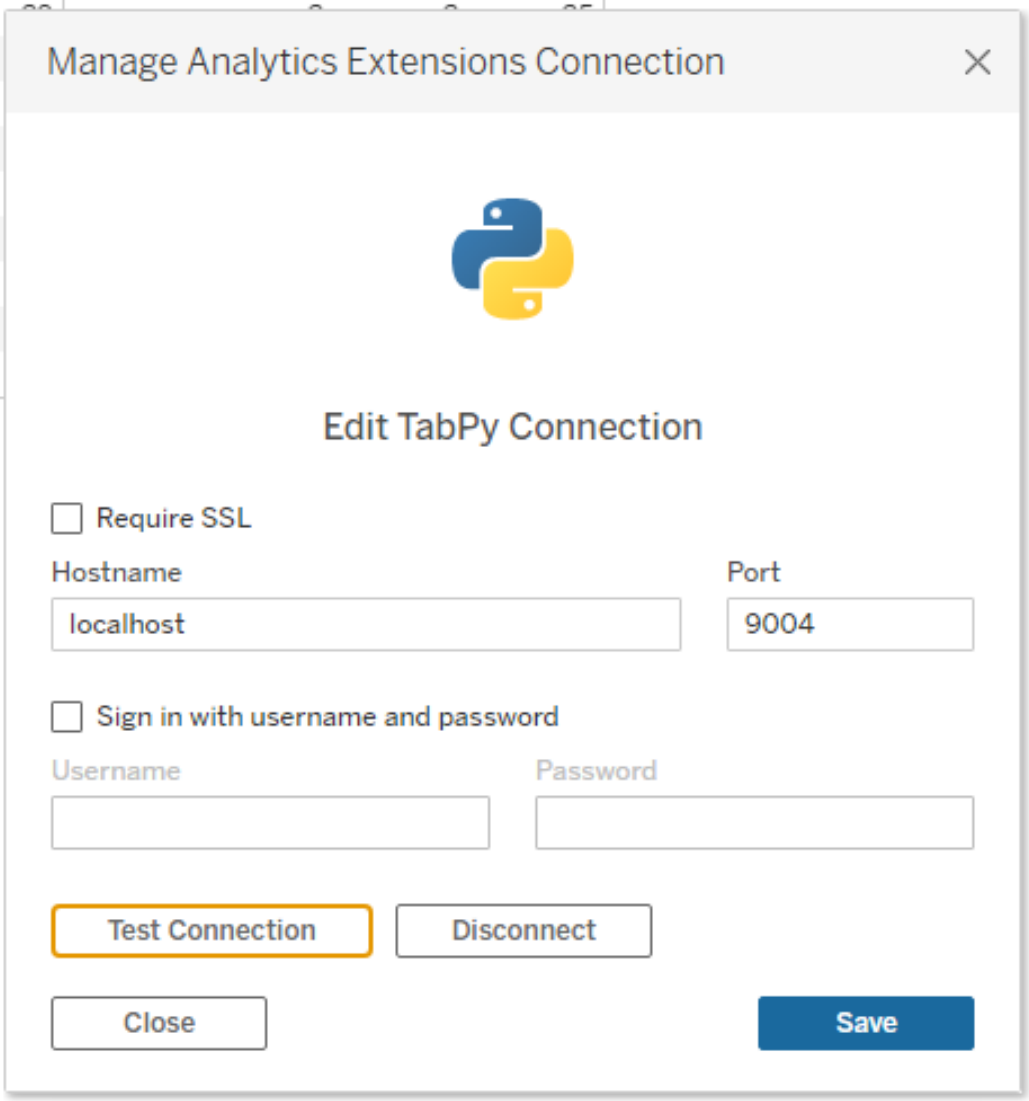

Рисунок 5 – Введеня кредів серверу

Після цього створив параметри (рис. 7.3), через які користувач зможе вводити вхідні параметри для нейромережі (параметри екг-досліджень та інформацію про пацієнта).

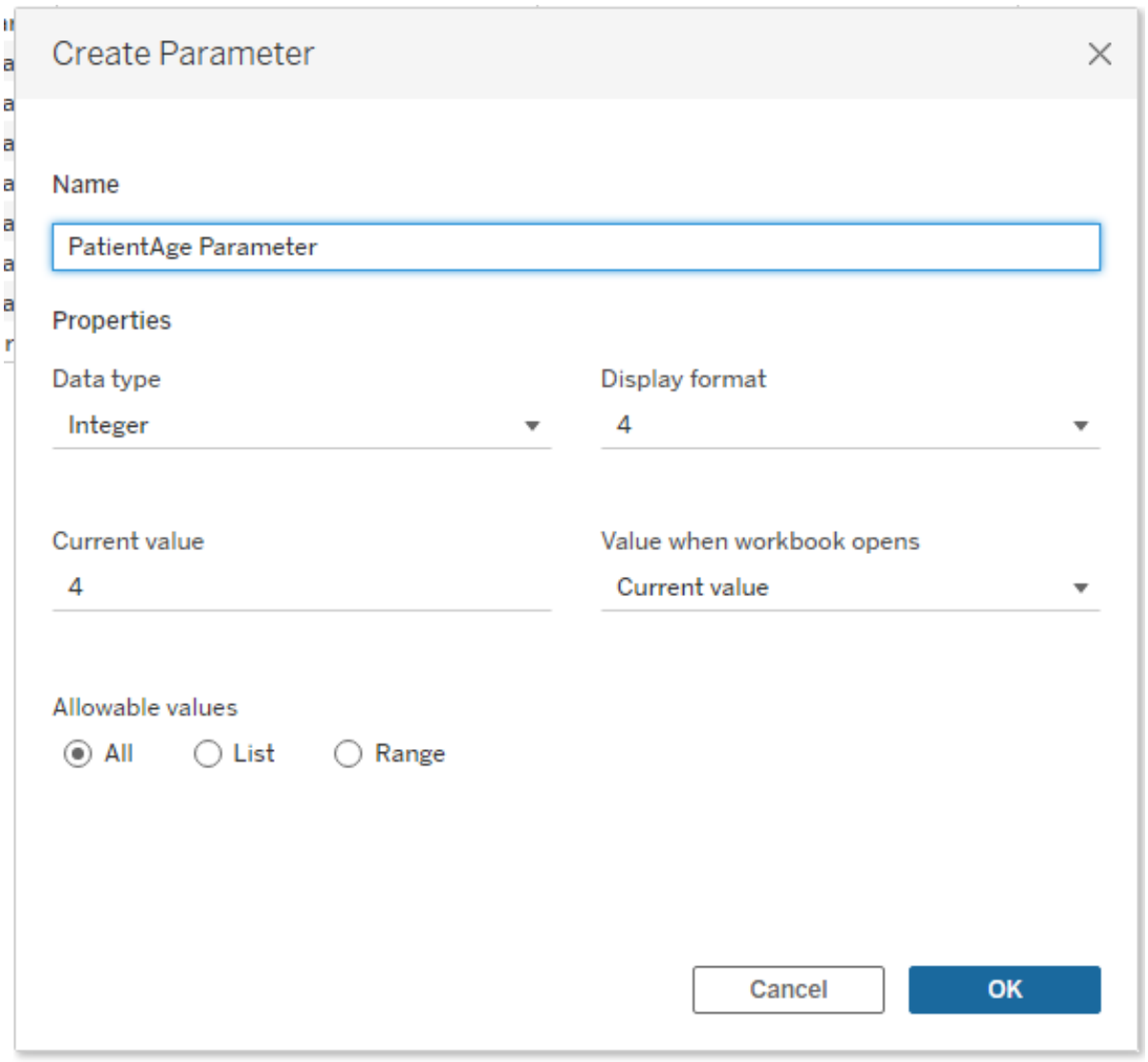

Рисунок 6 – Створення параметра на основі поля PatientAge

Такі параметри необхідно створити для всіх полів (рисунок 7.4).

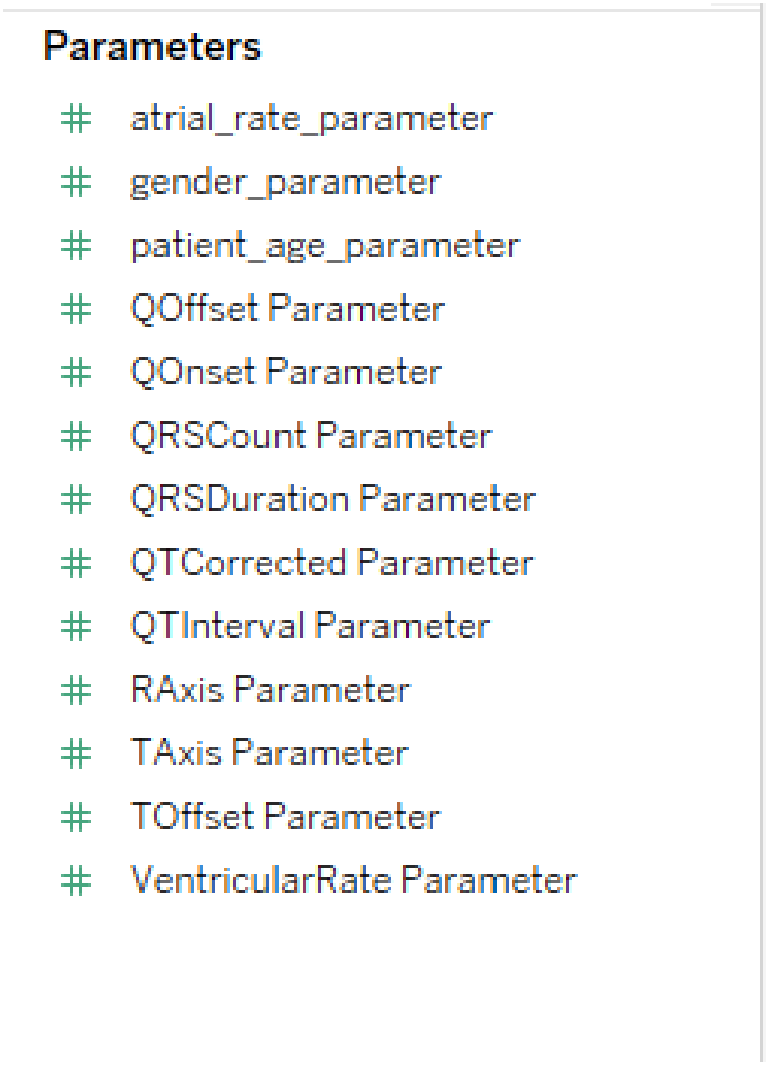

Рисунок 7 – Перелік всіх створених параметрів

Для того, щоб використовувати модель, що знаходиться на сервері – необхідно створити поле, яке буде приймати результат роботи нейромережі на основі вхідних параметрів (рис. 7.5). Також необхідно стоврити аналогічне поле, яке буде повертати ймовірність приналежності обєкту до цього класу.

arytmia_type_predict dataset

Results are computed along Table (across).

```
SCRIPT_STR("return tabpy.query('arytmia_classification_v4',
_arg1, _arg2, _arg3, _arg4, _arg5, _arg6, _arg7, _arg8, _arg9, _arg10, _arg11, _arg12, _arg13 )['response']",
[patient_age_parameter],
[gender_parameter],
[VentricularRate Parameter],
[atrial_rate_parameter],
[QRSDuration Parameter],
[QTInterval Parameter],
[QTCorrected Parameter],
[RAxis Parameter],
[TAxis Parameter],
[QRSCount Parameter],
[QOnset Parameter],
[QOffset Parameter],
[TOffset Parameter])
```

Рисунок 8 – Створення поля для отримання результатів від моделі

Для створення дашборду також додав таблиці та графіки в різних розрізах, щоб показати розподілення пацієнтів по розрізах та загалом, які пацієнти містяться в базі на основі йої було створено нейромережу.

Створено дві вкладки дашборда:

– Вона показує розподіл набору даних в різних розрізах (рис. 7.6);

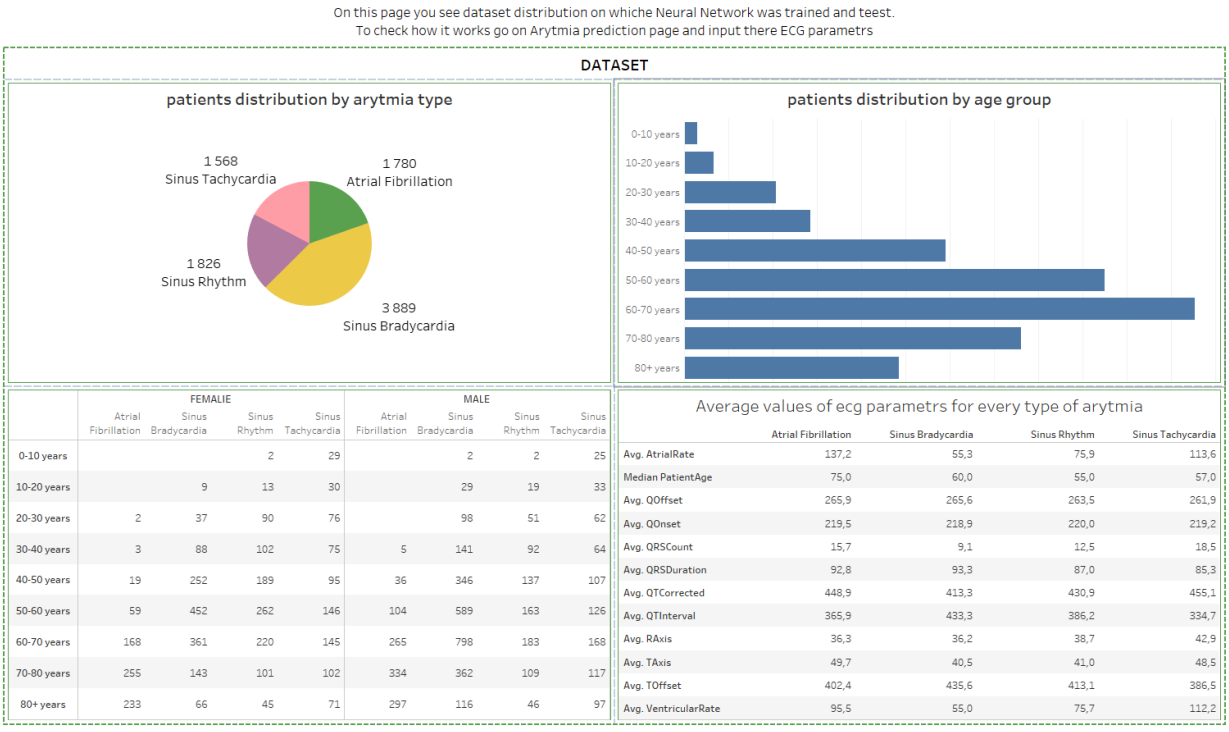

Рисунок 9 – Перша вкладка дашборду

– Створена для користування, на ній зображені вхідні парметри, на основі яких дашборд показує тип аритмії та її ймовірність (рис. 7.7).

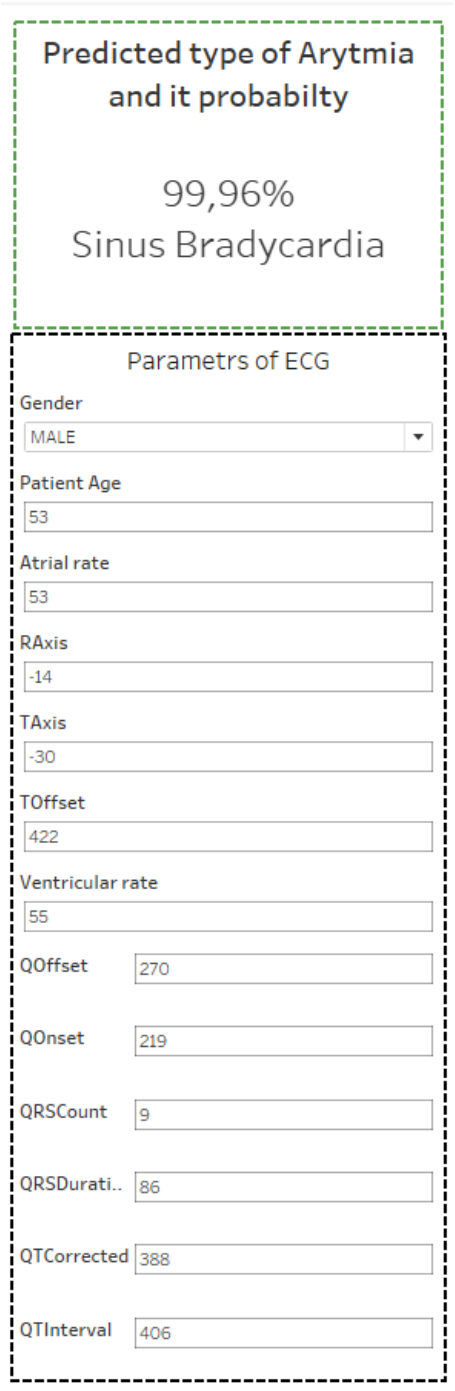

Рисунок 10 – Друга вкладка дашборду

ВИСНОВКИ

Мною було опрацьовано теоретичний матеріал з тем «визначення аритмії серця за даними екг-дослідження», «багато-класова класифікація на основі нейромереж». Було проаналізовано отриманий набір даних та створенно нейромережу за домогою цих даних. Нейромережа показала доволі хорошу точність при визначені виду аритмії. З використанням нейромережі та застосунку візуалізації Tableau було створено створено дашборд у якості сервісу для визначення виду аритмії. Даний сервіс може бути вдосконалений – а саме створення нової нейромережі на основі нових зібраних даниї, яка зможе визначати більше видів аритмій. Також корисним може бути заміна сервісу візуалізації Tableau на власну платформу, що дозволить стати сервісу незалежним від Tableau та більш гнучким для додавання інших сервісів, проте це буде більш ресурсно затратно.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Official documentation of TensorFlow [Електронний ресурс], – Режим доступу до ресурсу: https://www.tensorflow.org/
2. Official documentation of Tableau [Електронний ресурс], – Режим доступу до ресурсу: https://help.tableau.com/current/pro/desktop/en-us/welcome.htm
3. Scikit-learn – sklearn.cluster.AffinityPropagation [Електронний ресурс], – Режим доступу до ресурсу: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AffinityPropagation.html
4. Alday E. A. P. Classification of 12-lead ecgs: the physionet/computing in cardiology challenge 2020 / E. A. P. Alday, A. Gu, A. J. Shah, C. Robichaux, A. K. I. Wong, C. Liu, M. A. Reyna // Physiological measurement. – 2020. – Vol. 41. – №. 12. – P. 124003.
5. Acharya U. R. Automated detection of arrhythmias using different intervals of tachycardia ECG segments with convolutional neural network / U. R. Acharya, H. Fujita, O. S. Lih, Y. Hagiwara, J. H. Tan, M. Adam // Information sciences. – 2017. – Vol. 405. – P. 81-90
6. PhysioNet [Електронний ресурс], – Режим доступу до ресурсу: https://physionet.org/physiobank/database/mitdb/
7. Géron A. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. – “O'Reilly Media, Inc.”, 2022.
8. Francois C. Deep Learning with Python, Manning Publications. – 2017.
9. Sigg D. C. Cardiac electrophysiology methods and models. / D. C. Sigg, P. A. Iaizzo, Y. F. Xiao, B. He – Springer Science & Business Media, 2010.

10. Goldberger A. L. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals / A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, H. E. Stanley // circulation. – 2000. – Vol. 101. – №. 23. – P. 215-220.
11. Chen X. Atrial fibrillation detection based on multi-feature extraction and convolutional neural network for processing ECG signals / X. Chen, Z. Cheng, S. Wang, G. Lu, G. Xv, Q. Liu, X. Zhu// Computer Methods and Programs in Biomedicine. – 2021. – Vol. 202. – P. 106009.
12. Li J. Deep convolutional neural network based ECG classification system using information fusion and one-hot encoding techniques / J. Li, Y. Si, T. Xu, S. Jiang // Mathematical problems in engineering. – 2018. – Vol. 2018. – P. 1-10.