НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра математичного моделювання та аналізу даних

«До захисту допущено»
В.о. завідувача кафедри
_____ І.М. Терещенко
«__» _____ 2024 р.

Дипломна робота

на здобуття ступеня бакалавра

зі спеціальності: 113 Прикладна математика
на тему: **«Порівняння багатошарового перцептрону та $(1+\lambda)$-еволюційного алгоритму для задач класифікації»**

Виконав: студент 4 курсу, групи ФІ-02
Харь Дмитро Федорович

Керівник: асистент кафедри ММАД Яворський О.А. _____

Рецензент: звання, степінь, посада Прізвище І.П. _____

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

Студент _____

Київ — 2024

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені Ігоря СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра математичного моделювання та аналізу даних

Рівень вищої освіти — перший (бакалаврський)
Спеціальність (освітня програма) — 113 Прикладна математика,
ОПП «Математичні методи моделювання, розпізнавання образів та комп'ютерного зору»

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ І.М. Терещенко
«__» _____ 2024 р.

ЗАВДАННЯ
на дипломну роботу

Студент: Харь Дмитро Федорович

1. Тема роботи: «*Порівняння багатошарового перцептрону та* $(1+\lambda)$*-еволюційного алгоритму для задач класифікації*»,

керівник: асистент кафедри ММАД Яворський О.А.,

затверджені наказом по університету №__ від «__» _____ 2024 р.

2. Термін подання студентом роботи: «__» _____ 2024 р.

3. Вихідні дані до роботи:

4. Зміст роботи: *Порівняльний аналіз багатошарового перцептрону (англ. MLP, Multilayer Perceptron) з оптимізаційними алгоритмами в основі яких градієнтний спуск, MLP з оптимізаційним алгоритмом в основі якого одноточкова мутація та* $(1+\lambda)$*-еволюційного алгоритму з кодуванням генетичного програмування (англ.* $(1+\lambda)$*-EA with GP encoding,* $(1+\lambda)$*-evolutionary algorithm with genetic programming encoding), на прикладі задач бінарної та мультикласової класифікації табличних даних та картинок.*

5. Перелік ілюстративного матеріалу: *«Презентація доповіді»*

6. Дата видачі завдання: 10 грудня 2023 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	листопад-грудень 2023 р.	Виконано
2	Огляд та опрацювання опублікованих джерел за тематикою дослідження	грудень 2023 р - лютий 2024 р.	Виконано
3	Написання програмного забезпечення та проведення дослідження	березень-квітень 2024 р.	Виконано
4	Оформлення та опис результатів	травень 2024 р.	Виконано
5	Написання та оформлення дипломної роботи	травень-червень 2024 р.	Виконано
6	Отримання рекомендації до захисту	08.06.2024	Виконано

Студент _____ Харь Д.Ф.

Керівник _____ Яворський О.А.

РЕФЕРАТ

Кваліфікаційна робота містить: ??? стор., ??? рисунки, ??? таблиць, ??? джерел.

У даній роботі розглядаються методи для вирішення задач класифікації, а саме: MLP, який використовує оптимізаційні алгоритми в основі яких градієнтний спуск, MLP, який використовує оптимізаційний алгоритм на основі одноточкової мутації та $(1 + \lambda)$-EA with GP encoding. Ці методи порівнювались в задачах бінарної та мультикласової класифікації табличних даних та картинок.

У ході дослідження було встановлено, що всі три методи здатні досягти однакових метрик у всіх задачах. Найшвидшу сходимість до цих метрик продемонстрував MLP з використанням градієнтного спуску. Тим не менш, $(1 + \lambda)$-EA with GP encoding виділився завдяки здатності легко адаптуватись до умов задачі. Цей метод дозволяє вибирати кількість нащадків і тип мутацій, що надає можливість зосередити пошук рішень у конкретних областях простору рішень. Такий підхід є особливо корисним, коли потрібно зосередитися на важливих регіонах пошуку для вдосконалення рішень.

МАШИННЕ НАВЧАННЯ, ЕВОЛЮЦІЙНІ АЛГОРИТМИ, ГЕНЕТИЧНЕ ПРОГРАМУВАННЯ, МЕТОДИ ОПТИМІЗАЦІЇ, ЕКСПРЕСИВНІ КОДУВАННЯ

ABSTRACT

This paper considers methods for solving classification problems, namely: MLP, which uses optimization algorithms based on gradient descent, MLP, which uses an optimization algorithm based on one-point mutation, and $(1+\lambda)$-EA with GP encoding. These methods were compared in the tasks of binary and multiclass classification of tabular data and pictures.

During the research, it was established that all three methods are able to achieve the same metrics in all tasks. The fastest convergence to these metrics was demonstrated by MLP using gradient descent. Nevertheless, the $(1 + \lambda)$-EA with GP encoding stood out due to its ability to easily adapt to the task conditions. This method allows you to choose the number of offspring and the type of mutations, which makes it possible to focus the search for solutions in specific regions of the solution space. This approach is particularly useful when focusing on important search regions to improve solutions.

MACHINE LEARNING, EVOLUTIONARY ALGORITHMS, GENETIC PROGRAMMING, OPTIMIZATION METHODS, EXPRESSIVE ENCODINGS

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ML — машинне навчання (англ. Machine Learning)

MLP — багатошаровий перцептрон (англ. Multilayer Perceptron)

EA — еволюційний алгоритм (англ. Evolutionary Algorithm)

GP — генетичне програмування (англ. Genetic Programming)

Adam — адаптивна оцінка моменту (англ. Adaptive Moment Estimation)

$(1 + \lambda)$-EA with GP encodings — еволюційний алгоритм, який може використовуватися для вирішення задач класифікації (англ. $(1 + \lambda)$-Evolutionary Algorithm with Genetic Programming encodings).

MLP with gradient descent — багатошаровий перцептрон, який використовує метод на основі градієнтного спуску, в якості оптимізаційного алгоритму.

MLP with single-point mutation — багатошаровий перцептрон, який використовує одноточкову мутацію, в якості оптимізаційного алгоритму.

Фітнес-функція — це функція $F : \mathcal{S} \to \mathbb{R}$, яка відображає представлення рішення S на дійсне число f.

Індивід — I в еволюційних алгоритмах визначається як кортеж $I = (S, f)$, де S є представленням рішення в просторі рішень $\mathcal{S}$, природа S залежить від конкретної проблеми та може варіюватися в широких межах, від двійкових рядків, дійсних векторів, дерев до більш складних структур даних, f — це значення фітнес-функції, пов'язане з індивідом, кількісно оцінюючи якість індивіда як рішення цільової проблеми.

Популяція — P визначається як множина індивідів $P = \{I_1, I_2, \ldots, I_N\}$, де кожен окремий I_i є варіантом вирішення розв'язуваної проблеми.

Кросинговер — C, є бінарною функцією, яка бере два індивіди з

популяції як вхідні дані та створює одне або більше нащадків, потенційно включаючи генетичний матеріал від обох батьків. Формально це можна виразити так: $C : (I_i, I_j) \to (I_{i'}, I_{j'})$, де I_i і I_j є батьківськими індивідами, кожен з яких містить представлення рішення та значення фітнес-функції, $I_{i'}$ і $I_{j'}$ є особинами-нащадками, отриманими в результаті операції кросинговеру.

Мутація — це функція $M : I \to I'$, де I — оригінальний індивід, I' є мутованою особиною з потенційно зміненим представленням рішення S' і відповідним новим значенням фітнес-функції f', яка застосовує стохастичну модифікацію до індивіду, що потенційно призводить до появи нового варіанту рішення.

Контрольованість у контексті $(1 + \lambda)$-еволюційного алгоритму з генетичним програмуванням — визначається як здатність алгоритму дозволяти користувачу точно регулювати його параметри (наприклад, кількість нащадків λ і стратегії мутації), щоб оптимізувати процес пошуку рішення та адаптувати його під специфічні умови задачі.

Precision — це метрика, яка визначає відношення кількості правильно класифікованих позитивних прикладів до загальної кількості прикладів, що були класифіковані як позитивні.

Recall — це метрика, яка визначає відношення кількості правильно класифікованих позитивних прикладів до загальної кількості справді позитивних прикладів.

F1-score — це гармонійне середнє між precision та recall.

ВСТУП

Актуальність дослідження. Використання класифікаційних задач має широкий спектр застосування в різних сферах наукових досліджень та практичних областях. Наприклад, важливо класифікувати, чи особа є носієм певного захворювання, спираючись на ретгенівські знімки або аналізи крові, що дозволяє з високою точністю визначати наявність патологій. З цього випливає необхідність дослідження алгоритмів, які вирішують задачу класифікації, для того, щоб мати більшу гнучкість у налаштуванні процесу пошуку по певним областям простору рішень. Відповідно до цього актуальність дослідження полягає у порівнянні алгоритимів MLP та $(1 + \lambda)$-EA with GP encodings, щоб перевірити чи надає останній можливість контролювати гіперпараметри для більш детального пошуку та можливість краще адаптуватися до поточної задачі.

Метою дослідження є пошук оптимального методу класифікації серед методів MLP with gradient descent, MLP with single-point mutation, $(1 + \lambda)$-EA with GP encodings, для дослідження контрольованості (див. означення у розділі перелік умовних позначень, скорочень і термінів) у розв'язанні задач бінарної та мультикласової класифікації табличних даних та картинок.

Об'єктом дослідження є якісна поведінка MLP та $(1 + \lambda)$-еволюційних алгоритмів для задачі бінарної та мультикласової класифікації.

Предметом дослідження є особливості контролювання алгоритмів на прикладі MLP with gradient descent, MLP with single-point mutation, $(1 + \lambda)$-EA with GP encodings на прикладі застосування до задач бінарної та мультикласової класифікації табличних даних та картинок.

Наукова новизна полягає в дослідженні та порівнянні алгоритмів MLP with gradient descent, MLP with single-point mutation, $(1 + \lambda)$-EA with

GP encodings на прикладі задач бінарної та мультикласової класифікації.

Практичне значення результатів полягає в використанні перелічених вище методів, для задачі класифікації, для покращення контрольованості і збереженню такої ж точності та швидкості, як і в класичних методах.

1 МЕТОДИ ТА ПІДХОДИ ВИРІШЕННЯ ЗАДАЧ КЛАСИФІКАЦІЇ

В даному розділі будуть основні теоретичні відомості про об'єкт дослідження та огляд суміжних робіт в даній сфері.

1.1 Задача класифікації: визначення, види

Класифікація — це процес віднесення об'єкту до певної категорії або класу на основі його характеристик, серед заздалегідь встановленого набору категорій. Класифікація може бути бінарною, багатокласовою, багатомітковою, ієрархічною та інші. Бінарна класифікація - це класифікація, коли кожному об'єкту обирається група з наперед визначеної множини груп в якій знаходиться рівно дві групи; багатокласова класифікація - це класифікація, коли кожному об'єкту обирається група з наперед визначеної множини груп в якій може знаходиться довільна кількість груп. В поточній роботі ми зосередимося на бінарній та багатокласовій класифікації.

Задача класифікації зустрічається в багатьох сферах, наприклад: медицина (діагностика раку на основі зображень МРТ), фінанси (класифікація позичальників як „надійних“ чи „ризикованих“ на основі їхньої кредитної історії), роздрібна торгівля (класифікація покупців за типами покупок для надання персоналізованих знижок), транспорт (розрізнення між легковими авто, вантажівками та мотоциклами на дорозі), освіта (ідентифікація студентів, яким потрібна додаткова допомога в певних предметах), безпека (класифікація електронних листів як „безпечні,“ „спам“ або „фішинг“), біотехнології (розпізнавання мутацій, що спричиняють хвороби).

1.2 Способи вирішення задачі класифікації

Існує декілька способів для вирішення задачі класифікації: класичні статистичні методи (наприклад, логістична регресія [15]), алгоритми машинного навчання (наприклад, метод k-найближчих сусідів [6]), глибинне навчання (за допомогою нейронних мереж), а також задачу класифікації можна вирішувати за допомогою генетичних алгоритмів.

На початку розглянемо методи глибинного навчання для вирішення задач класифікації. Обчислювальним об'єктом в глибинному навчанні є нейронна мережа. Існують різноманітні типи нейронних мереж, але ми будемо їх розглядати на прикладі багатошарового перцептрону, оскільки саме його ми використовуємо для експериментів. Багатошаровий перцептрон складається з шарів нейронів. Кожен нейрон в шарі, приймає вхідні дані з попереднього шару та обчислює вихідний сигнал, який передається наступному шару. Формально штучний нейрон можна описати наступним чином:

$$a = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{1.1}$$

де $x_1, x_2, \ldots, x_n$ — вхідні сигнали до нейрону; $w_1, w_2, \ldots, w_n$ — ваги, що призначені для кожного вхідного сигналу; b — зсув (англ. bias), що додається до суми зважених вхідних сигналів; f — активаційна функція, яка має наступні властивості: нелінійність, дииференційовність, неперервність, монотонність.

В нейронній мережі може бути довільна кількість шарів та в кожному шарі може бути довільна кількість нейронів. Усі вони працюють за вище наведеним принципом: на вхід кожному нейрону в кожному шарі приходить сигнал з попереднього шару і кожний нейрон генерує вихід, якщо це перший шар, то на вхід подаються самі дані. Загалом схема нейронної мережі може виглядати наступним чином (рисунок 1.1).

Далі розглянемо генетичні алгоритми для вирішення задач

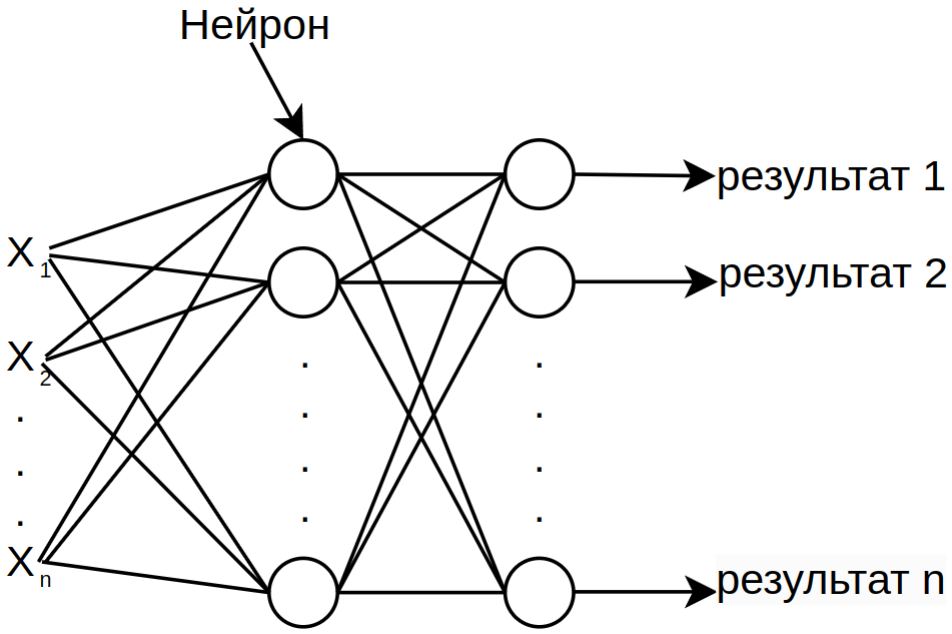

Рисунок 1.1 – Загальна архітектура повнозв'язної нейронної мережі

класифікації. Обчислювальним об'єктом в генетичних алгоритмах є індивід (див. означення в переліку умовних позначень, скорочень і термінів). Індивід може бути представлений різними способами, але ми будемо розглядати представлення, яке найчастіше використовується в генетичному програмуванні, а саме – дерево (приклад дерева – рисунок 1.2). В якості внутрішніх вузлів в дереві можуть бути функції будь якої арності, а в якості листків – ознаки (англ. features) вхідних даних, константи або змінні.

1.3 Метрики оцінки якості моделей та функції втрат, для задач класифікації

Існують різноманітні метрики для оцінювання якості моделі, наприклад точність (англ. accuracy) [12], precision [12], recall [17], f1-score [19]. Формально ці метрики можна записати наступним чином (таблиця **??**).

Основна метрика, що використовується для загальної оцінки якості моделі, — це accuracy. Ця метрика є найбільш інформативною, коли класи в

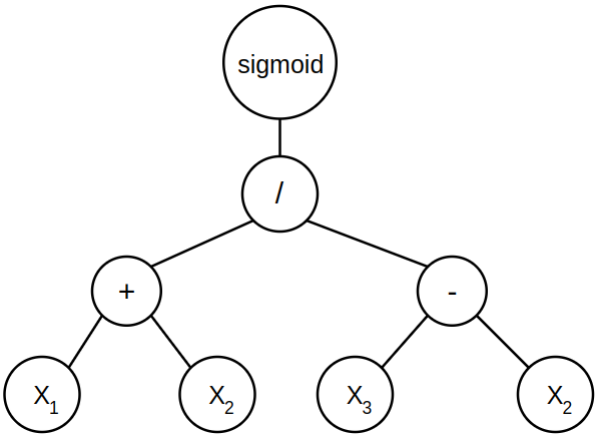

Рисунок 1.2 – Представлення індивіду у вигляді дерева, який отримує на вхід три фічі: X_1, X_2, X_3 та обраховує наступну функцію, яка залежить від цих фіч - $sigmoid\left(\frac{X_1+X_2}{X_3-X_2}\right)$, де $sigmoid(x) = \frac{1}{1+\exp(-x)}$

Таблиця 1.1 – Формули основних метрик якості класифікаційних моделей

Метрика	Формула
Accuracy	$\frac{TP+TN}{TP+TN+FP+FN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
F1-Score	$2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision}+\text{Recall}}$

даних розподілені рівномірно. Проте, в умовах сильного дизбалансу класів accuracy може давати занадто оптимістичну картину ефективності моделі, оскільки вона враховує лише загальну кількість правильних відгадок.

Precision краще використовувати, коли важливіше знизити кількість помилково позитивних результатів. Наприклад, у медичних тестах або в системах, де вартість помилки дуже висока.

Recall є ключовою метрикою, коли важливо виявити всі можливі позитивні випадки. Це критично для ситуацій, де пропуск позитивних результатів може мати серйозні наслідки, наприклад, в системах раннього виявлення захворювань.

F1-Score використовується для оцінки балансу між Precision та

Recall. Ця метрика особливо корисна, коли потрібно врахувати обидві ці характеристики одночасно, наприклад, у контексті інформаційного пошуку та класифікації текстів, де немає явної переваги між помилково позитивними та помилково негативними результатами.

Вибір метрики для конкретної задачі залежить від самої задачі, але гарною практикою є розрахунок одразу декількох метрик, для того, щоб бачити повну картину.

Функції втрат використовуються під час навчання моделей, щоб оптимізувати параметри моделі з метою мінімізації розбіжності між прогнозованими результатами та дійсними даними. Такі функції кількісно оцінюють помилки моделі та на основі значень такої функції оновлюються параметри моделі. Функцій втрат також існує велика кількість, але ми наведемо приклад двох функцій, одна з яких використовується для бінарної класифікації – бінарна крос ентропія, а інша для багатокласової класифікації – крос ентропія. Бінарна крос ентропія та крос ентропія виражаються наступними формулами:

$$\text{binary cross entropy loss} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right) \quad (1.2)$$

$$\text{cross entropy loss} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij}\log(p_{ij}) \quad (1.3)$$

де N – кількість спостережень у наборі даних, M – кількість можливих класів, y_i – фактична мітка класу для і-го спостереження, y_{ij} – бінарний індикатор, який показує чи належить i-те спостереження до класу j, p_i – прогнозована ймовірність, що спостереження i належить до класу з міткою 1, p_{ij} – прогнозована ймовірність, що i-те спостереження належиить до класу j, log – натуральний логарифм.

1.4 Процес навчання моделей для задач класифікації

В цьому підрозділі ми розглянемо, як відбувається процес навчання багатошарового перцептрону та генетичного алгоритму.

Почнемо розгляд з процесу навчання багатошарового перцептрону. Перед початком навчання ініціалізуються ваги мережі. Після того, як ваги були ініціалізовані, мережі подаються на вхід дані. Перший шар нейронів розраховує вихідний сигнал, де кожен нейрон розраховує його за формулою 1.1, далі цей сигнал передається наступному шару, наступний шар розраховує вихідний сигнал, передає його далі і так це повторюється стільки разів, скільки в мережі шарів, останній шар розраховує вихідний сигнал, у випадку бінарної класифікації це буде одне значення для кожного вхідного прикладу з даних, яке буде відображати ймовірність того, що поточний приклад належить до класу 1, у випадку багатокласової класифікації, для кожного вхідного прикладу будуть розраховуватись n - значень, де n – це кількість можливих класів, які будуть відображати ймовірність того, що поточний приклад належить до класу i. Після того, як були розраховані ймовірності належності прикладів до класу/класів, використовуючи значення цих ймовірностей розраховуються функції втрат за формулами 1.2, 1.3 для бінарної та мультикласової класифікації відповідно. Наступний крок є ключовим у навчанні багатошарового перцептрону - зворотнє поширення помилки. Зворотнє поширення помилки полягає в обчисленні градієнтів функції втрат по відношенню до кожного вагового коефіцієнту в мережі, використовуючи правило ланцюгового диференціювання. Обчисленні градієнти використовуються для оновлення ваг у напрямку, що зменшує помилку (зазвичай за допомогою методу градієнтного спуску, або його варіантів, наприклад, Adam [9]). Описаний вище процес повторюється ітеративно певну кількість ітерацій, або поки не буде виконана умова завершення.

Тепер розглянемо процес навчання генетичного алгоритму. Першим кроком ініціалізується популяція (див. означення в переліку умовних позначень, скорочень і термінів) індивідів. Ініціалізація індивідів може відбуватися як повністю випадковим чином, так і з наперед заданими конкретними структурами індивідів. Для кожного індивіду в популяції розраховується фітнес-функція (див. означення в переліку умовних позначень, скорочень і термінів), яка вимірює як добре особина вирішує поставлену задачу. Для задач бінарної та багатокласової класифікації в якості фітнес функції використовуються формули 1.2 та 1.3, в цьому випадку, чим менше буде значення фітнес-функцій тим краще індивід буде пристосований до поточної задачі. Після того, як для кожного індивіду були розраховані фітнес функції, за допомогою методу відбору обираються індивіди, які будуть брати участь у створені наступного покоління. Існує багато методів відбору, ось декілька прикладів: рулетковий відбір [10], турнірний відбір [4] (випадковим чином обирається групка індивідів з усієї популяції і з цієї групки для репродукції обирається той індивід у якого найкраща фітнес функція), стабільний відбір (англ. Steady State Selection) [3]. Далі для генерації наступного покоління, до відібраних індивідів застосовується операція кросинговеру (див. означення в переліку умовних позначень, скорочень і термінів). Обрані індивіди розбиваються на пари і обмінюютьсся частинами своїх хромосом для створення нових індивідів. Цей процес є стохастичним, тобто які саме частини хромосом будуть обмінюватись визначається випадковим чином. До поширених методів кросинговеру відносяться одноточковий кросинговер [16] та багатоточковий кросинговер [21]. При одноточковому кросинговері геноми обох батьків розділяються в одній випадково обраній точці, а потім їх сегменти обмінюються місцями. Багатоточковий кросинговер включає декілька таких точок, що дозволяє формувати потомство з ще більшою генетичною різноманітністю. Після кросинговеру, гени нових особин можуть випадково змінюватись з певною, зазвичай низькою, ймовірністю – цей процес називається

мутацією (див. означення в переліку умовних позначень, скорочень і термінів). Мутація запобігає можливій стагнації всієї популяції в локальному оптимумі, вносячи нові варіанти в генетичний матеріал. Створені особини заміщують деякі, або всі особини в поточній популяції, в залежності від методу відбору. Описаний вище процес повторюється певну кількість поколінь, або поки не буде досягнуто задовільне значення фітнес-функції.

Важливим кроком під час навчання моделей є розділення даних на тренувальну та тестувальну вибірки. Тренувальна вибірка використовується для оновлення ваг моделі, у випадку багатошарового перцептрону, та генерацію нових поколінь, у випадку генетичного алгоритму. Градієнти, у випадку багатошарового перцептрону, розраховуються використовуючи значення функції втрат, яка була отримана в результаті роботи багатошарового перцептрону, який отримав на вході тренувальну вибірку. Фітнес-функції для індивідів, у випадку генетичного алгоритму, також розраховуються використовуючи тільки тренувальну вибірку. Тестувальна вибірка використовується для оцінки якості моделі під час навчання, для того, щоб можна було відслідковувати в який момент почнеться перенавчання (англ. overfit) і використовувати ті параметри моделі, які вона мала до початку перенавчання, це значно покращить узагальнювальну здатність моделей. Важливо зазначити, що тестувальна вибірка ніяким чином не впливає на оновлення параметрів моделей.

1.5 Огляд суміжних робіт

Як вже було описано в розділі 1.2 існують наступні методи вирішення задач класифікації: класичні статистичні методи, методи машинного навчання, методи глибинного навчання та генетичні алгоритми.

Статистичні методи добре підходять в тих випадках, коли нам

важлива інтерпретованість результатів. Прикладом такого методу може бути логістична регресія. Логістична регресія – це статистичний метод, який використовується для задачі бінарної класифікації. Метод базується на логістичній функції, яка оцінює ймовірності приналежності спостережень до однієї з двох категорій. Основною перевагою логістичної регресії є її здатність працювати з даними, де таргетна зміна обмежена інтервалом [0,1], що робить її ідеальною для задач бінарної класифікації. Крім того, модель легко інтерпретувати, оскільки коефіцієнти моделі можуть бути представлені у вигляді шансів (англ. odds ratios). Застосування логістичної регресії виявилося ефективним у багатьох областях, включаючи медицину для діагностики захворювань, в банківській справі для оцінки кредитного ризику, а також в соціальних науках для аналізу виборчих перегонів. Однією з найважливіших областей застосування статистичних методів у медицині є прогнозування серцево-судинних захворювань. Логістична регресія часто використовується для аналізу ймовірності розвитку цих захворювань на основі різних ризикових факторів, таких як вік, кров'яний тиск, холестерин, куріння, сімейний анамнез та інші. В роботі Hosmer і Lemeshow [8], метод логістичної регресії було застосовано для визначення ймовірності настання серцевих нападів у пацієнтів на основі їхнього медичного анамнезу. Модель включала незалежні змінні, які були вибрані на основі клінічного досвіду та попередніх досліджень. Кожен з цих факторів був оцінений на його зв'язок з ризиком розвитку хвороби, і коефіцієнти моделі були інтерпретовані через шансові співвідношення, що дозволило медичним працівникам краще розуміти ризики. Зокрема, було встановлено, що високий кров'яний тиск та високий рівень холестерину значно підвищують шанси на розвиток серцевих захворювань, в той час як регулярні фізичні вправи та здоровий раціон харчування зменшують ці шанси. Ці висновки допомагають лікарям формулювати профілактичні рекомендації та лікувальні стратегії для пацієнтів з підвищеним ризиком. Такий підхід підкреслює значення логістичної регресії не тільки як

аналітичного інструменту, але й як засобу для підтримки клінічних рішень у медицині.

Методи машинного навчання, такі як k-найближчих сусідів (англ. k-Nearest Neighbors, knn), залишаються одними з найпопулярніших через їхню простоту та ефективність у багатьох випадках. У статті Guo і співавторів [7] досліджено модифікований підхід до методу knn, який використовується для класифікації даних у складних застосуваннях, таких як веб-майнінг. У цій статті автори фокусуються на застосуванні цього методу для класифікації великих наборів даних, де традиційні методи kNN часто зазнають труднощів через велику обчислювальну складність і залежність від вибору оптимального значення параметра k. Автори пропонують новий метод – kNNModel, який автоматизує вибір k і використовує передбачувальну модель для підвищення ефективності класифікації. Цей підхід передбачає попереднє моделювання даних за допомогою визначення представників кожної класифікаційної категорії на основі тренувального набору даних. Представники, визначені методом, є центрами кластерів, що представляють групи схожих за характеристиками екземплярів. Свій підхід автори тестують на даних з репозиторію UCI. Використовуючи kNNModel, автори провели експерименти, які показали значне покращення в точності класифікації порівняно зі стандартним методом kNN, особливо в умовах, де потрібно ефективно обробляти великі обсяги даних. Одним з ключових результатів експерименту є те, що застосування моделі kNNModel дозволило значно скоротити час обчислень, необхідний для класифікації нових екземплярів, завдяки використанню попередньо підготовлених представників замість повторного обчислення відстаней до всіх точок даних.

Дослідження Soliman та Abd-elaziem [20] розглядає використання MLP для спеціалізованої задачі класифікації зірок за їхніми спектральними характеристиками. MLP, варіант нейронної мережі, є винятково підходящим для обробки нелінійних завдань, як-от аналіз астрофізичних даних, де потрібно розпізнавати складні взаємозв'язки

між характеристиками. У цьому конкретному дослідженні використовувались дані з понад 100,000 спостережень, кожне з яких містить 18 ознак, таких як інтенсивність на різних довжинах хвиль. Ці особливості були використані для тренування MLP з метою класифікації об'єктів на галактики та зорі. MLP, яке було застосоване в дослідженні, містило кілька прихованих шарів, що дозволяло моделі ефективно вивчити складні патерни у даних. Ефективність класифікації, яку продемонструвала модель, склала 97%. Такий високий показник точності підкреслює здатність MLP ефективно обробляти великі обсяги складних даних та виділяти критично важливі особливості для розпізнавання патернів. Для оптимізації процесу тренування та досягнення максимальної точності було випробувано декілька оптимізаторів, серед яких Adagrad показав найкращі результати з найвищою валідаційною точністю. Ці результати не тільки демонструють потенціал MLP для вирішення астрофізичних задач класифікації, але й вказують на можливість його застосування в інших сферах, де потрібне швидке та ефективне рішення аналогічних задач. Завдяки такому дослідженню, можливо підвищити ефективність використання астрономічних даних та покращити розуміння структури та еволюції космосу.

Дослідження Robu та Holban [18] демонструє застосування генетичних алгоритмів у завданнях класифікації, де використовуються класичні набори даних: Car, Zoo та Mushrooms. В рамках цього дослідження автори впровадили новий підхід до фітнес-функції, який враховує точність прогнозування та інтерпретованість правил. Фітнес-функція, запропонована в їхній роботі, включає в себе вагові коєфіцієнти, які дозволяють регулювати значимість точності прогнозування та інтерпретованості правиил. Це важливо, оскільки в генетичних алгоритмах не тільки важлива здатність правил точно класифікувати дані, але й можливість інтерпретувати ці правила. Такий підхід дозволяє створювати правила, які не тільки ефективні, але й інтерпретовані, що є критично важливим для застосувань, де необхідно

пояснення моделі, наприклад, в медичних діагностиках чи у фінансовому секторі. Експериментальні результати, представлені в дослідженні, показали, що генетичні алгоритми можуть бути порівнянно ефективними з традиційними методами машинного навчання, такими як Наївний Баєс [24] та J48 [2], які також були застосовані до тих же даних. Це свідчить про великий потенціал генетичних алгоритмів в завданнях класифікації, особливо коли необхідно знайти баланс між точністю та інтерпретованістю результатів.

Висновки до розділу 1

В цьому розділі було розглянуто теоретичні відомості про об'єкт та предмет дослідження, а саме про задачі бінарної та багатокласової класифікації та існуючі методи вирішення цих задач. Було здійснено короткиий огляд суміжних розділів, таких як класичні статистичні методи, методи машинного навчання, глибинне навчання та генетитичні алгоритми. Було оглянуто процеси навчання модделей, якіі вирішують задачі класифікаціїї та метрики, що оцінюють якість роботи моделей.

Ми також вказали на важливості використання методів, які легко інтерпретувати та контролювати для сфер, де пояснення моделі є критично важливим.

2 ПІДГОТОВКА ДО ПРОВЕДЕННЯ ДОСЛІДЖЕННЯ

В даному розділі знаходиться огляд основних інструментів та методів аналізу та попередньої обробки даних, також ми зазначимо використані інструменти та ресурси для моделювання.

2.1 Використані інструменти та ресурси

В якості мови програмування було вибрано Python v3.11 [23], це ефективна та гнучка мова програмування, для розв'язання задач машинного навчання, для якої створено велику кількість бібіліотек та ресурсів, які дозволяють ефективно розв'язувати задачі, включаючи задачі бінарної та багатокласової класифікації табличних даних та картинок. Основними бібліотеками для створення моделей були бібліотеки Deap v1.4 [5] та scikit-learn v1.4 [14]. Обидві бібліотеки надають документацію, невеликі навчальні посібники та приклади для пришвидшення побудови моделей.

Бібліотека Deap — це спеціалізована бібліотека для створення еволюційних алгоритмів. Ця бібліотека має реалізовані рішення для різних задач, таких як генетичне програмування, еволюційні стратегії, генетичні алгоритми та багато інших. Вона забезпечує зручний інтерфейс для налаштування та запуску еволюційних експериментів, надаючи широкий набір інструментів для маніпуляції популяціями, відбору, кросинговеру та мутацій. Основними елементами бібліотеки Deap є індивідуми, популяції, фітнес-функції, оператори генетичних алгоритмів, такі як, відбір, кросинговер та мутація. Ця бібліотека також дозволяє налаштовувати багато параметрів, таких як розмір популяції, кількість поколінь, ймовірності мутацій та кросинговеру, що робить її дуже гнучкою для різних задач. Вона підтримує паралельні обчислення, що

значно прискорює процес еволюційного пошуку оптимальних рішень. В даному дослідженні буде використовуватись бібліотека Deap для реалізації $(1 + \lambda)$-EA with GP encodings, що дозволяє досліджувати ефективність та керованість цього алгоритму в контексті задачі класифікації. Зокрема, ми будемо використовувати такі оператори, як турнірний відбір та одноточкову мутацію. Крім того, буде проведено аналіз впливу різних гіперпараметрів, таких як, значення λ та глибина дерева, а також множини термінальних та внутрішніх вузлів, на якість розв'язків та швидкість конвергенції алгоритму.

Бібліотека scikit-learn — це популярна бібліотека для машинного навчання, яка надає великий набір інструментів для задач класифікації, регресії, кластеризації, зниження розмірності та попередньої обробки даних. Вона забезпечує простий і уніфікований інтерфейс для побудови та оцінки моделей машинного навчання, що дозволяє швидко розробляти і тестувати різні алгоритми. Основні компоненти бібліотеки scikit-learn включають реалізовані алгоритми для класифікації, регресії, кластеризації та зниження розмірності, а також методи для попередньої обробки даних. В даному дослідженні бібліотека scikit-learn буде використовуватись для підготовки даних, вибору ознак, побудови та оцінки моделей класифікації. Зокрема, ми будемо використовувати стандартні підходи до попередньої обробки даних, такі як масштабування ознак, зниження розмірності та розділення даних на тренувальну та тестову вибірки. Побудова моделей буде здійснюватись з використанням алгоритму MLP. Результати класифікації будуть оцінюватись за допомогою метрик, таких як accuracy, precision, recall та f1-score. Це дозволить порівняти ефективність різних підходів та обрати найкращий алгоритм для задачі класифікації.

Також були використані наступні бібліотеки: pandas [22] – для завантаження та попередньої обробки даних, optuna [1] – для оптимізації гіперпараметрів моделей, torch [13] та torchvision [11] – для обробки картинок та створення ембідінгів з моделей.

Проаналізувавши різноманітні сервіси, які надають доступ до даних, в якості вебресурсу з даними ми використовуємо вебсайт https://www.kaggle.com/datasets. Kaggle – це платформа для змагань з машинного навчання, яка також надає великий каталог відкритих наборів даних для різноманітних задач, включаючи класифікацію, регресію та кластеризацію. Набори даних на Kaggle часто добре документовані та попередньо оброблені, що дозволяє швидко приступити до експериментів.

2.2 Попередня обробка даних

В даній роботі використовувалися наступнii набори даниих:

– Pima Indians Diabetes Database (посилання: https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database/data) – це набір даних, який часто використовується для задач класифікації в області біомедичних досліджень. Цей датасет був зібраний Національним інститутом діабету, шлункових і ниркових захворювань США. Набір даних містить інформацію про жінок з племені Піма, що проживають в Арізоні, та включає показники здоров'я, які можуть впливати на розвиток діабету. Датасет складається з 768 зразків, кожен з яких має 8 вхідних ознаки і два вихідних класи, які вказують на наявність або відсутність діабету. Всі ознаки числові, що дозволяє легко використовувати їх у машинному навчанні. Датасет складається з наступних ознак: Pregnancies – кількість вагітностей у жінки; Glucose – рівень глюкози у плазмі крові через 2 години після навантажувального тесту; Blood Pressure – діастолічний артеріальний тиск; Skin Thickness – товщина шкірної складки трицепса; Insulin – рівень інсуліну у сироватці крові; BMI – індекс маси тіла; Diabetes Pedigree Function – функція родоводу діабету (враховує генетичну спадковість); Age – вік пацієнта; цільова змінна: Outcome – наявність діабету (0 - відсутній, 1 - наявний). Цей датасет є добре збалансованим з точки зору наявності та відсутності діабету серед обстежених жінок, що робить його придатним для задач

класифікації. Попередня обробка даних для цього датасету зазвичай включає масштабування ознак, обробку пропущених значень (якщо такі є) та розділення даних на тренувальну та тестову вибірки для подальшого навчання і оцінки моделей.

Висновки до розділу 2

Наприкінці розділу знову наводяться коротенькі підсумки.

3 (НАЗВА ТРЕТЬОГО РОЗДІЛУ)

3.1 (якийсь підрозділ)

Подивіться, як нераціонально використовується простір, якщо не писати вступи до розділів. :)

Зазвичай третій розділ присвячено опису практичного застосування або експериментальної перевірки аналітичних результатів, одержаних у другому розділі роботи. Втім, це не обов'язкова вимога, і структура основної частини диплому більш суттєво залежить від характеру поставлених завдань. Навіть якщо у вас є певне експериментальне дослідження, але його загальний опис займає дві сторінки, то краще приєднайте його підроздіром у попередній розділ.

При описі експериментальних досліджень необхідно:

– наводити повний опис експериментів, які проводились, параметрів обчислювальних середовищ, засобів програмування тощо;

– наводити повний перелік одержаних результатів у чисельному вигляді для їх можливої перевірки іншими особами;

– представляти одержані результати у вигляді таблиць та графіків, зрозумілих людському оку;

– інтерпретувати одержані результати з точки зору поставленої задачі та загальної проблематики ваших досліджень.

У жодному разі не потрібно вставляти у даний розділ тексти інструментальних програм та засобів (окрім того рідкісного випадку, коли саме тексти програм і є результатом проведення експериментів). За необхідності тексти програм наводяться у додатках.

Висновки до розділу 3

Висновки до останнього розділу є, фактично, підсумковими під усім дослідженням; однак вони повинні стостуватись саме того, що розглядалось у розділі.

ВИСНОВКИ

Загальні висновки до роботи повинні підсумовувати усі ваші досягнення у даному напрямку досліджень.

За кожним пунктом завдань, поставлених у вступі, у висновках повинен міститись звіт про виконання: виконано, не виконано, виконано частково (І чому саме так). Наприклад, якщо першим поставленим завданням у вас іде «огляд літератури за тематикою досліджень», то на початку висновків ви повинні зазначити, що «у ході даної роботи був проведений аналіз опублікованих джерел за тематикою (...), який показав, що (...)». Окрім простої констатації про виконання ви повинні навести, які саме результати ви одержали та проінтерпретувати їх з точки зору поставленої задачі, мети та загальної проблематики.

В ідеалі загальні висновки повинні збиратись з висновків до кожного розділу, але ідеал недосяжний. :) Однак висновки не повинні містити формул, таблиць та рисунків. Дозволяється (та навіть вітається) використовувати числа (на кшталт «розроблена методика дозволяє підвищити ефективність пустопорожньої балаканини на 2.71%»).

Наприкінці висновків необхідно зазначити напрямки подальших досліджень: куди саме, як вам вважається, необхідно прямувати наступним дослідникам у даній тематиці.

ЛІТЕРАТУРА

[1] Takuya Akiba та ін. “Optuna: A Next-generation Hyperparameter Optimization Framework”. В: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019.

[2] N.Sarav anaN та V.Gaya thri. “Performance and Classification Evaluation of J48 Algorithm and Kendall’s Based J48 Algorithm (KNJ48)”. В: *International Journal of Computer Trends and Technology* 59 (2018), с. 73—80. URL: https://api.semanticscholar.org/CorpusID:69700602.

[3] Juan Durillo та ін. “On the Effect of the Steady-State Selection Scheme in Multi-Objective Genetic Algorithms”. В: т. 5467. Квіт. 2009, с. 183—197. ISBN: 978-3-642-01019-4. DOI: 10.1007/978-3-642-01020-0_18.

[4] Yongsheng Fang та Jun li. “A Review of Tournament Selection in Genetic Programming”. В: жовт. 2010, с. 181—192. ISBN: 978-3-642-16492-7. DOI: 10.1007/978-3-642-16493-4_19.

[5] Félix-Antoine Fortin та ін. “DEAP: Evolutionary Algorithms Made Easy”. В: *Journal of Machine Learning Research* 13 (лип. 2012), с. 2171—2175.

[6] Gongde Guo та ін. “KNN Model-Based Approach in Classification”. В: *On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE*. За ред. Robert Meersman, Zahir Tari та Douglas C. Schmidt. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, с. 986—996. ISBN: 978-3-540-39964-3.

[7] Gongde Guo та ін. “KNN Model-Based Approach in Classification”. В: (серп. 2004).

[8] D.W. Hosmer та S. Lemeshow. *Applied Logistic Regression*. Applied Logistic Regression. Wiley, 2004. ISBN: 9780471654025. URL: https://books.google.com.ua/books?id=Po0RLQ7USIMC.

[9] Diederik P Kingma та Jimmy Ba. “Adam: A method for stochastic optimization”. В: *arXiv preprint arXiv:1412.6980* (2014).

[10] Adam Lipowski та Dorota Lipowska. “Roulette-wheel selection via stochastic acceptance”. В: *Physica A: Statistical Mechanics and its Applications* 391 (вер. 2011). DOI: 10.1016/j.physa.2011.12.004.

[11] TorchVision maintainers та contributors. *TorchVision: PyTorch’s Computer Vision library*. https://github.com/pytorch/vision. 2016.

[12] Antonio Menditto, Marina Patriarca та Bertil Magnusson. “Understanding the meaning of accuracy, trueness and precision”. В: *Accreditation and Quality Assurance* 12 (жовт. 2007), с. 45—47. DOI: 10.1007/s00769-006-0191-z.

[13] Adam Paszke та ін. “Automatic differentiation in PyTorch”. В: (2017).

[14] F. Pedregosa та ін. “Scikit-learn: Machine Learning in Python”. В: *Journal of Machine Learning Research* 12 (2011), с. 2825—2830.

[15] Joanne Peng, Kuk Lee та Gary Ingersoll. “An Introduction to Logistic Regression Analysis and Reporting”. В: *Journal of Educational Research - J EDUC RES* 96 (вер. 2002), с. 3—14. DOI: 10.1080/00220670209598786.

[16] Riccardo Poli та W. B. Langdon. “Genetic Programming with One-Point Crossover”. В: *Soft Computing in Engineering Design and Manufacturing*. За ред. P. K. Chawdhry, R. Roy та R. K. Pant. London: Springer London, 1998, с. 180—189.

[17] David Powers та Ailab. “Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation”. В: *J. Mach. Learn. Technol* 2 (січ. 2011), с. 2229—3981. DOI: 10.9735/2229-3981.

[18] Raul Robu та Holban Stefan. “A genetic algorithm for classification”. В: трав. 2011, с. 52—56.

[19] Marina Sokolova, Nathalie Japkowicz та Stan Szpakowicz. “Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation”. В: т. Vol. 4304. Січ. 2006, с. 1015—1021. ISBN: 978-3-540-49787-5. DOI: 10.1007/11941439_114.

[20] Tamer Soliman та Ayman Abd-elaziem. “A Multi-Layer Perceptron (MLP) Neural Networks for Stellar Classification: A Review of Methods and Results”. В: *International Journal of Advances in Applied Computational Intelligence* 3 (серп. 2023). DOI: 10.54216/IJAACI.030203.

[21] William M. Spears та Kenneth A. De Jong. “An Analysis of Multi-Point Crossover”. В: за ред. GREGORY J.E. RAWLINS. Т. 1. Foundations of Genetic Algorithms. Elsevier, 1991, с. 301—315. DOI: https://doi.org/10.1016/B978-0-08-050684-5.50022-7. URL: https://www.sciencedirect.com/science/article/pii/B9780080506845500227.

[22] The pandas development team. *pandas-dev/pandas: Pandas*. Вер. latest. Лют. 2020. DOI: 10.5281/zenodo.3509134. URL: https://doi.org/10.5281/zenodo.3509134.

[23] Guido Van Rossum та Fred L. Drake. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN: 1441412697.

[24] Vikramkumar, B Vijaykumar та Trilochan. “Bayes and Naive Bayes Classifier”. В: 2014. URL: https://api.semanticscholar.org/CorpusID:10272111.