

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіоелектроніки

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ

ПРОГНОЗУВАННЯ ФІНАНСУВАННЯ СОЦІАЛІЗАЦІЇ СИРИТ У

ЗАКЛАДАХ ВИЩОЇ ОСВІТИ

RESEARCH AND SOFTWARE IMPLEMENTATION OF THE METHODS

FOR FORECASTING FUNDING THE SOCIALIZATION OF ORPHANS IN

HIGHER EDUCATION INSTITUTIONS

Виконав: студент(ка) 2 курсу, групи КНТ-120м

Спеціальності 121 Інженерія програмного

забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Злиденний А. О.

(прізвище та ініціали)

Керівник Олійник А. О.

(прізвище та ініціали)

Рецензент Скрупський С. Ю.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Інститут , факультет ФКНТ

Кафедра програмних засобів

Ступінь вищої освіти магістр

Спеціальність 121 Інженерія програмного забезпечення

(код і найменування)

Освітня програма (спеціалізація) Інженерія програмного забезпечення

(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.

С. О. Субботін

“ ” 20 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Злиденного Артура Олександровича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів прогнозування фінансування соціалізації сиріт у закладах вищої освіти

Research and Software Implementation of the Methods for Forecasting funding the socialization of orphans in higher education institutions

керівник проєкту (роботи) Олійник А.О. д.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “05” листопада 2021 року

№ 435

2. Строк подання студентом проєкту (роботи) 13 грудня 2021 року

3. Вихід Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити 1. Аналіз предметної області 2. Вибір інструментів

розробки програмного забезпечення 3. Основні рішення щодо реалізації компонентів системи 4. Керівництво програміста 5 Тестування та експериментальне дослідження програми

4. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

5. Консультанти розділів проєкту (роботи)

Розділи	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Олійник А. О., професор		
Нормоконтроль	Камінська Ж. К., асистент		

6. Дата видачі завдання “02 ” вересня 2021 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2-4 тижні	Розділ 1
3	Вибір засобів розробки програмного продукту	5 тиждень	Розділ 2
	Розробка архітектури програми	6 тиждень	Розділ 3
4	Розробка програмного продукту	7-11 тижні	Розділ 4-5
5	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	12 тиждень	Додатки
6	Захист роботи	13 тиждень	

Студент(ка)

Злиденний А. О.
(підпис) (прізвище та ініціали)

Керівник проекту (роботи)

Олійник А. О.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра
122 с., 18 таблиці, 25 рис., 3 додатки, 36 джерел.

ГЕНЕТИЧНИЙ АЛГОРИТМ, НЕЙРОННІ МЕРЕЖІ,
ПРОГНОЗУВАННЯ, ЗАКЛАД ВИЩОЇ ОСВІТИ, MATLAB, C++.

Об'єкт дослідження – процес прогнозування витрат на основі інтелектуальних технологій.

Предмет дослідження – методи для прогнозування фінансування для дітей – сиріт які вступили та навчаються у ЗВО України.

Мета роботи – дослідження та модифікація еволюційних методів прогнозування.

Метод дослідження – традиційні моделі та еволюційні методи прогнозування (нейронні мережі та генетичний алгоритм).

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, мова програмування C++, середовище програмування “MatLab”, персональний комп'ютер з процесором AMD A10 – 5700 APU під управлінням операційної системи Microsoft Windows 10.

Результати. Створено модифікацію методу для визначення фінансування дітей – сиріт, які вступили та навчаються у ЗВО України.

Висновки. Розроблено модифікований метод, який дозволяє спрогнозувати фінансування дітей – сиріт які вступили та навчаються у ЗВО України на основі даних.

Галузь використання – програмний продукт розроблений для аналітиків в соціально – технічних галузях.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 122 pages, 26 figures, 25 tables, 3 appendixes, 36 sources.

GENETIC ALGORITHM, NEURAL NETWORKS, FORACASTING, HIGHER EDUCATION INSTITUTIONS, MATLAB, C++.

The object of research is the process of forecasting costs based on intelligent technologies.

The subject of research - methods for forecasting funding for orphans in free economic zones of Ukraine.

The purpose of the work is to study and modify the methods of evolutionary forecasting methods.

Research method - traditional models, neural networks and genetic algorithm.

Materials, methods and technical means: structural and object-oriented programming, C ++ programming language, MatLab programming environment, personal computer with AMD A10 - 5700 APU processor running Microsoft Windows 10 operating system.

Results. A modification of the algorithm for determining the funding of orphans studying in free economic zones or vocational colleges has been created.

Conclusions. A modified algorithm has been developed to finance orphans based on data.

Field of application - a software product designed for analysts in the socio - technical fields.

ЗМІСТ

Перелік скорочень та умовних позначень	10
ВСТУП	11
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	13
1.1 Аналіз поняття проблеми та задачі з приводу фінансування...	13
1.2 Огляд існуючих систем прогнозування у вирішенні проблеми...	14
1.2.1 Програма fxSolver	14
1.2.2 Програма GNU Octave	16
1.2.3 Програма “MatLab”	17
1.2.4 Програма Souolver.....	20
1.2.5 Програма R	21
1.2.6 Програма S. Відмінність між R та S середовищами	22
1.2.7 Порівняння систем прогнозування на основі основних характеристик	23
1.3 Висновок з поставленою задачею та з системами прогнозування.....	25
1.4 Методи прогнозування	26
1.4.1 Методи статистики для вирішення задачі.....	26
1.4.2 Метод “дерева”. Структура та переваги перед іншими методами	26
1.4.3 Штучні нейронні мережі. Структура та переваги перед іншими методами	30
1.4.4 Багатошарова нейронна мережа	32

1.4.5 Вибірковий метод аналізу. Актуальність та головні задачі методу	35
1.4.6 Логістична регресія. Значення методу та головна роль	36
1.4.7 Порівняльна характеристика методів аналізу.....	38
1.5 Постановка задачі	41
1.6 Висновки за розділом 1	43
2 ВИБІР ІНСТРУМЕНТІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	44
2.1 Вибір мови програмування	44
2.1.1 Мова програмування C++. Відмінність між мовою C	44
2.1.2 Мова програмування C#. Порівняння з мовою C++	45
2.1.3 Мова програмування Python. Переваги та недоліки в порівнянні з іншими мовами.....	47
2.1.4 Мова програмування MatLab.....	49
2.2 Порівняння мов програмування	50
2.3 Головні бібліотеки для реалізації задач в C++.....	52
2.3.1 Ознайомлення с бібліотекою <Cmath>. Детальний опис функцій.....	52
2.3.2 Інструментарії для створення нейронних мереж.....	54
2.3.3 Інструментарій Dlib	54
2.3.4 Інструментарій Caffé	56
2.3.5 Платформа TensorFlow	56
2.4 Порівняльна характеристика інструментаріїв для нейронної мережі.....	57
2.5 Висновки за розділом 2	58

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ.....	60
3.1 Структура програмного продукту	60
3.2 Опис коцепції роботи нейронної мережі з “MatLab”	62
3.3 Детальний опис роботи “MatLab” з генетичними алгоритмами.....	64
3.4 Огляд основних методів для нейронної мережі	65
3.5 Порівняння та вибір генетичного алгоритму	71
3.6 Модифікація генетичного алгоритму	75
3.7 Висновки за розділом 3	80
4 КЕРІВНИЦТВО ПРОГРАМІСТА	82
4.1 Призначення та умови застосування програми	82
4.1.1 Призначення програми	82
4.1.2 Умови застосування програми	82
4.2 Звернення до програми.....	83
4.3 Вхідні та вихідні дані.....	84
4.4 Повідомлення програмісту	87
4.5 Висновки за розділом	88
5 КЕРІВНИЦТВО КОРИСТУВАЧА.....	89
5.6 Призначення та умови застосування програми	89
5.6.1 Призначення програми	89
5.6.2 Умови застосування програми	89
5.2 Виконання програми.....	90
5.2.1 Екран входу	90
5.2.2 Екран роботи алгоритму	91

5.3 Повідомлення оператору.....	92
5.4 Висновки за розділом	92
6 ЕКСПЕРИМЕНТИ ТА РЕЗУЛЬТАТИ	93
6.1 Проведення експериментів	93
6.2 Висновки за розділом 6	95
Висновки	96
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	98
Додаток А Технічне завдання.....	101
Додаток Б Код програми	105
Додаток В Слайди презентації.....	115

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАЧЕНЬ

BAIR	– Berkeley AI Research;
FFNN	– Feed-Forward Neural Networks
GHz	– Одиниця частоти Герц;
GIMP	– GNU Image Manipulation Program;
LSTM	– Long Short-Term Memory
PC	– Персональний комп'ютер;
RNN	– Recurrent Neural Network
ГА	– Генетичний алгоритм;
ДС	– Дитина сирота;
ДСП	– Динамічна самоорганізація параметрів;
ЗВО	– Заклад Вищої Освіти;
НМ	– Нейронна мережа;
ОГА	– Острівний генетичний алгоритм;
РВКіМ	– Генетичний алгоритм з кросинговером та мутацією;
СС	– Соціальна сирота;
ШНМ	– Штучна нейронна мережа.

ВСТУП

Прогнозування — це метод, в якому використовується як накопичений у минулому досвід, так і поточні припущення щодо майбутнього з метою його визначення. Якщо прогнозування виконано якісно, результатом буде картина майбутнього, яку можна застосовувати як основу для планування фінансування в подальші роки. Тому прогнозування в сфері фінансування — одна із найважливіших сфер на сьогодні. Особливо сьогодні, коли дуже скрутне становище в сфері фінансування випускників дитячих будинків, котрі вступають до лав ЗВО. Станом на середину та другу половину 2021 року в Україні було більш як 30 тисяч дітей-сиріт і дітей, позбавлених батьківського піклування. До початку карантину в Україні 92 тисячі дітей навчалися приблизно в 600 інтернатних закладах (окрім тимчасово окупованих територій України). Необхідно спрогнозувати фінансування на наступний рік дітям — сиротам, які поступають до лав та навчаються у ЗВО на території України. Ці підрахунки можуть бути зроблені завдяки методам прогнозування, бо завдяки цим методам можна більш детально підрахувати дітей — сиріт, які навчаються у ЗВО та будуть вступати до нього [1].

Головні різновиди прогнозів, що часто використовуються разом із плануванням діяльності організації, такі:

- економічні, сутність яких полягає у передбаченні загального стану економіки для дітей — сиріт, які поступають до лав ЗВО та навчаються в них;
- прогнози на основі опитувань та дослідів дають можливість, використовуючи багато галузей знань, передбачити, що станеться у складних ситуаціях;
- соціальне прогнозування використовується для передбачення змін у соціальних установках людей та стані суспільства.

Саме останній різновид методу прогнозування підходить для вирішення поставленої задачі, бо саме соціальне прогнозування працює з людьми на конкретній території. Також при вирішенні задачі найкраще підходить метод кількісного прогнозування. Метод застосовується для прогнозування, якщо є підстави вважати, що діяльність у минулому мала певну тенденцію, яку можна продовжити в майбутньому, і коли наявної інформації достатньо для виявлення статистично достовірних тенденцій або залежностей. Крім того, розробник повинен знати, як використати кількісну модель, та пам'ятати, що вигоди від прийняття ефективнішого рішення мають покривати витрати на створення моделі. На сьогодні розробник має дуже великий інвентар програм, які здатні повністю та чітко виконати поставлену задачу та розробити в програмі алгоритм дії та на виході мати готовий результат.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз поняття проблеми та задачі з приводу фінансування

Проблема фінансування дітей – сиріт, які вступають та навчаються до ЗВО дуже велика для України на сьогоднішній день. Тож, перед розробником постає дуже велика та складна задача – розробити метод для прогнозування фінансування дітей – сиріт, які вступають до ЗВО та навчаються в ньому. Кожен рік до учбових закладів поступає велика кількість абітурієнтів, задачею аналітиків та статистиків – спрогнозувати фінансування. Необхідно проаналізувати загальновідомі методи та розробити або модифікувати метод для прогнозування фінансування дітей – сиріт (ДС), які вступають до лав ЗВО та навчаються в ньому [2].

Дипломна кваліфікаційна робота магістра має декілька цілей – характеристика та аналіз найкращого методу з усіх можливих для реалізації поставленої задачі:

- модифікація вибраного алгоритму;
- з'єднання його з середовищем програмування.

Проблема фінансування ДС дуже велика та вікрита на території України, бо в Україні напочаток 2021 року на обліку дітей-сиріт та дітей, позбавлених батьківського піклування, перебувало майже 70 тисяч осіб. Такі дані озвучили в Міністерстві соціальної політики. За минулий рік для цієї категорії дітей збільшили розмір допомоги, а також підвищили розмір грошового забезпечення батькам - вихователям і прийомним батькам [3].

Станом на 2021 рік в Україні 2 серпня завершився основний етап вступної кампанії – подання електронних заяв на вступ до закладів вищої освіти для отримання ступеня бакалавра та магістра на основі повної загальної середньої освіти. Зареєстровано більше 1 мільйон заяв, серед яких приблизно 15 – 20 % це діти – сироти. Згідно з цими цифрами, кожна дитина – сирота поступає на бюджет до будь – якого учбового закладу. Звісно, що в цьому році діти – сироти, які закінчили школи, інтернати поступили вже до

лав ЗВО. На сьогоднішній день згідно зі статистикою минулого року в 2021 році цифри координально не змінилися, також приблизно 15 – 18 відсотків з усіх абітурієнтів, які успішно потрапили на навчання до ЗВО – це діти – сироти [3].

Перед розробником постає задача взяти статистику за минули роки та на основі цих статистик спрогнозувати фінансування на подальші роки для фінансування дітей – сиріт, які успішно вступили до ЗВО та навчаються в ньому [3].

1.2 Огляд існуючих систем прогнозування у вирішенні проблеми

1.2.1 Програма fxSolver

fxSolver – це програма для вирішення математичних задач, бібліотека рівнянь, калькулятор для побудови графіків, а також помічник в питаннях науки і техніки.

За допомогою даної програми розробник може обчислювати кілька рівнянь одночасно, редагувати існуючі формули і створювати нові, обчислювати велику кількість значень, складати графіки, пов'язувати результати, вирішувати повні задачі [4].

fxSolver - це безкоштовний інструмент, розроблений командою інженерів і програмістів з єдиною метою - надати унікальний, корисний і безкоштовний сервіс. Мета команди розробників - дозволити розробникам - початківцям, інженерам вирішувати математичні завдання, не обов'язково знайомлячись з професійним математичним програмним забезпеченням або мовами програмування. На рисунку 1.1 можна побачити приклад роботи в програмі fxSolver.

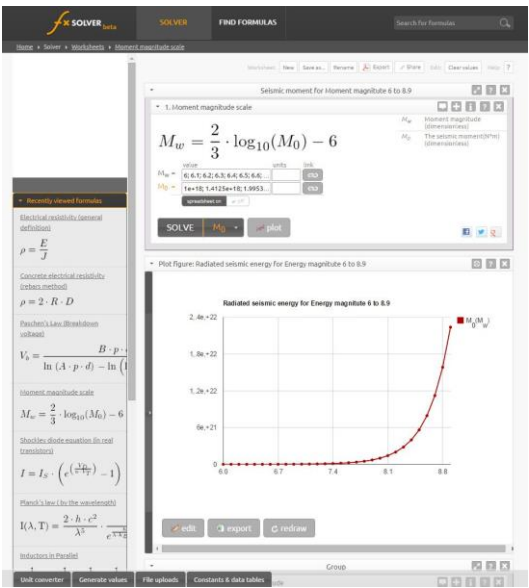


Рисунок 1.1 – Приклад програми fxSolver

До переваг fxSolver можна віднести:

- маленький розмір. Пакет fxSolver дуже маленький та швидко завантажується.
- простота інтерфейсу. Головний екран більш нагадує мобільний додаток, ніж продукт РС.

Недолік програма має один – нездатність розраховувати, або простота вирішення поставленої задачі.

Характеристики fxSolver можна побачити в таблиці 1.1.

Таблиця 1.1 – Характеристики програми fxSolver

Операційні системи	Linux	Android та IOS	Windows
Навчання	Документація та відеоуроки	Online та базова документація	Документація та відеоуроки
Підтримка	Online	Online	24/7
Останнє оновлення	11.07.2021 Патч 6.5.1	26.08.2019 Патч 5.2	24.02.2018 Патч для версій Windows XP,7,8,10

1.2.2 Програма GNU Octave

GNU Octave - це комп'ютерна програма для виконання чисельних розрахунків, яка в основному сумісна з “MatLab”. Частина проекту GNU. Це вільне програмне забезпечення відповідно до умов GNU General Public License. GNU Octave - це інтерпретована мова високого рівня, в першу чергу призначена для чисельних розрахунків. Програма надає можливості для чисельного рішення лінійних і нелінійних задач, а також для виконання інших чисельних експериментів. Також програма надає широкі графічні можливості для візуалізації даних і маніпулювання ними. Інтерпретатор Octave може бути запущений в режимі графічного інтерфейсу користувача (вперше в версії 4.0.0), як консолі або викликаний як частина сценарію оболонки. Octave зазвичай використовується через його інтерактивний інтерфейс, але його також можна використовувати для написання неінтерактивних програм. На рисунку 1.2 можна побачити головне меню програми [5].

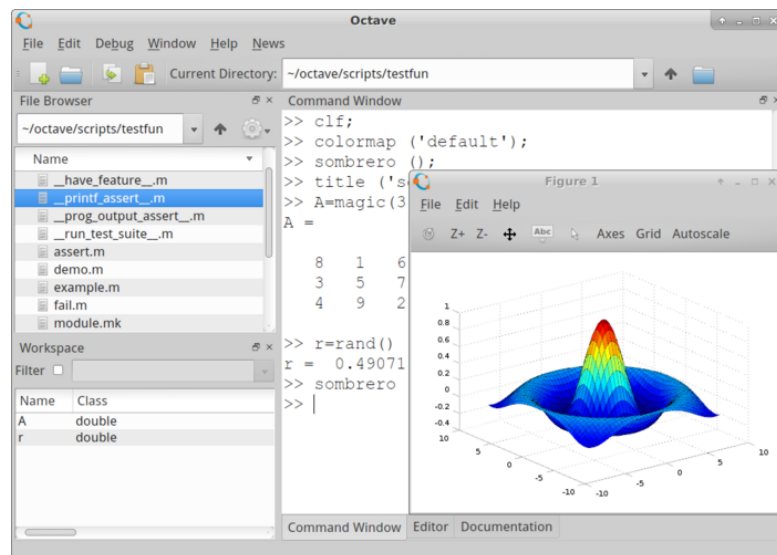


Рисунок 1.2 – Головне меню програми GNU Octave

Програма GNU Octave – дуже схожа на програму “MatLab” – тож, має однакові переваги на недоліки серед інших програм. Слід визначити та

віднести до переваг, що програма розробляє комп'ютерну модель, що описує динамічні процеси у впродовж робочого циклу. Характеристики програми GNU Octave можна побачити на таблиці 1.2.

Таблиця 1.2 – Основні характеристики програми GNU Octave

Операційні системи	Linux, Unix.	Cygwin	Windows (Android)
Навчання	Документація	Online та базова документація	Online та вся документація
Підтримка	Online	24/7	24/7
Останнє оновлення	11.07.2021 Патч 6.3.0	26.02.2020 Патч 5.2	20.03.2019 Робота з Windows XP та більш новими версіями

1.2.3 Програма “MatLab”

MatLab - це високорівнева мова технічних розрахунків, інтерактивне середовище розробки алгоритмів і сучасний інструмент аналізу даних. MatLab в порівнянні з традиційними мовами програмування (C / C ++, Java, Pascal, FORTRAN) дозволяє на порядок скоротити час вирішення типових завдань і значно спрощує розробку нових алгоритмів.

MatLab являє собою основу всього сімейства продуктів MathWorks і є головним інструментом для вирішення широкого спектра наукових і прикладних задач, в таких областях як: моделювання об'єктів і розробка систем управління, проектування комунікаційних систем, обробка сигналів і зображень, вимірювання сигналів і тестування, фінансове моделювання, обчислювальна біологія [6].

Ядро MatLab дозволяє максимально просто працювати з матрицями реальних, комплексних і аналітичних типів даних і зі структурами даних і таблицями пошуку. ”MatLab” містять задану вбудовані функції лінійної алгебри, швидкого перетворення Фур'є, функції для роботи з поліномами,

функції базової статистики та чисельного рішення диференціальних рівнянь; розширені математичні бібліотеки для Intel MKL. Всі вбудовані функції ядра MatLab розроблені й оптимізовані фахівцями і працюють швидше або так само, як їх еквівалент на C/C ++. Приклад програми MatLab можна побачити на рисунку 1.3.

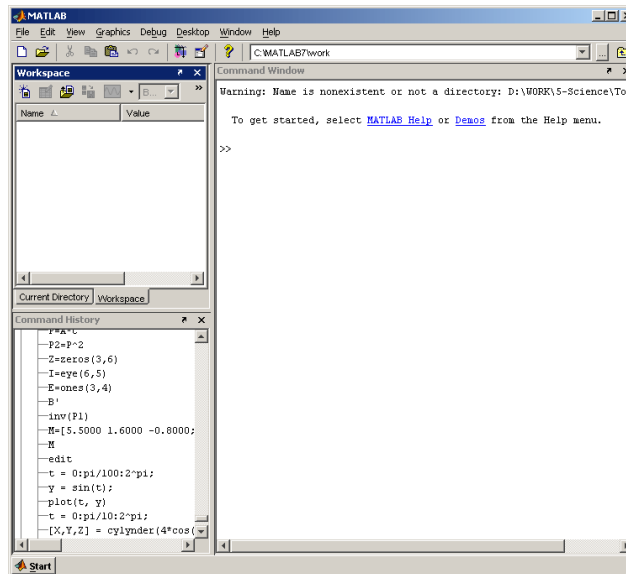


Рисунок 1.3 – Приклад програми “MatLab”

Програма “MatLab” має велику кількість переваг, серед яких:

- точність. Саме с великим пакет при інсталяції налагоджується точність в розрахунках;
- успішність. Після повного програмування розробник має більш успішну виборку на виході;
- забезпечує найшвидший функціонал для математичного обчислення матриць та лінійної алгебри;
- містить найкращі бібліотеки математичних пакетів для забезпечення підтримки всіх областей математики, починаючи від простого підсумовування до інверсії матриці;

- універсальність. “MatLab” широко використовується при проведенні обчислень ний в будь-якій області науки і техніки, в тому числі і економіці, фінансах;
- прозорість. Програмний код всіх функцій є відкритим для користувача. З ним можна ознайомитися і, при необхідності, внести в нього корективи.

На думку розробників програмний додаток “MatLab” має один недолік – обмежені можливості символьних обчислень. Дороговизна система та її пакетів розширень. Якщо перед розробником постає велика задача, то середовище “MatLab” може запропонувати придбати більший пакет з формулами та розрахунками для вирішення поставленої задачі. Характеристики програми “MatLab” можна побачити на таблиці 1.1.

Таблиця 1.3 – Основні характеристики програми “MatLab”

Вартість	Безкоштовна версія	Пробний період	
Операційні системи	Cloud, SaaS, Web	Mac	Windows (Android,IOS)
Навчання	Документація	Online	Персональні вебінари
Підтримка	Online	24/7	24/7
Останнє оновлення	20.03.2019 Робота з Windows XP та більш новими версіями	20.03.2019 Робота з Windows XP та більш новими версіями	20.03.2019 Робота з Windows XP та більш новими версіями

На таблиці 1.1 дуже добре показані основні характеристики програми “MatLab”, можна зрозуміти, що програмне середовище повністю сумісне для роботи в операційних системах Windows, MacOS та Linux. Останній реліз Matlab відбувся 20 березня 2019 року. Matlab - це ліцензоване програмне забезпечення, проте воно забезпечує пробний період 30 днів безкоштовно.

1.2.4 Програма Souver

Souver - це розумний блокнот з вбудованим калькулятором. Він дає миттєві відповіді на будь-які обчислення, які він знаходить у тексті. Це кращий спосіб для роботи, ніж з традиційним калькулятором, і більш легкий інструмент для швидких обчислень, ніж електронна таблиця. Souver можна встановити з набором зручних функцій для виконання багатокрокових обчислень. Наприклад, розроник може посилатися на попередні рядки, на зразок електронної таблиці. І, як і мова програмування, розробник може призначати імена числах і використовувати їх в якості змінних в своїх обчисленнях [7].

Підсумок внизу автоматично складає весь рядок, і фахівець може скласти проміжні підсумки по конкретним лініях. На рисунку 1.4 можна побачити інтерфейс програми Souver.

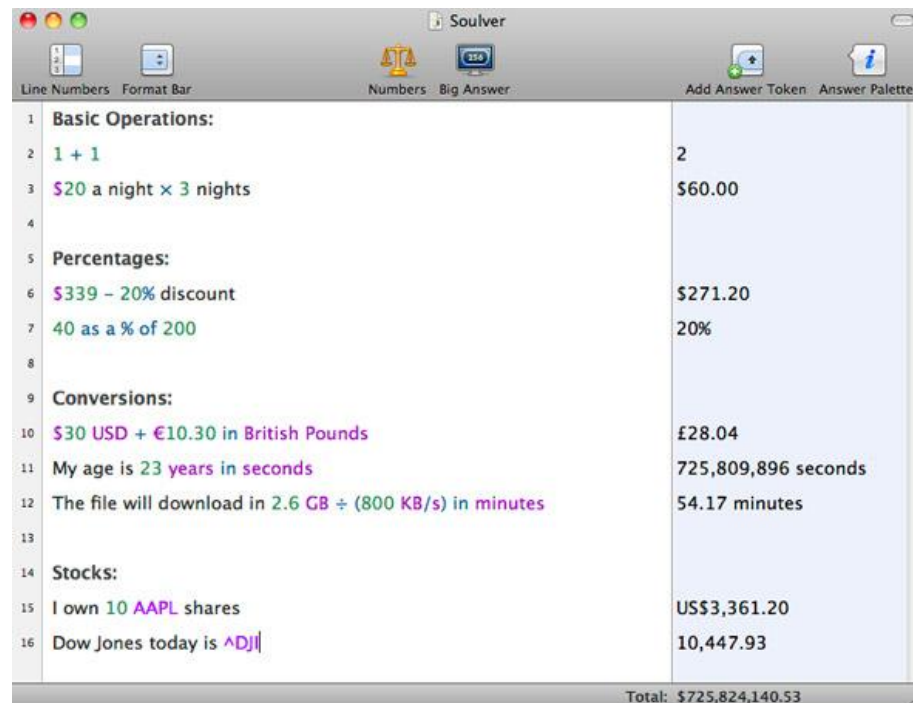


Рисунок 1.4 – Інтерфейс програми Souver

Програма Soulver дуже схожа на програму fxSoulver з перевагами, однак Soulver має недолік – нездатність до візуалізації відповіді на виході. Саме на виході розробник має тільки цифри, здатність програми до графіків та візуалізації немає. Основні характеристики однакові з характеристиками fxSolver, які описані в таблиці 1.3 та 1.4.

Таблиця 1.4 – Основні характеристики програми Soulver

Операційні системи	Android	Mac	Windows
Навчання	Документація та Google підтримка	Online	Документація та Google підтримка
Підтримка	Online	Online	24/7
Останнє оновлення	25.08.2020	Повна робота до версії 3.3	26.05.2021 Windows 10 та більш старими версіями

1.2.5 Програма R

R - це вільне програмне середовище для статистичних обчислень і графіків.

Це проект GNU, схожий на мову і середу S, який був розроблений в Bell Laboratories (раніше AT & T, зараз Lucent Technologies) Джоном Чамберсом і його колегами. R може розглядатися як інша реалізація S. Є деякі важливі відмінності, але більша частина коду, написаного для S, працює без змін під R. Найкраще підходить для користувачів, яким потрібен максимально потужний інструмент [8].

Основні характеристики можна побачити в таблиці 1.5.

Таблиця 1.5 – Характеристики до R – програми

Операційні системи	Windows
Навчання	Документація та Google підтримка Online
Підтримка	Сама програма все застаріла, підтримку можна знайти тільки в відеоуроках та в документаціях до самого середовища
Останнє оновлення	Повна робота до версії 3.3 26.05.2021 Windows 98 та більш нові версії

1.2.6 Програма S. Відмінність між R та S середовищами

S - мова програмування, розроблена фірмою AT & T Bell Labs (з'явився в середині 80-х років), призначена для обробки даних. Розроблено кілька версій розширення мови S - S-Plus, для різних платформ (UNIX, SunOS, Windows). Цей продукт містить об'єктно-орієнтовану мову програмування, що забезпечує можливість контролювати стан описаних на цій мові об'єктів в будь-якій точці алгоритму. Причому об'єктами є всі набори даних, функції і моделі аналізу. Це вигідно відрізняє даний продукт від спеціалізованих програмних пакетів. Переваги нового підходу полягають в забезпеченні контрольованості на кожному кроці обчислень, зрозумілості операцій і впевненості в результаті.

S - це мова/середовище програмування для статистики та статистичної графіки. S-Plus і R є реалізаціями S. Це означає, що обидва продукти побудовані на одній основі та обидва виконують однакові функції. Найпоширеніші статистичні функції, запрограмовані для S, працюватимуть під R і навпаки. Однак є ряд відмінностей, про які слід знати. На додаток до введення функцій тощо, S-Plus пропонує високорозвинений графічний інтерфейс користувача (GUI), тобто виглядає дуже сильно, ніж R. R має

мінімальний графічний інтерфейс користувача. (Однак існує низка подій, щоб додати до нього графічний інтерфейс) [9].

Стандартні файли даних для S та R несумісні (проте існують способи легко передати дані).

Графічний вихід досить часто відрізняється, оскільки у продуктах використовуються різні графічні механізми та інша модель програмування, особливо коли мова йде про програмування складної графіки та/та включає взаємодію. R

Інтерфейси іншого програмного забезпечення, особливо баз даних, абсолютно різні.

Висновок: більшість функцій, які не використовують графіку або базову графіку, без проблем працюють в обох середовищах. Однак потрібно очікувати проблем при використанні більш складних графічних інструментів. Оскільки цей клас стосується переважно візуальних інструментів, буде багато відмінностей, не стільки в тому, які інструменти доступні, але функції не будуть сумісними, мають різні назви та різні реалізації. Як правило, ці сторінки зосереджені на R; відзначаються важливі відмінності з S, і деякі документи будуть специфічними для S [9].

1.2.7 Порівняння систем прогнозування на основі основних характеристик

Спочатку перед розробником постає задача у виборі системи прогнозування, яка найкраще підходить для реалізації поставленої задачі. Процес прогнозування – дуже складний та має багато нюансів, які виникають під час розробки та у момент реалізації в самій системі.

Основні характеристики, які допоможуть розробнику вибрати систему прогнозування та працювати в ній для реалізації поставленої задачі на всіх етапах фінансування для дітей – сиріт:

- розробка та допрацювання власного коду;

- робота з готовими алгоритмами;
- візуалізація на виході.

Найкраще за все підходить саме “MatLab”, бо саме в цьому середовищі можна більш детально зробити аналіз та створити прогнозування. Саме мова “MatLab” має велику кількість характеристик, які допоможуть при розробці алгоритму, а саме:

- створення програмного коду;
- додавання всіх готових алгоритмів до власного коду;
- створення прогнозування та візуалізація даних на виході.

Звісно, що був аналіз інших систем прогнозування, але R та S – не підходить для вирішення поставленої задачі, бо розробник не має змогу подивитись результат роботи на рисунку в програмі, ці два середовища – дуже схожі та мають однакові характеристики. fxSolver та Souolver – також дуже схожі з собою, вони теж мають спільні між собою характеристики та добре підходять для реалізації задачі, але на основному етапі перед розробником може стати проблема в детальності розрахунків. Більш детально в цих системах розробник не може побачити або щось виправити. На основі цих характеристик можна розробити таблицю 1.6 для більш детального огляду всіх існуючих систем прогнозування.

Таблиця 1.6 – Основні характеристики систем прогнозування

Характеристика	Власна розробка	Додавання алгоритму	Безкоштовність	Дата останнього оновлення
“MatLab”	Можна додавати свій код та алгоритми	Повністю задовольняє потреби розробника	Має безкоштовний період 30 днів	20.03.2019
GNU Octave	Найкраще працює разом з “MatLab”	Можливість проведення чисельних експериментів	Повністю безкоштовний продукт	11.07.2021
Solver	Працює як багатофункціональний калькулятор	Додавання легкого алгоритму	Повністю безкоштовний продукт	13.09.2021
fxSolver	Математичний калькулятор	На виході маємо лиш відповідь	Повністю безкоштовний продукт	23.01.2020
R та S	Здатність до власної розробки	Працює лише в рамках роботи з GNU	Повністю безкоштовний продукт	10.08.2021

1.3 Висновок з поставленою задачею та з системами прогнозування

Проблема фінансування дітей – сиріт присвячено багато досліджень як теоретичного, так і прикладного характеру, які навчаються та вступають до лав ЗВО. Зі зростанням дітей – сиріт на території України, фінансування виходить на новий рівень. Кожен рік велика кількість абітурієнтів дітей – сиріт навчається та вступає до лав ЗВО. Перед розробником постає задача

шляхом систем прогнозування та методів статистики розробити модифікований метод, який дасть змогу швидко знаходити правильну кількість дітей – сиріт та СС які навчаються та вступають до ЗВО.

1.4 Методи прогнозування

1.4.1 Методи статистики для вирішення задачі

Сучасна статистична наука є складною, багатогалузевою системою наукових дисциплін. Основні складові частини статистики:

- теорія статистики;
- економічна статистика;
- галузеві статистики;
- соціальна статистика.

Саме для рішення проблем з фінансуванням для дітей – сиріт нам може допомогти соціальна та економічна статистики, бо саме в них можна знайти рішення даної проблеми [10].

Предметом статистики є вивчення розмірів і кількісних співвідношень між якісно визначеними масовими явищами, а також закономірностей їх формування, розвитку, взаємозв'язку у конкретних умовах місця та часу.

Завдання статистичного дослідження полягає в узагальненні даних і у виявленні закономірностей явищ в конкретних умовах місця та часу, які проявляють себе лише у великій масі явищ через подолання властивої одиничним елементам випадковості [10].

Головна функція статистики – постачання інформації та передача її в готовий програмний продукт [10].

1.4.2 Метод “дерева”. Структура та переваги перед іншими методами

Метод “дерева рішень” є одним з найбільш ефективних інструментів

інтелектуального аналізу даних і самий корінь аналітики, які дозволяє вирішувати завдання класифікації і регресії.

Вони представляють собою ієрархічні деревоподібні структури, що складаються з вирішальних правил виду «Якщо ..., то ...». Правила автоматично генеруються в процесі навчання на навчальній множині і, оскільки вони формулюються практично на природній мові дерева рішень як аналітичні моделі більш вербалізуємі і інтерпретовані, ніж нейронні мережі.

Оскільки правила в деревах рішень виходять шляхом узагальнення безлічі окремих спостережень, що описують предметну область, то за аналогією з відповідним методом логічного висновку їх називають індуктивними правилами, а сам процес навчання - індукцією дерев рішень.

Структура “дерева рішень” - це метод уявлення вирішальних правил в ієрархічній структурі, що складається з елементів двох типів - вузлів (node) і листя (leaf). У вузлах знаходяться вирішальні правила і проводиться перевірка відповідності прикладів цього правила з якого-небудь атрибуту навчальної множини [11].

У найпростішому випадку, в результаті перевірки, безліч прикладів, які потрапили в вузол, розбивається на дві підмножини, в одне з яких потрапляють приклади, що задовольняють правилу, а в інше - що не задовольняють [11]. З рисунка 1.5 можна побачити приклад структури “Дерева рішень”.

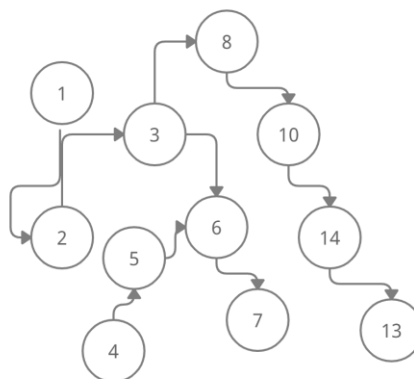


Рисунок 1.5 – Приклад структури “Дерева рішень”

При формуванні правила для розбиття в черговому вузлі дерева необхідно вибрати атрибут, за яким це буде зроблено. Загальне правило для цього можна розбити наступним чином: обраний атрибут повинен розбити безліч спостережень в вузлі так, щоб результуючі підмножини містили приклади з однаковими мітками класу, або були максимально наближені до цього, тобто кількість об'єктів з інших класів в кожному з цих множин було якомога менше. Метод “дерева рішень” заснований на поняттях теорії інформації (1.1).

$$H = -i = 1 \sum n N N i \log(N N i), \quad (1.1)$$

де n - число класів у вихідному підмножині;

Ni – число параметрів i – класу;

N – загальна кількість прикладів в множині.

Таким чином, метод може розглядатися як міра неоднорідності підмножини за представленими в ньому класів. Коли класи представлені в рівних частках і невизначеність класифікації найбільша, ентропія також максимальна. Якщо все приклади в вузлі відносяться до одного класу, $N = Ni$ логарифм від одиниці звертає середню швидкість генерування $N = Ni$ значень в нуль. $[i_1, i_2, \dots, i_n]$.

Таким чином, найкращим атрибутом розбиття A_j буде той, який забезпечить максимальне зниження ентропії результуючої підмножини щодо батьківського. Тоді найкращим атрибутом розбиття буде та, яка забезпечить максимальний приріст (1.2) інформації результуючого вузла щодо вихідного:

$$Gain(A) = Info(S) - (InfoSA), \quad (1.2)$$

де $Info(S)$ – інформація, яка зв'язана с підмножиною S до розбиття;

InfoSA - інформація, пов'язана з підмножиною, отриманими при розбитті по атрибуту *A*.

Розглянувши основні проблеми, що виникають при побудові методу дерев, було б несправедливо не згадати про їх переваги:

- швидкий процес навчання;
- генерація правил в областях, де експерту важко формалізувати свої знання;
- витяг правил на природній мові;
- інтуїтивно зрозуміла класифікаційна модель;
- висока точність передбачення, порівнянна з іншими методами аналізу даних (статистика, нейронні мережі).

В силу цих та багатьох інших причин, дерева рішень є важливим інструментом в роботі кожного фахівця та розробника, що займається аналізом даних.

Модулі для побудови і дослідження дерев рішень входять до складу більшості аналітичних платформ. Вони є зручним інструментом в системах підтримки прийняття рішень та інтелектуального аналізу даних:

- банківська справа. Оцінка кредитоспроможності клієнтів банку при видачі кредитів;
- промисловість. Контроль за якістю продукції (виявлення дефектів), випробування без руйнувань (наприклад, перевірка якості зварювання)
- медицина. Діагностика захворювань;
- система аналізу. Здатність до швидкого аналізу та вирішення конкретної задачі та постанови.
- торгівля. Класифікація клієнтів і товарів.

Слід визначити головний недолік методу “дерева рішень” – це обмежене число варіантів вирішення проблеми. У процесі побудови методу необхідно звернути увагу на його розмір. Воно не повино бути занадто

перевантаженим – це зменшує здатність до узагальнення і здатність давати вірні відповіді.

1.4.3 Штучні нейронні мережі. Структура та переваги перед іншими методами

Штучні нейронні мережі є математичними моделями біологічної структури мозку та його функцій сприйняття, обробки, зберігання та продукування інформації. Головною особливістю біологічних НМ є здатність до навчання та самоорганізації на основі набутого досвіду. Нейронні мережі мозку побудовані з великої кількості з'єднаних між собою клітин – нейронів. Моделі нейромереж організовані за схожим принципом, але мають ряд обмежень. Штучні нейронні мережі є груповим об'єднанням штучних нейронів. Дані об'єднання утворюють прошарки, які також з'єднуються між собою. Структура типової нейронної мережі показана на рис. 1.6. Як правило, кожна нейронна мережа складається мінімум з трьох базових прошарків: вхідного, прихованого та вихідного. Відомі також архітектури, що містять меншу кількість прошарків, або лише один обчислювальний елемент. В деяких джерелах вхідний та вихідний прошарки мережі не враховуються, тому зазначається, що мережа містить один прошарок. На даний час порядок нумерації не регламентований. Вхідний прошарок мережі призначений для отримання даних з датчиків або зовнішніх файлів. Вихідний прошарок передає відповідь мережі до інших блоків системи або програми. Всі основні обчислення відбуваються у прихованих прошарках, які містять набори з певної кількості нейронів. Входи прихованих нейронів приймають дані з попереднього прошарку, виходи – передають обчислене значення нейрона на наступний прошарок [12].

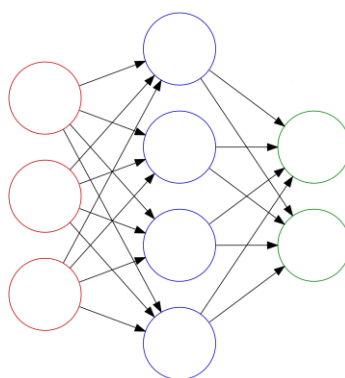


Рисунок 1.6 – Схема нейронної мережі

Фундаментальною особливістю біологічного мозку є здатність навчатись. У випадку штучних НМ навчання розглядається як процес визначення синаптичних ваг, що пов'язують нейрони між собою. Спосіб навчання залежить від конкретної задачі. НМ навчається за допомогою спеціальної множини навчальних даних. Ця особливість робить НМ більш гнучкими та привабливими, ніж використання традиційних програмних алгоритмів. Конкретні особливості перетворень у НМ визначаються параметрами нейронів, архітектурою та структурою мережі, способами синхронізації та управління інформативними даними. Вибір оптимальної кількості прихованих прошарків, кількості нейронів та видів зв'язків між ними значно впливає на ефективність мережі. Для опису НМ часто застосовуються терміни, які можуть мати різне значення в різних джерелах. Такими поняттями можуть бути:

- структура нейромережі. Спосіб зв'язків нейронів у нейромережі;
- архітектура нейромережі. Структура нейромережі та типи нейронів;
- парадигма нейромережі. Спосіб навчання та використання; іноді вміщує і поняття архітектури.

За фактом, переважну більшість відомих нейронних мереж, що застосовуються на практиці, можна віднести до двох загальних класів за архітектурою:

- з послідовними односкерованими зв'язками (НМ прямого поширення);
- з рекурентними зв'язками (НМ зворотного поширення).

Мережі прямого поширення, в свою чергу, відносять до статичних, оскільки на вхід мережі подається незалежний від попереднього стану мережі вхідний вектор. Мережі зворотного поширення є динамічними, так як входи нейронів змінюються завдяки наявності зворотних зв'язків, що впливає на стан мережі. В залежності від типу, змінюється сфера застосування даної НМ. Вибір типу мережі є одним з основних завдань під час проектування інтелектуальних систем [13].

Основні переваги нейронних мереж полягають в здатності навчатися самостійно і автоматично на основі вибірок, бути продуктивними на зашумлених або нечітких даних, мати можливість паралельної реалізації і бути ефективними інструментами для обробки великих баз даних.

1.4.4 Багатошарова нейронна мережа

Найбільший розвиток на сьогоднішній день отримали багатошарові нейронні мережі. Така мережа побудована з кількох прошарків нейронів. У вхідному прошарку містяться нейрони, які не виконують жодних обчислень та призначені для прийому і передачі до мережі вхідного вектора $[i_1, i_2, \dots, i_n]$, який складається з n елементів (відліків сигналу).

В нейронах прихованих прошарків відбувається перетворення отриманих вхідних даних, за формулою (1.3):

$$A = X(j)W(i, j) + b(i) \quad (1.3)$$

Обчислене значення нейрона A пропускається через активаційну функцію. В результаті, можна записати в загальному вигляді рівняння нейрона зі зміщенням (1.4):

$$[o_1, o_2, \dots, o_n], X(j) = [x_1, x_2, \dots, x_j], \quad (1.4)$$

де $X(j) = [x_1, x_2, \dots, x_j]$ – вектор даних з виходу нейронів попереднього прошарку;

j – кількість нейронів у попередньому прошарку;

i – індекс поточного нейрона;

$b(i)$ – зміщення поточного нейрона.

Значення виходів нейронів прошарку передається нейронам наступного прошарку. Останній прошарок багатошарової мережі називається вихідним, значення його нейронів $[o_1, o_2, \dots, o_n]$ спередуються до наступних функціональних блоків нейромережевої системи. Наразі не існує чітких правил підбору числа прихованих прошарків та кількості нейронів в них. Вибір параметрів мережі відбувається у відповідності до загальних рекомендацій або емпірично. Після визначення архітектури та вибору активаційних функцій відбувається процес навчання нейронної мережі [13].

1.4.4 Принципи навчання нейронних мереж

Здатність до навчання за зразками, які формують навальну множину, є головною особливістю нейронних мереж. Процедuru навчання НМ можна розглядати як ітеративний процес корекції значень таким чином, щоб мережа виконувала необхідну задачу найбільш оптимально. Іноді до поняття навчання відносять і процес визначення архітектури мережі.

Загальний принцип навчання можна описати як послідовну подачу на вхід мережі векторів навчальних даних $X(i)$ з відомими правильними відповідями $Y(i)$, які називають цілями.

Існує дві основних парадигми навчання:

- контрольоване;
- неконтрольоване.

Контрольоване навчання. Переважна більшість практичних застосувань нейронних мереж побудована на використанні алгоритмів контрольованого навчання, або навчання «з вчителем». Такий алгоритм передбачає порівняння поточних значень на виході мережі з бажаними. На початку навчання синаптичні ваги ініціалізуються випадковими значеннями, а далі коригуються з метою досягнення найкращої відповідності між поточним та бажаним результатом на виході. Задовільна точність мережі досягається шляхом мінімізації похибок, які в наслідок проходження через мережу всіх навчальних зразків утворюють функцію помилки. Саме ця помилка розглядається як помилка мережі. Критеріїв зупинки навчання може бути декілька, але навчання зупиняється у випадку досягнення одного із них. В якості таких критеріїв можуть використовуватись кількість виконаних епох, досягнення бажаного значення помилки або припинення її зменшення. Зростання помилки на тестовій множині може свідчити про перенавчання.

Саме контрольоване навчання ідеально підходить для моніторингу та знаходження шляхів для вирішення поставленої задачі. Контрольоване

навчання дасть змогу знаходженні дітей – сиріт, які або вже вчать, або тільки збираються поступати в вищі учбові заклади на території України.

Неконтрольоване навчання. Навчання «без вчителя» вважається одним із найбільш перспективних напрямів розвитку нейромережових технологій. На сьогоднішній день самонавчання використовується здебільшого у вузькому колі задач. Неконтрольоване навчання передбачає відсутність зовнішніх втручань у процес корекції синаптичних ваг. Значення ваг коректується автоматично, ефективність мережі контролюється внутрішніми алгоритмами. Для адаптації ваг використовуються спеціальні навчальні функції. Інформація щодо архітектури і організації мережі та критерії оцінки її ефективності мають бути закладені в алгоритми роботи мережі і навчальні правила [14].

1.4.5 Вибірковий метод аналізу. Актуальність та головні задачі методу

Вибірковий метод - статистичний метод дослідження загальних властивостей сукупності будь-яких об'єктів на основі вивчення властивостей лише частини цих об'єктів. Сукупність досліджуваних об'єктів, що цікавлять дослідника - генеральна сукупність. А частина об'єктів, що підлягають вивченню, називають вибірковою сукупністю або вибіркою.

Необхідність вибіркового методу може бути викликана об'єктивними причинами:

- об'єкт дослідження дуже великий. Як велика українська база, звідки можна брати всі дані для рішення поставленої задачі;
- необхідність в зборі первинної інформації в перших дослідженнях.

Головне завдання вибіркового методу - з мінімальним обсягом вибірки отримати якомога більше точний опис цікавої генеральної сукупності на основі вибірових даних. Досягти бажаного результату можна

тільки на основі вибірки, яка об'єктивно відображає властивості генеральної сукупності.

Точність результатів вибірових обстежень досягається за рахунок використання складних методів формування вибірки (кластерного відбору, завдання розшарування, використання ймовірносно-пропорційного відбору, простого випадкового або випадкового відбору, повторного або неповторного відбору).

Вибірковий метод використовують, перш за все, в соціології, маркетингу, клінічних дослідженнях. Але фактично при статистичному аналізі даних в будь-якій області дослідник працює, як правило, не з генеральною сукупністю, а з вибіркою. Помилка багатьох розробників, що вони не надають цьому значення, не замислюються, якими методами була отримана анализируемая інформація і наскільки дотримана методологія вибірового обстеження. Через це одержувані результати не відповідають реально об'єктивно існуючим закономірностям, тому що аналізується нерепрезентативна вибірка, а в цьому і полягає недолік цього методу, що розробник на перших етапах може помилитись і йому далі буде більш складно знайти помилку та вирішити її [15].

1.4.6 Логістична регресія. Значення методу та головна роль

Логістична регресія або логіт-регресія - це статистична модель, використовувана для передбачення ймовірності виникнення даного події за допомогою логістичної функції.

Логістичну регресію відносять до моделей бінарного вибору. Регресійна модель бінарного вибору - це регресійна модель, в якій залежна змінна бінарна. Залежна змінна може приймати лише два значення і означати, приналежність до певної групи (дитина – сирота), що робляться дію (вступ до вищого навчального закладу), варіанти відповіді «так» або «ні» (вступ до ЗВО). Будувати звичайну лінійну регресійну модель з бінарними

залежними змінними не можна. У цьому випадку неможливо буде інтерпретувати передбачені по регресії в безперервної кількісної шкалою значення залежної змінної.

Значення факторів в моделях бінарного вибору повинні бути виміряні кількісної шкалою. Також в моделі бінарного вибору можна включати в якості факторів категоріальні змінні. Отже, в моделях бінарного вибору будується регресійна модель залежності ймовірності того, що результативна дихотомическая змінна прийме значення 0 або 1 при заданому значенні факторів.

В якості спеціальної функції в моделях бінарного вибору зазвичай використовують:

- логістичну функцію;
- функцію стандартного нормального розподілу.

Моделі бінарного вибору на основі логістичної функції називаються логістичною регресією. Моделі бінарного вибору на основі функції стандартного нормального розподілу називають пробітний-моделями.

За допомогою логістичної регресії прогнозується ймовірність відгуку для залежної змінної від включених в модель незалежних змінних. На основі прогнозних значень ймовірності можна зробити класифікацію всіх спостережень на дві групи. Окремим аналізом при побудові моделі логістичної регресії є аналіз ROC-кривих. ROC-аналіз дозволяє вибрати оптимальне значення порогового значення ймовірності для класифікації. ROC-крива - крива, яка використовується для представлення результатів бінарної класифікації та оцінки ефективності класифікації. У прикладному статистичному аналізі логістична регресія використовується для вирішення двох завдань: моделювання взаємозв'язку і класифікації спостережень. Логістичну регресію застосовують при проведенні клінічних досліджень в медицині, в банківському скоринг для побудови рейтингу позичальників і управління кредитними ризиками, в споживчому скоринг для моделювання поведінки покупців та інших сферах [16].

1.4.7 Порівняльна характеристика методів аналізу

При вирішенні поставленої задачі був проведений великий аналіз серед головних методів аналізу, встановлення головних рис, сутності серед основних методів аналізу. В таблиці 1.7 можна побачити основні методи аналізу статистики. При формуванні таблиці були розібрані основні характеристики для створення її. Також, був проведений аналіз існуючих середовищ програмування, де найкраще можна буде з'єднати метод аналізу з середовищем програмування, адже не всі методи аналізу добре працюють з обраними середовищами програмування. Найкраще з усіх можна з'єднати нейронну мережу, бо вона дуже добре працює з усіма системами прогнозування. Звісно, що будь – який метод має свої переваги та недоліки в порівнянні з іншими методами. Також, був проведений детальний огляд всіх бібліотек, які можуть допомогти при вирішенні поставленої задачі. Кількість бібліотек зростає щодня і навіть виник вже певний набір бібліотек, які стабільно встановлюються іншими розробниками, які займаються обробкою великих даних та на виході отримують результат. Також, ці створені бібліотеки дозволяють перетворити комп'ютер на готову станцію, де можна починати реалізувати будь – які алгоритми та аналіз великих даних.

Таблиця 1.7 – Основні методи аналізу статистики

Метод	Сутність	Основні етапи побудови	Переваги	Використання (галузь)
Дерева рішень	Модель прогнозує значення цільової змінної на основі декількох змінних на вході.	Вибрати атрибут для здійснення розбиття в даному вузлі; визначити критерій зупинки навчання; вибрати метод відсікання гілок; оцінити точність побудованого дерева.	Можливість пов'язати ставлення цілі з діями, що підлягають реалізації	В галузі статистики та аналізу даних для прогнозних моделей
Нейронна мережа	Працюють як з кількісними, так і з якісними характеристиками, певним чином підготовленими, дають змогу встановити зв'язки.	Дані поступають на нейрон; обчислюється їх вага; результати обчислень передаються на наступний нейрон; процес повторюється.	Універсальність, дешевизна і низька швидкість навчання	Економіка, медицина, наукові дослідження, інформаційні технології
Багатoshарова нейронна мережа	Дають змогу встановити зв'язки більш детально.	Задати навчальну вибірку; розрахувати помилку мережі E; Встановити: $s = s + 1$; розрахувати помилку	Навчання на основі даних середовища і підвищувати свою продуктивність.	Наукові дослідження, інформаційні технології
Вибірковий метод	З значенням характеристик вибіркової сукупності роблять певні висновки	Визначення основи вибірки; побудова дизайну вибірки; обчислення обсягу вибірки; відбір одиниць для безпосереднього обстеження.	Розповсюдити висновки отримані на основі вивчення частини вибірки	Соціальні дослідження, економіка
Логічна регресія	При запровадженні значення може знаходити застосування у класи	Логістичну регресію можна представити у вигляді одношарової нейронної мережі	Простота розрахунків і прийняття на точність прогнозу	Використовується як готові реалізації для вирішення прикладних задач

Для вирішення поставленої задачі, а саме для розробки методу для прогнозування фінансування найкраще підходять три основних методи аналізу статистики:

- дерева рішень;
- нейронна мережа;
- вибірковий метод.

Щоб обрати найкращий метод треба розглянути основні переваги та недоліки трьох методів та обрати найкращий, який допоможе при реалізації задачі. В таблиці 1.8 можна побачити основні методи аналізу статистики для поставленої задачі.

Таблиця 1.8 – Методи статистики при реалізації задачі

Метод	Час при навчанні	Залежність вхідних та вихідних даних	Точність прогнозу
Дерева рішень	Дуже швидкий процес навчання	Параметри дерева необхідно налаштовувати в залежності від вхідних даних	Має велику точність, але поступається нейронній мережі
Нейронна мережа	Залежить від процесу (з вчителем, або без вчителя)	Здатна вирішувати завдання, в яких невідомі закономірності розвитку ситуації та залежності між вхідними та вихідними даними	Дуже велика точність прогнозу завдяки адаптації до середовища
Вибірковий метод	Значно програє дереву рішень та нейронній мережі	Спостереження не всіх елементів сукупності що вивчаються, а лише відібраних	Залежить від виборки

Проаналізувавши методи інтелектуального аналізу даних можна відзначити такі переваги нейронних мереж перед традиційними обчислювальними системами:

- нейронна мережа здатна вирішувати завдання, в яких невідомі закономірності розвитку ситуації та залежності між вхідними та вихідними даними;

- можливість роботи за наявності великої кількості неінформативних, шумових вхідних сигналів;
- нейронна мережа здатна адаптуватися до змін навколишнього середовища;
- нейронні мережі мають потенційну надвисоку швидкість за рахунок використання масового паралелізму обробки інформації.

Переваги використання дерев рішень:

- інтуїтивно зрозуміла класифікаційна модель;
- висока точність прогнозу, порівнянна з іншими методами, але все одно поступається в точності з використанням нейронних мереж;
- побудова непараметричних моделей. З цих і багатьох інших причин, методологія дерев рішень є важливим інструментом у роботі фахівця-аналітика.

В результаті дослідження було встановлено, що для вирішення поставленої задачі найкраще підходить алгоритм нейронних мереж.

1.5 Постановка задачі

Проблема фінансування дітей – сиріт – це велика проблема на сьогоднішній день на всій території України. Кожен рік абітурієнти подають свої документи до ЗВО та успішно проходять вступну компанію. Звісно, що серед усіх абітурієнтів – є чимала кількість дітей – сиріт, які теж подали документи до ЗВО. Шляхом вирішення актуальної задачі перед розробником постає формула (1.4).

З формули 1.5 можна побачити відбір основних рис для реалізації задачі.

$$F = (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}), \quad (1.5)$$

де F - фінансування для дітей – сиріт, які вступають та навчаються в ЗВО;

x_1 - всі абітурієнти, які поступили в учбовому році;

x_2 - похибка, яка може бути кожний рік вступивших до лав ЗВО;

x_3 - похибка (інфляція);

x_4 - субвенція (харчування, проживання, одяг, проїзд) для дітей – сиріт;

x_5 - оплата роботи викладачів ЗВО;

x_6 - середній відсоток дітей – сиріт, які вступили до ЗВО на перший рік навчання;

x_7 - похибка щодо дітей – сиріт, які закінчили своє навчання;

x_8 - стипендія для дітей, які вступили та навчаються на лавах ЗВО;

x_9 - вік;

x_{10} - область навчання.

З формули (1.5) можна зрозуміти, що шлях до реалізації задачі лежить в даних, які вже існують.

1.6 Висновки за розділом 1

В розділі розглянуто проблематику на основні методи вирішення проблеми фінансування, розглянуто основні методи дослідження та вибрано найкращий, який може допомогти реалізувати задачу. Також, в ході прогнозування було розглянуто методи статистики, за допомогою яких, можна реалізувати метод для вирішення проблеми.

2 ВИБІР ІНСТРУМЕНТІВ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Вибір мови програмування

Оскільки нейронної мережі можна реалізувати практично будь-якою мовою програмування, то питання її вибору є досить важливим. Від обраної мови буде залежати й правильність та легкість опрацювання даних, швидкість роботи моделі, підтримка алгоритмів тощо. Найчастіше для розробки нейронних мереж використовують мови C++, C#, Python та Matlab. Зараз, перед розробником постає питання з приводу мови програмування для реалізації задачі. Треба розглянути основні мови програмування та з'ясувати, яка мова найкраще підходить для неронній мережі [15].

2.1.1 Мова програмування C++. Відмінність між мовою C

C++ - універсальна мова програмування високого рівня з підтримкою декількох парадигм програмування. Зокрема: об'єктно-орієнтованої та процедурної. Розроблена Б'ярном Страуструпом у 1979 році та названа «C з класами». Страуструп перейменував мову у C++ у 1983 році. Базується на мові C.

Мова C++ була розроблена з урахуванням продуктивності, ефективності і гнучкості, що робить його ідеальним вибором для багатьох проєктів програмування штучного інтелекту, які вимагають швидкості. У порівнянні з іншими мовами програмування, C++ має більш швидке виконання і більш низьку затримку, що робить його корисним для пошуку рішень складних проблем штучного інтелекту [16].

При створенні C++ прагнули зберегти сумісність з мовою C. Більшість програм на C справно працюватимуть і з компілятором C++. C++ має синтаксис, заснований на синтаксисі C. Стандартні бібліотеки C++

включають стандартну бібліотеку Cі з невеликими змінами, які роблять її більш відповідною для мови C++ [16].

Нововведеннями C++ порівняно з C є:

- підтримка об'єктно - орієнтованого програмування через класи;
- підтримка узагальненого програмування через шаблони;
- доповнення до стандартної бібліотеки;
- додаткові типи даних;
- обробка винятків;
- простори імен;
- вбудовані функції;
- перевантаження операторів;
- перевантаження імен функцій;
- посилення і оператори управління вільно розподіленою пам'яттю.

Великою перевагою мови можна віднести можливість роботи з пам'яттю та ядрами комп'ютера, а також можливість програм, які написані даною мовою запускатися на всіх операційних системах, що робить цю мову дуже потужною в процесі роботи з програмістом. Однак, мова має значний недолік – її складність роботи та важкість навчання. Для розробки будь-якого застосування необхідно буде витратити багато часу, для розуміння роботи з пам'яттю комп'ютера та підводних каменів, що можуть зустрітися під час роботи та розробки [16].

2.1.2 Мова програмування C#. Порівняння з мовою C++

Мова C# була розроблена компанією Microsoft, один з найпопулярніших сучасних мов програмування. Він затребуваний на ринку розробки в різних країнах, C# застосовують при роботі з програмами для PC, створення складних веб-сервісів або мобільних додатків. З'явилася мова для власних потреб платформи Microsoft .NET, поступово ця мова стала дуже

популярною. Підтримка Microsoft приділяє значну увагу підтримці мови розробки, а тому регулярно з'являються оновлення та доповнення, виправляються виявлені баги в компіляторі, оновлюються старі бібліотеки. Розробники зацікавлені в популяризації інструменту і докладають до цього масу зусиль [17].

Інструментарій C# дозволяє вирішувати широке коло завдань, мова дійсно дуже потужний і універсальний. На ньому розробляють:

- додатки для WEB;
- різні ігрові програми;
- додатки платформ Android або iOS;
- програми для Windows.

Звісно, що мова програмування C# - це нова мова та має велику підтримку, ніж мова програмування C++. Тож, потрібно зрозуміти відмінності між цими двома великими мовами:

- покажчики і управління пам'яттю. У мові C++ робота з покажчиками займає одне з центральних місць. Нормальний стиль програмування на C# передбачає написання безпечного коду, а це значить — ніяких покажчиків, ніякої адресної арифметики, ніякого управління розподілом пам'яті. Можливість роботи з покажчиками в душі C++ обмежена небезпечними блоками;

- спадкування і шаблони. В мові C# не реалізовані такі важливі для C++ моменти, як множинне спадкування і шаблони. Множинне спадкування в C# можливо тільки для інтерфейсів;

- типи даних. В мові C# з'явилася принципово нова класифікація типів, підрозділяються на типи значущі і смітники. Як наслідок, застосовуються різні способи роботи з об'єктами цих типів. У мові усунена різниця між змінними і об'єктами. Всі змінні в C# — теж об'єкти, які мають єдиного предка — клас System.Object;

- структури. У мові C++ структури подібні класу, за невеликими винятками. В C# різниця між структурою і класом більш істотна: структури

не можуть мати спадкоємців, класи відносяться до посилальних типів, а структури — до значущим;

— масиви та класи. В мові C# є можливість як оголошувати класичні масиви, так і працювати з масивами за допомогою вбудованих класів. Робота з масивами в C# більш безпечна, оскільки вихід за межі масиву контролюється (за умови використання безпечного коду). А класи мають розходження в підходах до приховування властивостей класу. В C++ таке поняття, як властивість, взагалі відсутнє. В C# введені процедури-властивості `get` і `set`. Синтаксис звернення до властивостей класу в C# аналогічний синтаксису звертання до даних [18].

2.1.3 Мова програмування Python. Переваги та недоліки в порівнянні з іншими мовами

Python – багатоцільова мова програмування, яка дозволяє писати код, що добре читається. Відносний лаконізм мови Python дозволяє створити програму, яка буде набагато коротше свого аналога, написаного на іншій мові. Python - багатоплатформова мова програмування. Це означає, що програми на Python можна запускати в різних операційних системах без будь-яких змін. Ще однією перевагою Python є його стандартна бібліотека, яка встановлюється разом з Python і містить готові інструменти для роботи з операційною системою, веб-сторінками, базами даних, різними форматами даних, для побудови графічного інтерфейсу програм тощо. Програми, написані на мові програмування Python, можуть бути як невеликими скриптами, так і складними системами. Мова програмування абсолютно безкоштовна.

Python використовується для різних цілей: для створення ігор і веб-застосунків, розробки внутрішніх інструментів для різноманітних проектів. Мова також широко застосовується в науковій області для досліджень і розв'язування прикладних завдань [19].

Застосування цієї мови програмування дуже велике, бо на мові Python написано велика кількість сайтів, пошукових додатків, серед яких є:

- BitTorrent – протокол для обміну даними;
- Ubuntu Software Center – вільне програмне забезпечення для пошуку, установки і видалення пакунків в системі Ubuntu Linux;
- Blender – програма для створення тривимірної комп'ютерної графіки, що включає засоби моделювання, анімації, вимальовування, пост-обробки відео, а також створення відеоігор;
- GIMP – растровий графічний редактор, із підтримкою векторної графіки;
- кросплатформна онлайн гра World of Tanks;
- вільна енциклопедія Вікіпедія;
- пошукова система Google;
- DropBox – файловий хостинг, що включає персональне хмарне сховище, синхронізацію файлів і програму – клієнт;
- YouTube – популярне відеосховище, де користувачі мають змогу завантажувати та дивитись різноманітні відео в будь – якому якості.

Але, мова програмування Python звісно має декілька недоліків в порівнянні з іншими мовами. Один з можливих недоліків Python - швидкість виконання коду. Python не є компільованою мовою. Код на Python спочатку компілюється у внутрішній байт-код, який потім виконується інтерпретатором Python. У більшості випадків при використанні Python виходять програми повільніші в порівнянні з такими мовами, як С. Втім, сучасні комп'ютери мають таку обчислювальну потужність, що для більшості застосунків швидкість розробки важливіша швидкості виконання, а програми на Python зазвичай пишуться набагато швидше. Окрім того, Python легко розширюється модулями, написаними на С або С++. Такі модулі можуть використовуватися для виконання частин програми, що створюють інтенсивне навантаження на процесор [19].

2.1.4 Мова програмування MatLab

Мова програмування MatLab є вискорівневою мовою програмування, що включає широкий спектр функцій, інтегроване середовище розробки, засновані на матричних структури даних, об'єктно-орієнтовані можливості написаних на інших мовах програмування. Пакет MatLab був створений компанією Math Works більше десяти років тому. Робота сотень вчених і програмістів спрямована на постійне розширення його можливостей і вдосконалення закладених алгоритмів. В даний час MatLab є потужним і універсальним засобом вирішення задач, що виникають в різних областях людської діяльності. MatLab є патентованою мовою програмування та одною з найстаріших, ретельно опрацьованих і перевірених часом систем автоматизації математичних розрахунків, побудована на розширеному поданні та застосуванні матричних операцій. Незважаючи на те, що спочатку MatLab призначався виключно для обчислень, в процесі еволюції в системі з'явилися широкі можливості для програмування. Але мова "MatLab" має декілька недоліків, а саме, системи MatLab можна відзначити невисоку інтегрованість середовища, не дуже виразну довідкову систему і специфічний редактор коду MatLab-програм. Сьогодні система широко використовується в техніці, науці та освіті для аналізу даних і організації обчислень. Корінь "MatLab" містить великий набір бібліотек, які реалізують сучасні чисельні методи моделювання, створені за останні декілька десятків років. MatLab може бути застосований для розрахунків в будь-якій області науки та техніки. Також система містить потужні засоби графічної візуалізації. Серед недоліків варто відзначити низьку швидкість компіляції та застарілість її [20].

Останні бібліотеки оновлювались дуже давно с останнім релізом, який був ще в 2017 році.

Однією із основних задач системи Matlab завжди було надання користувачам потужної мови програмування, орієнтованої на технічні і

математичні розрахунки і здатної перевершити можливості традиційних мов програмування, котрі багато років використовувались для реалізації чисельних методів. При цьому особлива увага приділялась як підвищенню швидкості обчислень, так і адаптації системи до розв'язку самих різноманітних задач користувачів [21].

2.2 Порівняння мов програмування

Згідно з вибором середовища програмуванням перед розробником постала задача вибору мови програмування. Після детального аналізу стало зрозуміло, що найкраще “MatLab” працює з C++, C#, Python та з встроєною мовою MatLab. Для найкращої реалізації задачі фінансування розробником була створена таблиця 2.1 – порівняння мов програмування з середовищем “MatLab”. Перед вибором середовища та мови програмування розробник повинен розглянути основні мови програмування для реалізації задачі. Тож, серед оптимальних та ефективних мов було обрано чотири основні та популярні мови серед розробників в усіх галузях, від машиннобудівної мови до розробки додатків до будь – якого гаджету.

Таблиця 2.1 – Порівняння мов програмування

Критерії порівняння	Мова програмування			
	C++	C#	Python	Matlab
Адаптація до РС та його середовища	+	+	+	-
Оновлення пакетів	-	+	+	-
Безкоштовність	+	+	+	-
Платформонезалежність	+	+	+	+
Наявність бібліотек для навчання	+	+	+	+
Продуктивність в процесі	+	-	+	-
Кросплатформність	-	+	+	-
Швидкість компілювання коду	+	+	+	-
Робота з ядром комп'ютера	+	-	+	+
Співпраця з іншими мовами	+	+	+	+

Після аналізу основних характеристик та критерій було зроблено декілька висновків. Згідно з таблицею 2.1 – найкраще для реалізації завдання підходить мова програмування C++. Хоча, C++ має великий недолік серед інших мов – це важкість в навчанні, все одно, для технічної реалізації в програмному середовищі “MatLab” – найкраще підходить при розробці мова програмування C++, але попередній вибір був також C#, бо ця мова програмування має схожі параметри як і мова C++, але мова C# - нова мова та зроблена більш для створення програмних додатків, кросплатформні

додатки та розробка WEB – застосунків. Тож, остаточний вибір пав на мову програмування C++.

2.3 Головні бібліотеки для реалізації задач в C++

На початку кожної програми необхідно підключити бібліотеки, які потрібні для реалізації того чи іншого дії, і простір імен.

Кожна бібліотека має своє призначення:

- `<Iostream>` - бібліотека, в якій містяться потоки введення і виведення `cin` і `cout`;
- `<Iomanip>` - бібліотека, в якій міститься модифікатор `setw`. З його допомогою можна, наприклад, встановити ширину поля для кожної координати вектора, або елемента матриці;
- `<Cmath>` - в цій бібліотеці містяться тригонометричні функції, логарифми, модулі та інші функції, призначені для виконання найпростіших математичних операцій;
- `<Cstddef>` - бібліотека, в якій міститься покажчик `NULL`.

В мові програмування C++ дуже добре реалізовані функції, в які можна вставити будь – який алгоритм для реалізації поставленої задачі. Мову C++ строго можна віднести до машинної мови, тобто, шлях до реалізації задачі матиме напрям сурово по заданій траєкторії. Для реалізації всієї етапів фінансування найкраще підходить бібліотека `<Cmath>`, бо в ній дуже добре можна підставити потрібний алгоритм, функцію та з'вязати все це і на виході отримати бажаний результат. Звісно, що без бібліотеки `<iostream>` жодна програма не буде працювати [22].

2.3.1 Ознайомлення с бібліотекою `<Cmath>`. Детальний опис функцій

В C ++ визначені в заголовки `<cmath>` функції виконують деякі часто використовувані математичні завдання. Наприклад, знаходження кореня,

піднесення до степеня, $\sin()$, $\cos()$ і багато інших. У таблиці 2.2 показані основні математичні функції, прототипи яких містяться в заголовки `<cmath>`.

Таблиця 2.2 – Основні функції в бібліотеці `<Cmath>`

Функція	Опис роботи функції
<code>double sqrt(double x);</code>	Обчислює і повертає корінь з позитивного числа, приймаючи його як в аргумент x
<code>double pow(double x, double y);</code>	Зводить число, яке приймається в якості аргументу x , в ступінь, прийняту в якості аргументу y .
<code>double fabs(double x);</code>	Обчислює абсолютне значення числа x
<code>double fmod(double x, double y);</code>	Обчислює залишок від ділення x на y .
<code>double ceil(double x);</code>	Обчислює найменше ціле, значення якого не буде меншим, ніж x .
<code>double floor (double x);</code>	На відміну від попередньої функції ця функція обчислює найбільше ціле, по значенням не перевершує x .
<code>double modf (double value, double ptr);</code>	Розбиває значення аргументу <code>value</code> на цілу і дробові частини. Цілу частину функція зберігає в об'єкті на який вказує показник <code>* ptr</code> , а дробову повертає.
<code>double cos (double x);</code>	Обчислює косинус аргументу x , який задається в радіанах
<code>double cos (double x);</code> <code>double sin (double x);</code> <code>double tan (double x);</code>	Обчислює косинус, синус та тангенс аргументу x , який задається в радіанах
<code>double acos (double x);</code>	Обчислює головне значення арккосинуса x . Аргумент x повинен бути з інтервала $[-1;1]$

На основному етапі розробки саме всі ці функції знадобляться програмісту для вирішення поставленої задачі.

2.3.2 Інструментарій для створення нейронних мереж

Для реалізації завдання та проблеми фінансування дітей – сиріт, які навчаються в ЗВО потрібна прив'язка програмного продукту с нейронними мережами. Мова програмування C++ дуже добре працює саме з простими та багат шаровими нейронними мережами, бо, як говорилося раніше, мова C++ - машинна мова, яка направляєтся строго по заданій траєкторії. Процес самонавчання нейронній мережі в C++ не відрізняється від мов програмування C# та Python.

Можливість розробки нейронній мережі в мові програмування C++ найкраще описана в інструментаріях Dlib, Caffe, TensorFlow.

Звісно, що тепер треба більш детально описати кожен алгоритм та знайти найкращій для реалізації заданої задачі [23].

2.3.3 Інструментарій Dlib

Dlib - це сучасний інструментарій C++, який містить алгоритми машинного навчання та інструменти для створення складного програмного забезпечення на C++ для вирішення проблем реального світу. Він використовується як у промисловості, так і в наукових колах у широкому діапазоні областей, включаючи робототехніку, вбудовані пристрої, мобільні телефони та великі високопродуктивні обчислювальні середовища. Ліцензування з відкритим кодом Dlib дозволяє безкоштовно використовувати його в будь - якій програмі.

Щоб слідкувати або брати участь у розробці Dlib, потрібно підписатись на Dlib на Github. Також, велика перевага Dlib в тому, що кожен розробник може завантажити свою роботу в інструментарії Dlib та ділитись зі всіма своєю роботою.

Документація Dlib – на відміну від багатьох проєктів з відкритим кодом, цей надає повну та точну документацію для кожного класу та функції. Існують також режими налагодження, які перевіряють задокументовані передумови функцій. Якщо це ввімкнено, він виявить переважну більшість помилок, викликаних неправильним викликом функцій або неправильним використанням об'єктів.

Портативний код високої якості має гарне охоплення одиничними тестами. Співвідношення одиничних тестових рядків коду до бібліотечних рядків коду становить приблизно 1 до 4.

Бібліотеку регулярно тестують у системах MS Windows, Linux та Mac OS X. Однак Dlib повинен працювати на будь - якій системі POSIX і використовувався на Solaris, HP/UX та BSD.

Для використання бібліотеки не потрібні інші пакети. Потрібні лише API, які надаються стандартною ОС.

Перед використанням бібліотеки не потрібно встановлювати або налаштовувати крок. Весь код, специфічний для операційної системи, ізольований всередині шарів абстракції ОС, які зберігаються якомога меншими. Інша частина бібліотеки розташовується поверх шарів абстракції ОС або є чистим стандартом ISO C++.

Переваги інструментарія:

- аналізатор командного рядка з можливістю аналізу та перевірки командних рядків з різними типами аргументів та параметрів;
- багато класів контейнерів;
- багато об'єктів менеджера пам'яті, які реалізують різні стратегії об'єднання пам'яті
- інструмент, який дозволяє легко викликати C++ з “MatLab”

Портативний код високої якості Портативний код високої якості [24].

2.3.4 Інструментарій Caffe

Caffe - це система глибокого навчання, створена з урахуванням виразності, швидкості та модульності. Він розроблений Berkeley AI Research (BAIR) та спільнотами спільноти. Caffe випускається за ліцензією BSD 2-Clause.

Виразна архітектура заохочує застосування та інновації. Моделі та оптимізація визначаються конфігурацією без жорсткого кодування. Можливість перемиканням між процесором та графічним процесором, встановивши єдиний прапор для навчання на машині з графічним процесором, а потім розгорнення його до кластерів товарів або мобільних пристроїв.

Розширений код сприяє активному розвитку. У перший рік роботи Caffe його розробили понад 1000 розробників, і було внесено багато істотних змін. Завдяки цим авторам, фреймворк відстежує найсучасніший рівень у коді та моделях. Як і інструментарій Dlib, Caffe має великий вплив серед розробників, бо кожен з професійних програмістів може оновити, змінити або розробити в цьому фреймворке щось нове, яке здатне зпростити подальшу роботу інших користувачів, або розробників. Caffe надає підтримку академічним науково-дослідним проектам, прототипам запуску та навіть масштабним промисловим додаткам у зорі, мовленні та мультимедії [25].

2.3.5 Платформа TensorFlow

TensorFlow - це комплексна платформа з відкритим вихідним кодом для машинного навчання. Він має все можливе для збільшення гнучкості екосистемних інструментів, бібліотеки та ресурсів повідомлень, що дозволяє дослідникам просувати нові досягнення у сферах машинного навчання,

розробників легко створювати та розширювати додатки на основі машинного навчання.

До переваг TensorFlow можна віднести:

- проста побудова моделі. Можна з легкістю створювати і навчати моделі машинного навчання за допомогою інтуїтивно зрозумілих високорівневих API, таких як Keras, з активним виконанням, що забезпечує негайну ітерацію моделі і простоту налагодження;
- надійне виробництво машинного навчання в будь-якому місці. Легке навчання і відкриття моделі в хмарі, локально, в браузері або на пристрої, незалежно від того, якою мовою розроблений продукт;
- потужні експерименти для дослідження. Проста і гнучка архітектура для перетворення нових ідей від концепції до коду, до сучасних моделей і до швидшої публікації;
- відкритий основний код. Кожен розробник може завантажити код та реалізувати його в своїх інтересах [26].

2.4 Порівняльна характеристика інструментаріїв для нейронної мережі

Згідно з створенням нейронної мережі на мові програмування C++ перед розробником постає вибір між трьома основними інструментаріями, які здатні повністю реалізувати задачу дипломної роботи. В таблиці 2.3 можна побачити основні характеристики, переваги та недоліки інструментаріїв та фреймворків.

Таблиця 2.3 – Характеристики інструментаріїв

Характеристики	Інструментарії		
	Dlib	Caffe	TensorFlow
Здатність до самонавчання	+	+	+
Великий обсяг кодування	+	-	-
Здатність до швидкої перевірки	+	+	+
Використання ресурсів комп'ютера	+	+	+
Робота C++ з MatLab	+	+	-
Кросплатформність	+	+	+
Оновлення версії	+	+	+
Велика спілька розробників	+	+	+

З таблиці 2.4 можна зрозуміти, що основні інструментарії не мають великої відмінності між собою, в основних характеристиках вони однакові та вибір падає саме на бажанність самого розробника, який інструментарій або фреймвор більш подобається йому.

2.5 Висновки за розділом 2

З цього розділу можна зрозуміти, що вибір пав на мову програмування C++, на думку багатьох розробників – саме ця мова найкраще відноситься до машинної мови, найкраще підходить до середовища програмування “MatLab”, хоча C++ дуже технічна мова для розробників –

початківців, бо має великий обсяг інформації, де розробнику потрібно дати особливу увагу для розробки саме з цією мовою. Також, для реалізації проблеми фінансування було вирішено розроблювати за допомогою функції та нейронній мережі – розроблювати в середовищі програмування “MatLab”, бо саме в цьому середовищі найкраще можна побачити помилку, знайти її та змінити її. В ході компілювання коду можна побачити процес в графіку, за потребою. Вибір між інструментаріями та фреймворком був дуже довгим. Відразу стало зрозуміло, що для реалізації задачі фреймворк TensorFlow – не дуже добре підходить, бо фреймворк найкраще працює з машинними задачами, а не з прогнозуванням фінансів та бюджетів та подальші роки. Тож, вибір був між Dlib та Caffe.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Структура програмного продукту

Оскільки "MatLab" – матрична система, розумно почати її опис з початкових відомостей про вектори та матриці. Двовимірний масив чисел математичних виразів прийнято називати матрицею. А одновимірний масив називають вектором. Вектори можуть бути двома типів: вектор рядок і вектор стовпець. Вектори та матриці характеризуються розмірністю та розміром. Розмірність визначає структурну організацію масивів у вигляді рядка, сторінки, куба. Так що вектор є одновимірним масивом, а матриця є двовимірним масивом з розмірністю. "MatLab" допускає завдання та використання багатовимірних масивів.

Розмір вектора – це число його елементів, а розмір матриці визначається добутком числа її рядків m та стовпців n . Зазвичай розмір матриці вказують як mn . Матриця називається квадратною, якщо $m = n$, тобто число рядків матриці дорівнює числу її стовпців.

"MatLab" – одна з найстаріших, ретельно опрацьованих та перевірених часів систем автоматизації математичних та науково-технічних розрахунків, будована на розширеному поданні та застосуванні матричних операцій. Це знайшло відображення у назві системи – Matrix Laboratory. Застосування матриць як основних об'єктів системи сприяє різкому зменшенню числа циклів, які дуже поширені при виконанні матричних обчислень звичайними мовами програмування високого рівня, та полегшення реалізації паралельних обчислень [27].

Одним з основних завдань при створенні системи "MatLab" завжди було представлення користувачам потужної мови програмування, орієнтованої на технічні та математичні розрахунки та здатного перевершити можливості традиційних мов програмування, які багато років використовувалися для реалізації чисельних методів. При цьому особлива

увага приділялася як підвищенню швидкості обчислень, так і адаптації системи до вирішення різноманітних завдань користувачів.

“MatLab” реалізує три важливі концепції програмування:

- процедурне модульне програмування, засноване на створенні модулів – процедур та функцій;
- об'єктно - орієнтоване програмування, особливо цінне у реалізації графічних засобів системи;
- візуально - орієнтоване програмування, спрямоване створення засобів графічного інтерфейсу користувача.

Мова програмування “MatLab” належить до класу інтерпретаторів. Це означає, що будь - яка команда системи розпізнається (інтерпретується) за її ім'ям (ідентифікатором) і негайно виконується в командному рядку, що забезпечує легку перевірку частинами будь-якого програмного коду. Одночасно інтерпретуючий характер мови програмування “MatLab” означає, що з перших рядків опису засобів цієї системи фактично описується її мова програмування [27].

Важливими перевагами системи є її відкритість та розширюваність. Більшість команд і функцій системи реалізовані як .m файлів текстового формату (з розширенням .m) і файлів мовою C / C++, причому всі файли доступні для модифікації. Розробнику дана можливість створювати не лише окремі файли, а й бібліотеки файлів для реалізації специфічних завдань. Будь - який набір команд у довідці можна виконати за допомогою команди Evaluate Selection контекстного меню правої клавіші миші.

Структура програмного продукту “Matlab” взагалі не відрізняється від інших програмних продуктів, основа яких — вирішення великих математичних задач шляхом додаванні даних до вже готової платформи. На рисунку 3.1 можна побачити схему роботи програми “MatLab” [27].

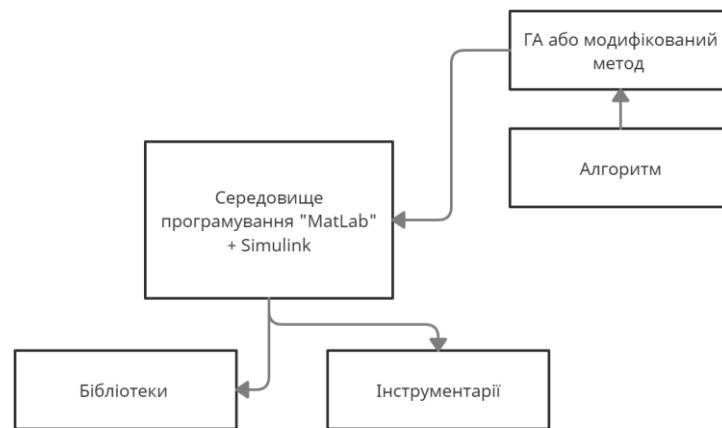


Рисунок 3.1 – Схема роботи середовища “MatLab”

З схеми 3.1 можна побачити, що середовище програмування “MatLab” складається з: бібліотек загального призначення; інструментаріїв; алгоритмів роботи та прикладних бібліотек, з якими можна працювати розробнику. На виході розробник отримує бажаний результат.

3.2 Опис коцепції роботи нейронної мережі з “MatLab”

Для реалізації поставленої задачі необхідно було провести аналіз всіх існуючих методів вдосконалення нейронної мережі та приєднання її до середовища програмування “MatLab”. Для вдосконалення нейронної мережі було обрано генетичний код та його алгоритм, бо саме на думку багатьох розробників генетичний код найпопулярніший серед інших алгоритмів. Також, для реалізації задачі була розроблена формула, за якою можна спрогнозувати фінансування, ця формула має вигляд (3.5).

Формула має основні риси, які потрібні для реалізації задачі.

$$F = (x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10}), \quad (3.5)$$

де F - фінансування для дітей – сиріт;

x_1 - всі абітурієнти, які поступили в учбовому році;

x_2 - похибка, яка може бути кожний рік;

x_3 - похибка (інфляція, зростання або спад курсу державної валюти);

x_4 - субвенції на харчування дітей – сиріт;

x_5 - робота викладачів ЗВО та технічних коледжах;

x_6 - середній відсоток дітей – сиріт, які поступило до ЗВО;

x_7 - похибка щодо дітей – сиріт, які закінчили своє навчання;

x_8 - стипендія для дітей;

x_9 - вік;

x_{10} - область, або територія на якій проходить навчання.

Для коректної роботи нейронної мережі та “MatLab” надо було зрозуміти основу роботи. Для цього була розроблена рисунок 3.3, на ній можна побачити роботу “MatLab” та генетичного коду [28].

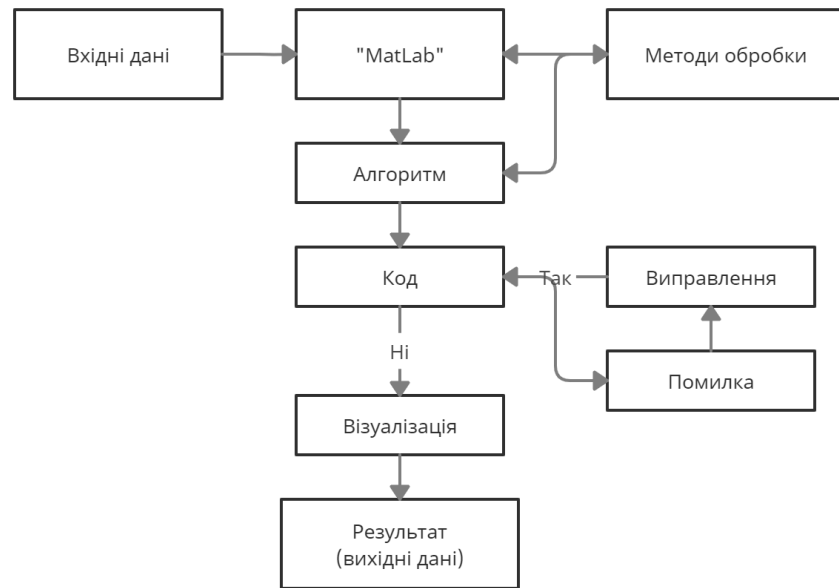


Рисунок 3.2 – Схема алгоритму роботи програми

З рисунка 3.2 можна зрозуміти, що на вході потрібні дані, які мають розробники. Програмне середовище “MatLab” включає в себе декілька робіт, а саме:

- алгоритм роботи. В нього можна вставити свій генетичний алгоритм, або корегувати все готові алгоритми;
- код. Наступний крок за алгоритмом. Саме в алгоритм потрібно вставити код для реалізації задачі;
- помилка. Якщо код має помилку, потрібно її вирішити, щоб дістатись наступного кроку;
- візуалізація. Якщо ніяких помилок не виявлено, розробник може для більш детальної уваги подивитись результат на графіку, або у “Profiler”;
- результат. Дані на виході.

3.3 Детальний опис роботи “MatLab” з генетичними алгоритмами

При розробці методу прогнозування фінансування на подальші роки для дітей – сиріт маютьсся дані про фінансування за минулі роки. Для повної

реалізації розробнику потрібно розробити структурну схему фінансування для дітей – сиріт. На рисунку 3.3 можна побачити основний процес роботи середовища програмування “MatLab” з ГА та основні риси при реалізації задачі.

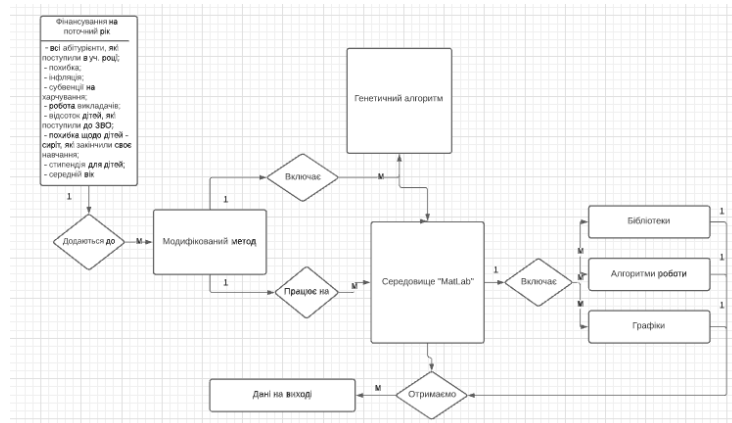


Рисунок 3.3 – Процес роботи ГА з “MatLab”

З рис. 3.3 можна побачити, що кожен рік фінансування для дітей – сиріт, які вступають та навчаються в ЗВО складається з однакових рис, але ці риси кожен рік змінюються, це залежить від фінансування для цих рис. Також с рисунка можна побачити залежність фінансування на наступний рік, тобто, щоб правильно підрахувати фінансування для дітей – сиріт на наступний рік беруться точні такі дані, що будуть в поточному році, риси не змінюються, а змінюється тільки бюджет для кожної риси.

3.4 Огляд основних методів для нейронної мережі

Працюючи з нейронними мережами можна зіткнутися з ситуацією, коли градієнт починає стрімко згасати (або навпаки, зростати). Дана проблема називається проблемою згасаючого (вибухового) градієнта або проблемою обігу градієнтів у нуль. В цьому випадку стає проблематично навчити нейронну мережу передбачати результат на основі вхідних даних, що відносяться до далекого для передбачуваного моменту минулому, в

результаті мережа швидко «забуває» інформацію про попередні стани, враховуючи лише недавні. Це може стати перешкодою при вирішенні завдань, у яких потрібно брати до уваги інформацію, отриману порівняно давно, про попередні стани нейронна мережа враховуватиме лише слова останньої пропозиції, втрачаючи загальний зміст. На сьогоднішній день розробник має великий та основний спектр типів мереж, з якими можна працювати. Маються на увазі LSTM, RNN та FFNN мережі.

Мережі з довгою короткостроковою пам'яттю (Long Short-Term Memory, LSTM), запропоновані З. Хохрайтером та Ю. Шмідхубером, були створені для вирішення проблеми градієнта, що згасає. Роль нейронів у такої мережі грають осередки, які самі по собі є невеликими нейронними мережами. Для збереження цінних даних протягом тривалого часу ці осередки використовують спеціальні фільтри, називаються вентилями чи шлюзами (gates). Таких вентилів три: вхідний, вихідний та вентиль забування. Вхідний вентиль при отриманні нової інформації визначає, чи варто нейрону запам'ятовувати її і якщо так, то яку її частину. Вихідний вентиль дозволяє визначити, яка частина з інформації, що знаходиться в пам'яті, важлива в даний момент, а якою слід знехтувати. І, нарешті, вентиль забування допомагає мережі видалити з пам'яті ту інформацію, яка вже не стане в нагоді в подальшій роботі [29].

Рекурентні нейронні мережі (RNN) - вид нейронних мереж, в яких є зворотний зв'язок. При цьому під зворотним зв'язком мається на увазі зв'язок від більш віддаленого елемента до менш віддаленого. Наявність зворотних зв'язків дозволяє запам'ятовувати та відтворювати цілі послідовності реакцій однією стимул. З точки зору програмування в таких мережах з'являється аналог циклічного виконання, а з точки зору систем така мережа еквівалентна кінцевому автомату. Такі особливості потенційно надають безліч можливостей для моделювання біологічних нейронних мереж [30].

Нейронні мережі прямого поширення (FFNN) - дуже прямолінійні, вони передають інформацію від входу до виходу. Найпростіша нейронна

мережа має дві вхідні клітини і одну вихідну, і може використовуватися як модель логічних вентилів. FFNN зазвичай навчається методом зворотного поширення помилки, у якому мережа отримує безлічі вхідних і вихідних даних. Цей процес називається навчанням з учителем, і він відрізняється від навчання без вчителя тим, що у другому випадку безліч вихідних даних мережу складає самостійно. Вищезгадана помилка є різницею між введенням та висновком. Якщо мережа має достатню кількість прихованих нейронів, вона теоретично здатна змодельовати взаємодію між вхідним і вихідними даними. Практично такі мережі використовуються рідко, але часто комбінують з іншими типами для отримання нових та модифікованих методів [31].

Також, якщо вже чітко описані методи потрібно розібратись з поняттям Dropout. Метод регуляції штучних нейронних мереж, призначений для зменшення переобучення мережі для обліку передотвращення складних коадаптацій окремих нейронів на тренувальних даних під час навчання [32].

Для навчання та тестування нейронних мереж були завантажені дані про підрахунки соціального населення з 01.01.2016 по 31.12.2020, які доступні на різних сайтах.

Завантажені дані мали наступні поля:

- всі абітурієнти, які поступили до лав ЗВО;
- похибка вступивших до ЗВО;
- інфляція;
- харчування, проживання для дітей – сиріт;
- оплата роботи викладачів ЗВО (середня зарплата для одного викладача);
- відсоток дітей – сиріт, які вступили до ЗВО;
- середній вік для дитини, яка поступає до навчального закладу.

Приклад завантажених даних можна побачити на рисунку 3.4.

```

151034,2132,3.0,4003240,6200,88.3,6032,2505,16
142434,2000,3.0,3856085,6000,88.0,6032,2505,16
138975,1887,3.0,3630541,5600,87.2,5754,2300,16
137751,1752,3.0,3418401,5325,86.0,5670,2234,16
130014,1670,3.0,3301321,5201,85.6,5560,2012,16

```

Рисунок 3.4 – Завантаження даних

Оскільки нейронна мережа погано сприймає ненормовані дані, було проведено попередню обробку даних. Поля дати та часу були відкинуті, оскільки надані дані мали фіксований інтервал між записами, що дорівнює одній годині. Потім була зроблена стандартизація даних всіх полів, що залишилися, в результаті дані були перетворені на значення, що лежать на відрізку $[-1; 1]$.

Вибір малого розміру, маємо що при використанні вибірки розміру 100 та навчанні мереж на протязом 60 епох були отримані схожі результати для мереж усіх видів, відмінності в точності прогнозування незначні. В таблиці 3.1 можна побачити середньоквадратичну помилку мереж при вибірці малого розміру.

Таблиця 3.1 – Помилка при вибірці малого розміру

LSTM	LSTM+ Dropout	RNN	RNN+ Dropout	FFNN	FFNN+ Dropout
0.06 %	0.04 %	0.08 %	0.06 %	0.09 %	0.07 %

Вибірка середнього розміру мається, що при використанні вибірки розміру 5000 найбільшої точності можна добитись рекурентною нейронною мережею. Особливо добре це помітно при використанні мереж, що використовують шари з 120 нейронів та навчених за 20 епох. У цьому випадку добрі результати були показані мережами як із Dropout, так і без нього. При цьому мережі LSTM показали найкращі результат як у плані

точності, і за часом навчання. Середньоквадратична помилка та час навчання мереж на вибірці середнього розміру представлені в Таблиці 3.2.

Таблиця 3.2 – Помилка при вибірці середнього розміру

Тип мережі	Помилка у відсотках та час обробки у секундах			
	Нейрони: 40, епохи: 60	Нейрони: 80, епохи: 60	Нейрони: 120, епохи: 30	Нейрони: 120, епохи: 90
LSTM	0.56% 60 с.	0.43% 66 с.	0.45% 61 с.	0.15 % 82 с.
LSTM+ Dropout	0.52 % 65 с.	0.38% 72 с.	0.39 % 69 с.	0.09 % 87 с.
RNN	0.61% 70 с.	0.42 % 74 с.	0.39 % 69 с.	0.15 % 113 с.
RNN+ Dropout	0.55 % 72 с.	0.38 % 78 с.	0.34 % 69 с.	0.13 % 135 с.
FFNN	0.64 % 75 с.	0.54 % 87 с.	0.48 % 86 с.	0.17 % 124 с.
FFNN+ Dropout	0.56 % 74 с.	0.42 % 91 с.	0.43 % 88 с.	0.12 % 138 с.

Вибір великого розміру мається, що на вибірці розміру 10000 більшість мереж стали виявляти ознаки перенавчання. Особливо це було виражено для мереж LSTM, не що використовують Dropout. Однак мережі LSTM з методом Dropout, навпаки, показали більш високу точність і виявили ознаки перенавчання лише зі збільшенням кількості нейронів у шарі. Як і в попередніх випадках, мережі LSTM вимагали найбільшого часу навчання. Середньоквадратична помилка та час навчання мереж на вибірці великого розміру представлені в Таблиці 3.3. Також, після вибору типу мережі LSTM було вирішено саме цей тип модифікувати для реалізації задачі, модифікований метод має назву HUGO.

Таблиця 3.3 – Помилка при вибірці великого розміру

Тип мережі	Помилка у відсотках та час обробки у секундах		
	Нейрони: 40 епохи : 60	Нейрони: 80 епохи : 60	Нейрони: 120 епохи : 60
LSTM	0.53 % 116 с.	0.34% 156 с.	0.18 % 401 с.
LSTM+ Dropout	0.48 % 143 с.	0.28 % 163 с.	0.16 % 406 с.
RNN	0.58 % 69 с.	0.38 % 61 с.	0.24 % 71 с.
RNN+ Dropout	0.54 % 68 с.	0.35 % 63 с.	0.21 % 81 с.
FFNN	0.51 % 19 с.	0.42 % 19 с.	0.26 % 20 с.
FFNN+ Dropout	0.47 % 24 с.	0.37 % 24 с.	0.23 % 25 с.

Після цих процесів слід сказати, що мережі із прямим поширенням сигналу показали найменшу точність прогнозування, але були найменш вимогливими до обчислювальним ресурсам у процесі навчання. Нейронні мережі з довгою короткочасною пам'яттю показали найкращий результат при навчанні на вибірках великого розміру та використання методу Dropout, що значно скорочує вплив перенавчання під час використання об'ємних вибірок, проте поступилися рекурентних нейронних мереж за показником мінімальної помилки. Крім того, вони показали найгірший час навчання. Це пов'язано з більш складною структурою та використанням як функція активації гіперболічний тангенс. Також, були проведені графіки точності, з яких стало зрозуміло, що найкраще для реалізації поставленої задачі підходить саме LSTM метод.

3.5 Порівняння та вибір генетичного алгоритму

При вирішенні поставленої задачі перед розробником постає задача вдосконалення та модифікація вже існуючих методів генетичного алгоритму. Серед існуючих методів ГА найкраще всього підходить острівна модель генетичного алгоритму.

Острівна модель є найпоширенішим методом паралельного генетичного алгоритму. Її суть полягає в тому, що популяція, як правило, що складається з дуже великої кількості особин, розбивається на однакові за розміром підпопуляції. кожна підпопуляція обробляється відбільним процесором за допомогою одного з різновидів непаралельного генетичного алгоритму. Справжнє рішення перебуває лише одному острові, а кілька помилкових домінант - інших островах. Тоді при міграції кількість хибних особин на островах зростає (на кожному острові міграції відбуваються не менше двох островів), генетичним алгоритмом вірне рішення буде зруйновано. Тим самим у невеликій популяції при генетичному дрейфі можлива поява помилкових домінуючих особин і сходження алгоритму до помилкового оптимуму. Введення міграції в острівній моделі дозволяє знаходити різні особи-домінанти у підпопуляціях, що сприяє підтримці різноманіття у популяції.

Кожну підпопуляцію можна сприйняти за острів. Під час міграції підпопуляції обмінюються генетичним матеріалом. При частому мігруванні великої кількості особин відбувається перемішування генетичного матеріалу. Тим самим усуваються локальні різниці між островами. Дуже рідкісні міграції не дозволяють запобігти передчасній збіжності алгоритму на маленьких популяціях. На рисунку 3.8 можна побачити розглянуту модель з кожного острова міграції можуть відбуватися лише на певну відстань: 2-5 кістяків залежно від кількості популяцій. Таким чином, кожен острів виявляється майже ізольованим. Кількість островів, куди можуть мігрувати

особини однієї підпопуляції, називають відстанню ізоляції. Слід зазначити, що у такій моделі взаємоміграції виключені.

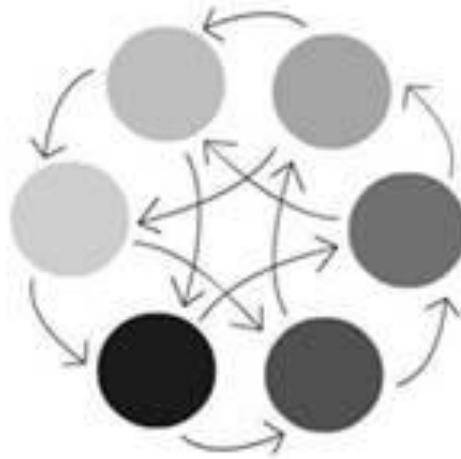


Рисунок 3.5 – Приклад острівної моделі ГА

Основною перевагою острівної моделі генетичних алгоритмів є те, що використовуючи таку модель, можна моделювати різні комбінації способів відбору та формування наступного покоління або зробити так, що в різних популяціях використовуватимуться різні комбінації операторів генетичного алгоритму.

Отже, перед модифікацією генетичного алгоритму шляхом острівного методу потрібно з'ясувати схему роботи нейронної мережі з середовищем програмування “MatLab”. На рисунку 3.9 представлений принцип роботи ГА з нейронною мережею та дуже зрозуміло показаний принцип роботи всіх компонентів для реалізації задачі.

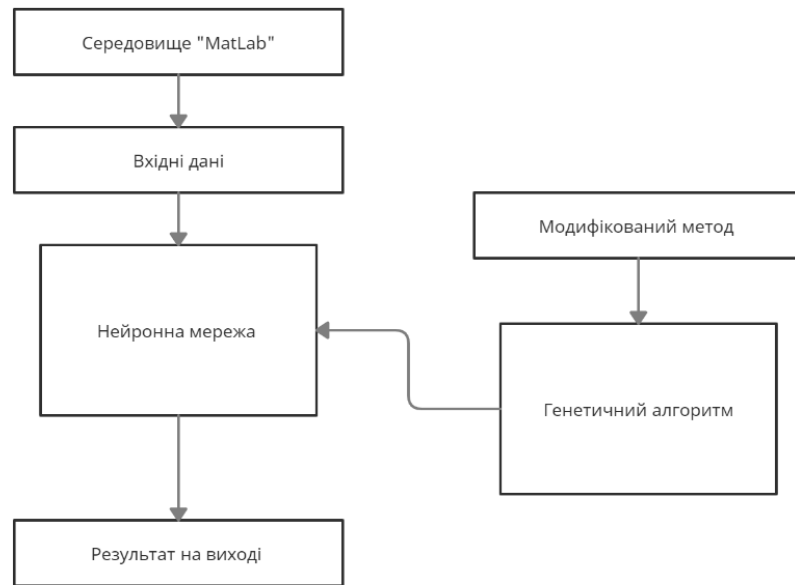


Рисунок 3.6 – Схема принципу роботи ГА з нейронною мережею

При виборі найкращого методу генетичного алгоритму для реалізації прогнозування фінансування для дітей – сиріт було розроблено таблицю всіх популярних ГА, які мають допомогти при вирішенні проблеми. В таблиці 3.4 можна побачити зрівняння основних методів ГА та острівний метод, який має п'ять острова [33].

Таблиця 3.4 – Зрівняння методів генетичного алгоритму

Генетичний алгоритм	Час пошуку вирішення		Запит до функції		Кількість популяцій	
	Ср. Сек.	Дис. ϵ^2 .	Ср.	Дис.	Ср.	Дис.
Стандартний ГА	29.9	592.4	1267039	1.017*10 ¹²	74.9	3584
Га + РВКіМ (регуляція кросингвера та мутації)	26.4	670.3	1114035	9.152*10 ¹¹	89.4	5600
Га + КіМ (в поколінні)	15.6	284	780658	6.156*10 ¹¹	237	58600
Га + ДСП (динамічна самоорганізація операторів)	54.2	1360	1925972	2.469*10 ¹²	68.4	1226
Острівний ГА (2 острова)	12	17	560247	2.113*10 ¹⁰	-	-
Острівний ГА (3 острова)	19	37	859917	5.173*10 ¹⁰	-	-
Острівний ГА (4 острова)	23	48	113007	6.433*10 ¹⁰	-	-
Острівний ГА (всі острови)	31	81	135768	9.944*10 ¹⁰	-	-

ГА з КіМ у кожному поколінні показав кращі результати порівняно з близьким до нього та адаптованим ГА з КіМ. Це можна пояснити складністю функції Растрігіна. Очевидно, велика кількість мутацій забезпечила більше швидке досягнення значення. Генетичний алгоритм острівного метода дозволяє в 3 рази швидше знаходити рішення, порівняно з ПГА (простим генетичний алгоритмом). При цьому перебирається у 3 рази менша кількість хромосом. Слід також відзначити високу надійність рішень, отримуваних острівним методом – дисперсія часу пошуку рішення за допомогою острівного метода менше, ніж у решти ГА. З таблиці 3.4 видно, що острівний генетичний алгоритм з динамічним визначенням періоду міграцій та

динамічною визначенням періоду міграції вибору показує дуже добрі показники, аніж всі інші методи. Регулюванням ймовірностей вибору генетичних алгоритмів показує відносно добрі результати і для більшої кількості островів. Це дозволяє позиціонувати ГА острівного методу як найбільш підходящий алгоритм пошуку. рішень цільових функцій, розрахунків яких займає тривалий час [33].

3.6 Модифікація генетичного алгоритму

Під час реалізації поставленої задачі було обрано генетичний алгоритм острівного методу, зараз, перед розробником постає ціль модифікувати метод для реалізації прогнозування фінансування дітей – сиріт, які вступили та навчаються в ЗВО.

Для вирішення завдання прогнозування фінансування дітей – сиріт у ЗВО адекватним буде метод оптимізації, який здатен знаходити репрезентативні множини альтернативних оптимальних конструкцій, замість однієї або кількох дуже близьких рішень, а алгоритми, що їх реалізують, повинні мати високою обчислювальною швидкістю і, зокрема, здатними використовувати переваги комп'ютерів з багатоядерними процесорами. Тому розробка саме таких алгоритмів для вирішення числових та завдань в сфері прогнозування є найбільш затребувана на сьогодні. У цьому, перспективним виглядає підхід, заснований на спільному використанні як класичних, і сучасних методів оптимізації генетичного алгоритму, і навіть їх модифікацій та різних комбінацій. Основним призначенням методів теорії прогнозування фінансування, в даному у випадку, є приведення опису якості та функціональних обмежень у просторі конструктивних та режимних параметрів до єдиного універсального вигляду, що представлено в формулі 3.1.

$$Y(q) = A_0 + \sum_{i=1}^n (A_i q_i + A_{ii} q_i^2) + \sum_{i=1}^{n-1} \sum_{j=i+1}^n A_{ij} q_i q_j, \quad (3.1)$$

де A - матриця коефіцієнтів;

Q - вектор нормованих значень факторів;

n - число факторів;

Q - вектор параметрів, що варіюються.

Для всіх компонентів вектору Q є відношення $-1 \leq q_i \leq 1$. Крім того, за допомогою прямої залежності, яка представлена в формулі 3.2

$$q_i = f_{iQ}(Q_i) = (2Q_i - Q_{i\max} - Q_{i\min}) / (Q_{i\max} - Q_{i\min}) \quad (3.2)$$

В формулі здійснюються операції нормування, що забезпечують однозначні зв'язки між параметрами простору G_Q та відповідного йому простору.

Не вдаючись докладно в теорію ГА, потрібно відзначити лише, що вони засновані на моделюванні процесу еволюції з урахуванням факторів спадкування або схрещування, мінливості, тобто мутації та відбору найбільш пристосованих особин (кращих рішень). Робота генетичного алгоритму при вирішенні оптимізаційних завдань починається з побудови деякого множини (початкової популяції) допустимих рішень (особин). Потім за допомогою численних випадкових змін компонент вектора параметрів керування здійснюється цілеспрямований пошук найефективніших рішень (більш пристосованих особин). При цьому, чим більша пристосованість особини, тим більша її можливість відтворити своїх нащадків (нових рішень) при формуванні популяцій нових поколінь. Класична схема роботи генетичного алгоритму можна побачити на рисунку 3.7.

Існує безліч варіантів реалізації ГА, що відрізняються методами схрещування, мутації, відбору рішень тощо. Для вирішення поставленого завдання за основу було взято варіант ГА, який реалізує ідею міграції та

штучної селекції, а саме острівну модель еволюції. На відміну від звичайних ГА, тут реалізується макроеволюція, тобто. створюється не одна популяція, а кілька популяцій, в залежності від островів [34].

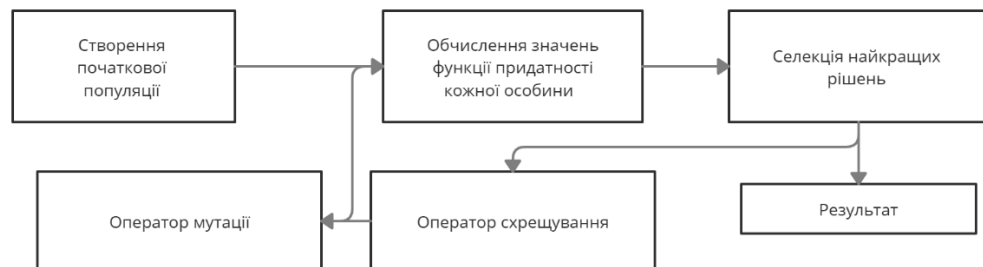


Рисунок 3.7 – Схема роботи генетичного алгоритму

У стандартному варіанті острівної моделі генетичний пошук здійснюється шляхом об'єднання батьків із різних (острівних) популяцій (рисунок 3.11).

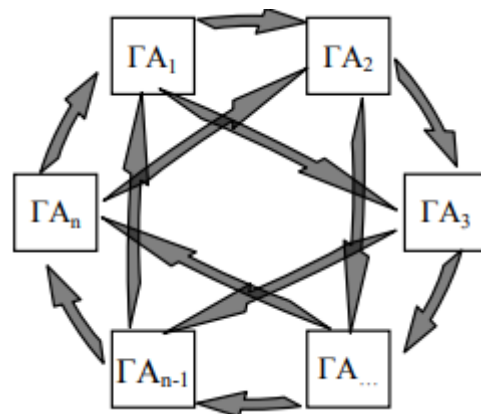


Рисунок 3.8 – Острівна модель ГА

У цій моделі кожна з популяцій (безліч допустимих конструкцій) удосконалюється окремо за допомогою одного із стандартних ГА, що асоціюється з еволюційним розвитком особин на кількох ізольованих островах (незалежних оптимізаційних процесах). Через кілька поколінь відбувається міграція – острівні популяції конструкцій обмінюються певною кількістю кращих варіантів, що дозволяє поєднати «досягнення» незалежних

оптимізаційних процесів на різних островах отримання найкращого рішення. Слід зазначити, що острівна модель ГА добре пристосована для реалізації багатопотокових сценаріїв обчислень, тому що острівні еволюційні процеси можуть моделюватись на окремих процесорах, що відповідає одному з принципів розвитку сучасних оптимізаційних алгоритмів.

Найкращий варіант модифікації острівної моделі ГА для вирішення завдання прогнозування фінансування для дітей – сиріт, які вступили та навчаються в ЗВО має такі відмінні особливості. Число островів дорівнює числу можливих режимів експлуатації. На кожному острові умови розв'язання острівної оптимізаційної задачі визначаються режимними параметрами, значення яких відповідають конкретному режиму експлуатації. Тобто, острівна модель здійснюється для двох островів. Після трьох острівних поколінь здійснюється селекція осіб, які призначені для «переселення» на інші острови. При цьому вибираються лише ті особи (варіанти конструкції), які найбільш ефективно працюють на всій гамі режимів. Всі разом вони утворюють безліч конструкцій, які на наступній ітерації вирішення загальної оптимізаційної задачі відіграватимуть роль нової популяції всіх острівних ГА [35].

При схрещуванні (модифікації) відібраних батьків (рис фінансування) у процесі генерації нових поколінь використовується алгоритм кросовера речовим кодуванням. Особливість даного алгоритму в тому, що він допускає можливість виходу значень параметрів нащадків за межі значень цих параметрів у батьків на величину Δa , при цьому батьки генерують одного нащадка, що характеризується вектором $\bar{q}_3$, компоненти, якого є випадковими числами з відповідних інтервалів формули 3.3.

$$\left[\bar{d}_{\min} - \bar{\Delta} a \bar{d}_{\max} + \bar{\Delta} a \right], \quad (3.3)$$

$$\text{де } \bar{d}_{\max} = \max(\bar{q}_1 \bar{q}_2);$$

$$\bar{d}_{\min} = \min(\bar{q}_1 \bar{q}_2);$$

$$\bar{\Delta} = \bar{d}_{\max} - \bar{d}_{\min}.$$

Для випадкової зміни значень конструктивних параметрів застосовується низка загальновідомих операторів мутації. Після закінчення роботи ГА з останніх острівних популяцій прогнозування фінансування здійснюється вибір конструкцій найбільш пристосованих для роботи із заданим графіком зміни навантаження. Зрештою, формується безліч конструкцій, які відповідають вимогам поставленого завдання прогнозування фінансування. Відповідне ранжування показників їхньої інтегральної ефективності, а також спеціальна процедура селекції забезпечують знаходження репрезентативного безлічі необхідних рішень на безлічі Парето. У формалізованому вигляді описаний процес оптимізації наводиться нижче (рис. 3.). При здійсненні критерій якості компоненти вектору $\bar{Q}$ формуються з параметрів простору G_q . Відповідних конструктивним та режимним параметрам простору G_x . У нормованому вигляді він має такий вигляд (формула 3.4).

$$\bar{q} = \{ \bar{x}_c, \bar{x}_r \}, \quad (3.4)$$

де $\bar{x}_c$ та $\bar{x}_r$ - вектори шляху. З огляду на це функція пристосованості острівних оптимізаційних завдань виглядатимуть так на формулі 3.5.

$$\bar{F}_{is}(\bar{x}_c, \bar{x}_r) = \sum_{k=1}^{Nk} \mu_k (\bar{Y}_k(\bar{x}_c, \bar{x}_r))^2, \quad (3.5)$$

де $\bar{F}_{is}$ - функція модифікації острівної моделі;

Nk - кількість критерій якості;

μ_k - вага критерій якості;

$\bar{Y}_k$ - нормоване значення критерію якості.

Далі, щоб отримати загальну функцію оптимізації острівного методу потрібно з'єднати головні формули та можна отримати формулу 3.6.

$$\bar{F}_{glob}(\bar{x}_c, \bar{x}_r) = \sum_{r=1}^{Nr} (\bar{F}_{is}(\bar{x}_c, \bar{x}_r))^2 \quad (3.6)$$

Функція 3.6 несе в собі інформацію про сумарну ефективність методу на всіх режимах модифікації, на те, що сама конструкція не може мати максимально високі значення критеріїв якості на всіх режимах оптимізації.

3.7 Висновки за розділом 3

В результаті виконання розділу розглянуто структурну схему середовища програмування, а саме “MatLab”. Також, був проведений аналіз існуючих методів для роботи з нейронними мережами, та був обраний саме LSTM метод, бо для реалізації поставленої задачі саме він найкраще підходить. Також, в ході роботи з'ясували принцип роботи середовища програмування “MatLab” в співпраці з нейронними мережами. Після з'ясування принципу роботи було проаналізовано ГА та методи генетичного алгоритму, після аналізу стало зрозуміло, що найкраще за всі основні методи ГА підходить саме острівний метод, після чого острівний метод було

модифіковано. Всі зрівняння, графічний опис та експерименти LSTM нейронної мережі та модифікації можна побачити в 6 розділі.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування програми

4.1.1 Призначення програми

Розроблений програмний продукт повністю відповідає реалізовано відповідно до поставленого завдання та задовольняє всім необхідним вимогам.

Програмний продукт призначений для визначення та прогнозування дітей - сиріт, що поступають до навчання в ЗВО. Може бути використаний для статистиків, які прогнозують фінансування дітей – сиріт на подальші декілька років.

4.1.2 Умови застосування програми

Зазначене програмне забезпечення написане мовою C++. Розробка і тестування проводилися на обладнанні з такими конфігураціями:

- процесор Intel Core i3 – 4005U, 1.7 GHz;
- 8 GB оперативної пам'яті;
- операційна система Windows 10 (x-64).

Розроблений метод може бути використаний на операційній системі Windows. Враховуючи використання в розробленому програмному пакеті компонентів, готовий програмний продукт повністю працює на операційній системі та задовольняє всі програмні потреби. Але, якщо, щось із пакетів до операційної системи Windows буде не завантажено до РС – можлива некоректна робота с продуктом, або зовсім можливі помилки на початкових стадіях роботи. До таких основних залежностей належать:

- мова програмування C++;
- нейромережева бібліотека Caffe;
- бібліотека машинного навчання <iostream>;

– виклик двох бібліотек для роботи в “MatLab” – “MatLabEngine.hpp” та “MatLabDataArray.hpp”.

4.2 Звернення до програми

Для запуску програми необхідно впевнитися у відповідності характеристик та параметрів, на яку встановлюється програма, до мінімальних системних вимог, упевнитися у справності та у коректній співпраці всіх компонентів та електронного носія, з якого буде здійснений запуск програми. Впевнетись, що програма “MatLab” повністю завантажена з усіма додатковими пакетами до PC на платформі Windows.

На рисунку 4.1 можна побачити файлову структуру та програмного продукту “MatLab”.

Файл `main.m` містить повністю робочу програму, де розробник має змогу її компілювати в реальному часі. Шлях до цього файлу, як і сам файл, повинен знаходитись на комп’ютері користувача.

Весь процес роботи знаходиться в файлі `Editor`, саме цей файл – керівник програмного коду, в ньому можна редагувати, доповнювати, вставляти свої алгоритми та відстежувати роботу в реальному часі.

Для програмної реалізації в програмі “MatLab” для розробника значно впростили спосіб збереження даних на виході. Сам основний файл, де розробник готує програмний продукт – завжди невеликого розміру. Цей файл можна з легкістю перенести на додаткові носії для роботи на інших платформах.

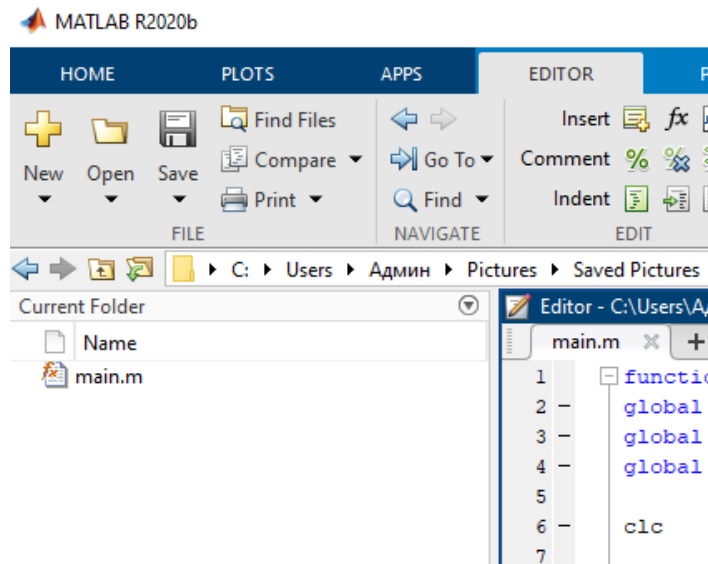


Рисунок 4.1 – Файлова структура “MatLab”

4.3 Вхідні та вихідні дані

Для реалізації задачі були взяті дані з різних форумів статистики для вирішення проблеми фінансування дітей – сиріт, які поступають до ЗВО на території України. Ці дані можна завантажити до свого РС, або просто переписати та вставити до програмного коду в програмний продукт “MatLab”. Приклад вхідних даних та результат роботи можна побачити на рисунку 4.2.

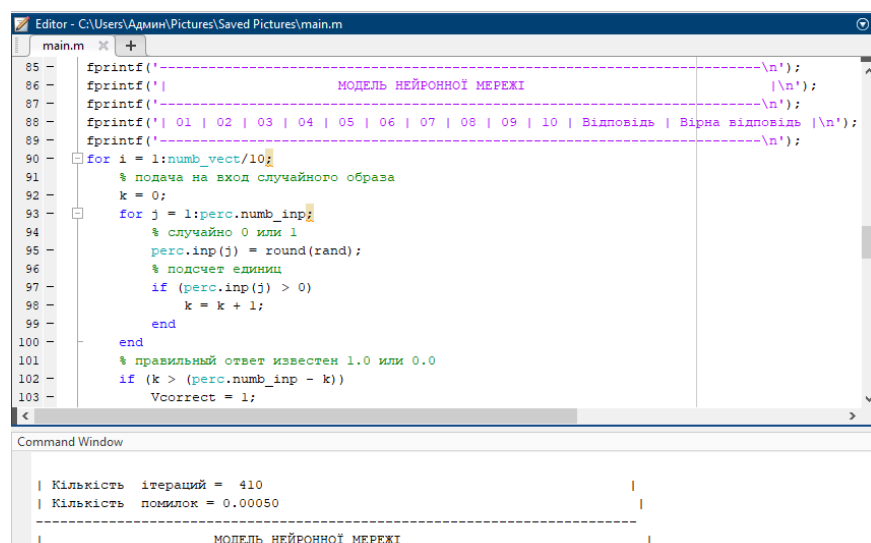


Рисунок 4.2 – Вхідні дані до “MatLab”

В компілюванні “MatLab” дуже добре розвинута здатність до знаходження та відображення помилок в програмному коді. У розробника є можливість відстежувати помилку на заданому кроці. Можна виділити ту строчку коду, яка, на думку розробника, має проблему. Якщо, розробник закомпілює свою роботу та на виході буде помилка – розробнику в вікні повідомлень буде повідомлення. На рисунку 4.3 можна побачити повідомлення до розробника, що програмний код має помилку.

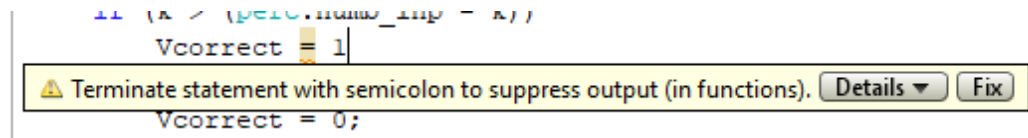


Рисунок 4.3 – Повідомлення розробнику про помилку

З новою версією робота в “MatLab” стала для розробника ще легша, якщо розробник допустив декілька помилок – то сам “MatLab” вкаже на ці помилки та перенаправить розробника до них. В процесі знаходження помилок в коді розробник може натиснути на кнопку в новому вікні на “Fix” та помилка пропаде змінивши своє значення на робоче.

Для більш детальної роботи розробник в процесі компілювання може відкрити вікно “Profiler” – де може побачити детальну інформацію по кожному кроку. На рисунку 4.4 – показана робота “Profiler” в нейронній мережі. Профіль дії (“Profiler”) визначає час виконання функції. Його можна використовувати, щоб запустити, зупинити та перезапустити “Profiler”, також для детальної інформації можна переглянути чи очистити статистику профілю.

Функціональна таблиця під рамкою відображає інформацію, подібну до графіку на часі. Спочатку функції відображаються в порядку, за яким вони були оброблені.

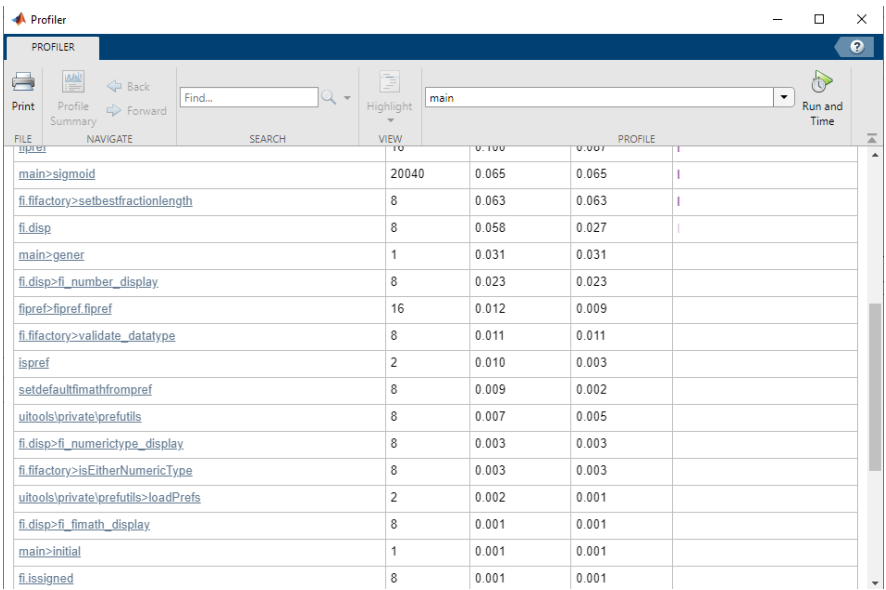


Рисунок 4.4 - Робота “Profiler”

Розробник може на кожному кроці побачити роботу “Profiler” та побачити кількість часу на обробку функції.

Для кожної функції “Profiler” можна зробити основну таблицю, яка описує функції та дуже добре описана в таблиці 4.1.

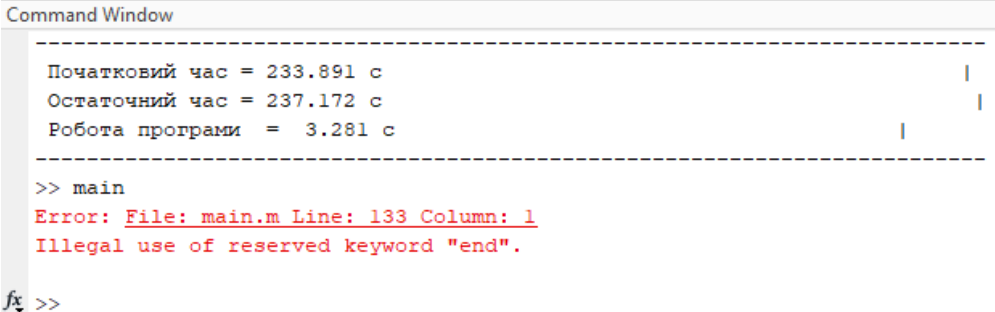
Таблиця 4.1 – Функції “Profiler”

Колона	Опис
Назва функції	Ім'я функції, яку викликає код
Визов функції	Кількість разів, коли код викликав функцію
Основний час на обробку	Загальний час, проведений у функції, в секундах. Час виконання функції включає час, проведений у функціях. Сам профайлер займає деякий час, який входить в результати.
Самостійний час на обробку	Загальний час у секундах, проведений в функції, без урахування часу, проведеного в будь-яких функціях
Ділянка загального часу	Графічний дисплей, що показує власний час у порівнянні з загальним часом

4.4 Повідомлення програмісту

В середовищі “MatLab” дуже добре розвинуте повідомлення програмісту про помилку в коді, також можливі повідомлення можуть виникати, коли встановлені залежності є некоректними (невірні версії бібліотек або мови програмування).

На рисунку 4.5 наведено можливе повідомлення програмісту, що в процесі розробки та компілювання є велики, або незначні помилки, які можуть вплинути на програмний код та на реалізацію задачі. Звісно, якщо що розробники мають різні версії – можлива некоректна робота з сумісництвом версій.



```

Command Window

-----
Початковий час = 233.891 с
Остаточний час = 237.172 с
Робота програми = 3.281 с
-----
>> main
Error: File: main.m Line: 133 Column: 1
Illegal use of reserved keyword "end".
fx >>
  
```

Рисунок 4.5 – Помилка в програмному коді

“Command Window” – дуже корисне вікно для розробників, бо саме в ньому можна побачити всі основні процеси, які можуть бути при розробці. Це вікно показує, що в програмному коді маються помилки, які не впливають, або впливають на результат компілювання та самої роботи. Також, це вікно показує результат самої роботи, вже готової програми. На рисунку 4.6 можна побачити помилки, які при компілюванні можуть бути, але вони ніяк не впливають на результат роботи.

“Warnings Window” – відображає попереджувальне повідомлення та встановлює стан попередження для функції останнього попередження.

Якщо повідомлення порожнє, попередження скидає стан попередження до останнього попередження, але не відображає жодного тексту.

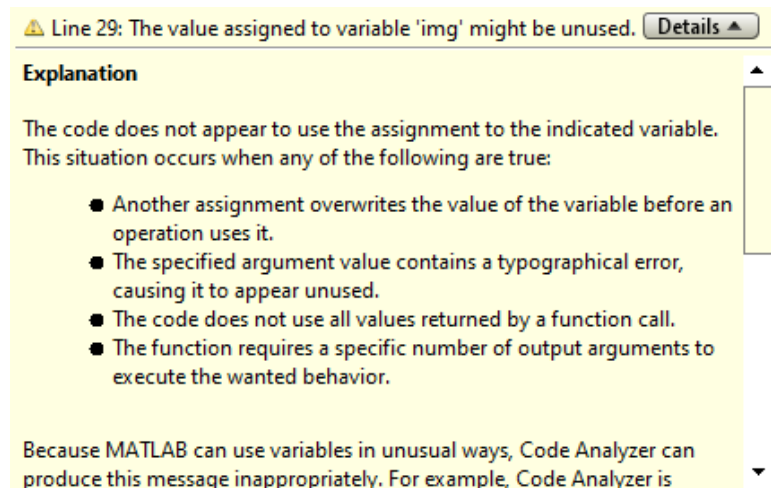


Рисунок 4.6 – Програмне вікно “Warnings Window”

4.5 Висновки за розділом

У розділі наданий опис для роботи з розробленим програмним продуктом. Надані параметри програмних та апаратних складових. Описані необхідні програмні залежності, попереднього встановлення яких потребує програмний продукт.

У розділі надано детальний опис методів розгортання програмного продукту, опис структурної схеми, вхідних та вихідних даних, а також можливих повідомлень під час роботи з ним, а саме виявлення помилок та допрацьовування в програмному коді.

5 КЕРІВНИЦТВО КОРИСТУВАЧА

5.6 Призначення та умови застосування програми

5.6.1 Призначення програми

Розроблений програмний продукт відповідає всім характеристикам до поставленого завдання та задовольняє всі необхідні вимоги поставленої задачі.

Програмний продукт призначений для визначення та фінансування дітей – сиріт, які в потчному році поступають до навчання в ЗВО або в технічні коледжі. Може бути використаний для статистики у відповідних статистичних галузях.

5.6.2 Умови застосування програми

Зазначене програмне забезпечення написане мовою C++. Розробка і тестування проводилися на обладнанні з такими конфігураціями:

- процесор Intel Core i3 – 4005U, 1.7 GHz;
- 8 GB оперативної пам'яті;
- операційна система Windows 10 (x-64).

Розроблене програмне забезпечення може бути використано на операційній системі сімейств Windows. Розроблене програмне забезпечення може бути використано на операційній системі сімейств Windows. Враховуючи використання в розробленому програмному пакеті компонентів, готовий програмний продукт повністю працює на операційній системі та задовольняє всі програмні потреби. Але, якщо, щось із пакетів до операційної системи Windows буде не завантажено до РС – можлива некоректна робота с продуктом, або зовсім можливі помилки на початкових стадіях роботи. До таких основних залежностей належать:

- мова програмування C++;

- нейромережева бібліотека Caffe;
- бібліотека машинного навчання <iostream>;
- виклик двох бібліотек для роботи в “MatLab” – “MatLabEngine.hpp” та “MatLabDataArray.hpp”.

5.2 Виконання програми

5.2.1 Екран входу

Програмний продукт “MatLab” ідеально був розроблений, бо працює майже на всьому. Користувач може працювати вже з встановленою програмою на РС з усіма, як для користувача, додатоками на функціями. Якщо користувач не має на своєму комп’ютері програми – це не проблема, бо розробники “MatLab” розробили спеціально браузерний додаток “MatLab” Online для користувачів, у яких не має встановленого “MatLab” на РС. Це дуже спрощує затрати та час у користувачів. На рисунку 5.1 можна побачити браузерний додаток “MatLab” Online. Слід сказати, що онлайн додаток буде працювати на безкоштовній основі всього 30 днів. Потім, потрібно буде придбати онлайн ліцензію. “MatLab” Online має ще одну незначну помилку. Web додаток не працює на всіх браузерах, Дуже добре він працює з Microsoft Edge та Google Chrome, с Opera – не працює.



Рисунок 5.1 – Вебзастосунок “MatLab” Online

5.2.2 Екран роботи алгоритму

Як було сказано раніше, що для роботи користувачу потрібно завантажати програму “MatLab”, або користуватись Web застосунком “MatLab” Online. Звісно, що користувач може працювати не тільки в програмі “MatLab”, а в будь – якому програмному додатку, який повністю задовольняє програмні потреби. На рисунку 5.2 та 5.3 можна побачити програмну реалізацію нейронної мережі.

Command Window

```
| Кількість ітерацій = 889 |
| Кількість помилок = 0.00050 |
-----
|               НЕЙРОННА МЕРЕЖА               |
-----
| 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | Відповідь | Вірна відповідь |
-----
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0.01 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0.00 | 0 |
fx | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1.00 | 0 |
```

Рисунок 5.2 – Робота в “Command Window”

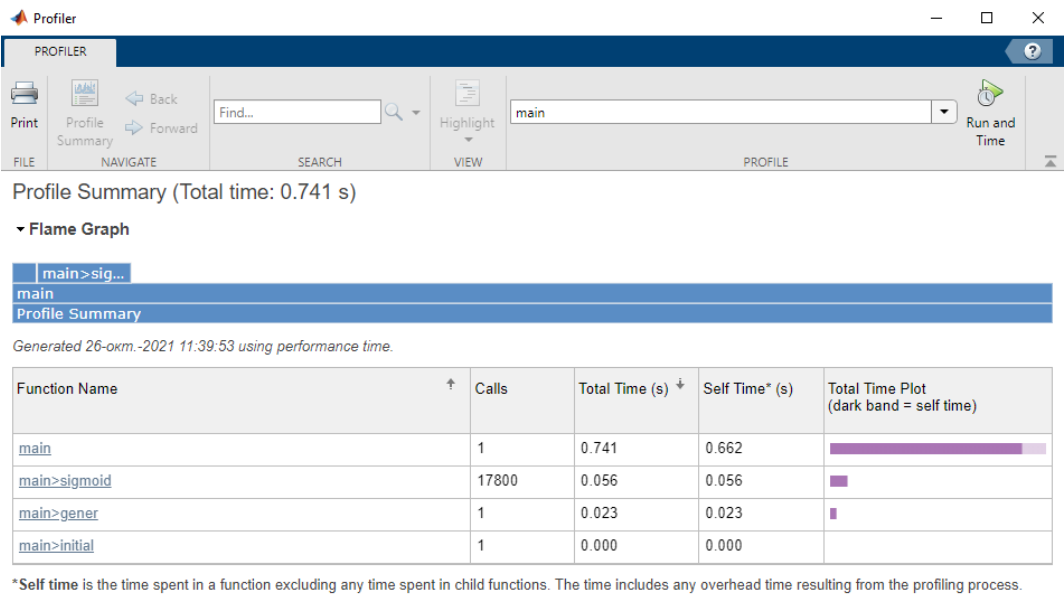


Рисунок 5.3 – “Profiler” в нейронній мережі

5.3 Повідомлення оператору

Повідомлення можуть впливати в ході некоректної роботи, або помилки в встановленні програми до РС. Ці функції дуже корисні, бо в процесі роботи не повинно бути жодних помилок для правильної роботи програми.

5.4 Висновки за розділом

Розділ містить опис роботи оператора з розробленим програмним продуктом. Надані параметри програмних та апаратних складових. Описані необхідні програмні залежності, попереднього встановлення яких потребує програмний продукт.

У розділі надано опис способу роботи з програмним продуктом, як виконується робота з ним, які дані та у якому вигляді отримує користувач під час роботи. Наведено приклади виводу графічної інформації.

6 ЕКСПЕРИМЕНТИ ТА РЕЗУЛЬТАТИ

6.1 Проведення експериментів

Після модифікації генетичного алгоритму потрібно з'ясувати наскільки готовий модифікований метод може змагатися з іншими методами. В таблиці 6.1 представлені вже існуючі методи та LSTM метод ОГА.

Таблиця 6.1 – Порівняння нейронних мереж

Тип мережі	Помилка у відсотках та час обробки у секундах		
	Нейрони: 40 епохи : 60	Нейрони: 80 епохи : 60	Нейрони: 120 епохи : 60
LSTM	0.53 % 116 с.	0.34% 156 с.	0.18 % 401 с.
LSTM+ Dropout	0.48 % 143 с.	0.28 % 163 с.	0.16 % 406 с.
HUGO	0.51% 138 с.	0.25 % 159 с.	0.14 % 400 с.
RNN	0.58 % 69 с.	0.38 % 61 с.	0.24 % 71 с.
RNN+ Dropout	0.54 % 68 с.	0.35 % 63 с.	0.21 % 81 с.
FFNN	0.51 % 19 с.	0.42 % 19 с.	0.26 % 20 с.
FFNN+ Dropout	0.47 % 24 с.	0.37 % 24 с.	0.23 % 25 с.

У таблиці 6.2 представлено порівняння між типом мережі LSTM та HUGO.

Таблиця 6.2 – Зрівняння LSTM та модифікованого алгоритму

Назва	Точність розрахункова, %	Час, с.
LSTM	82	401
HUGO	86	397

З експериментів зрозуміло, що обрано та модифіковано було саме LSTM нейронну мережу, тому що саме вона найкраще та точніше працює з поставленою задачею. Графічний приклад роботи мереж LSTM та модифікованого методу під назвою HUGO можна побачити на рисунках 6.1 та 6.2.

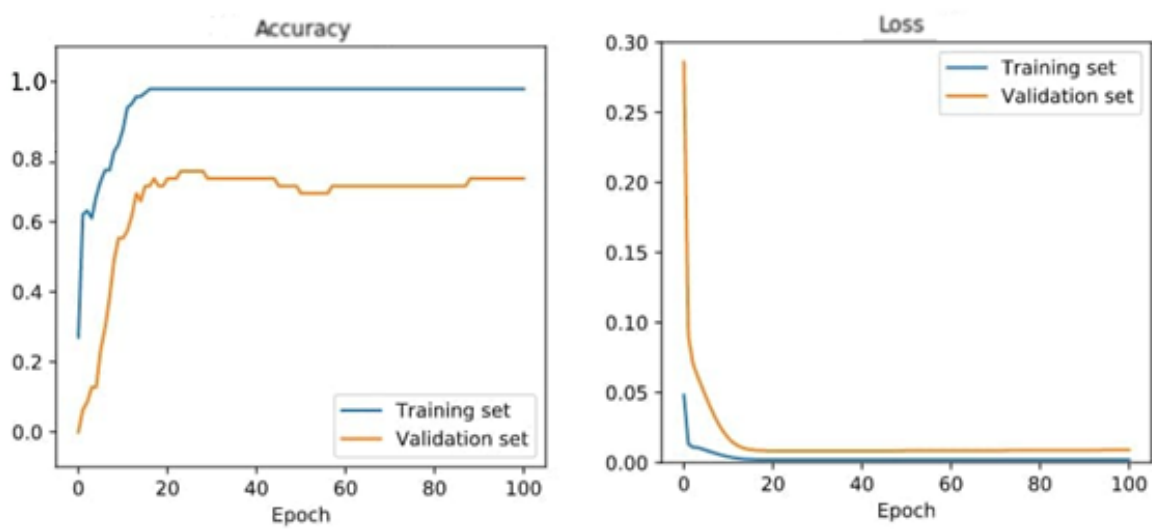


Рисунок 6.1 – Тип мережі LSTM

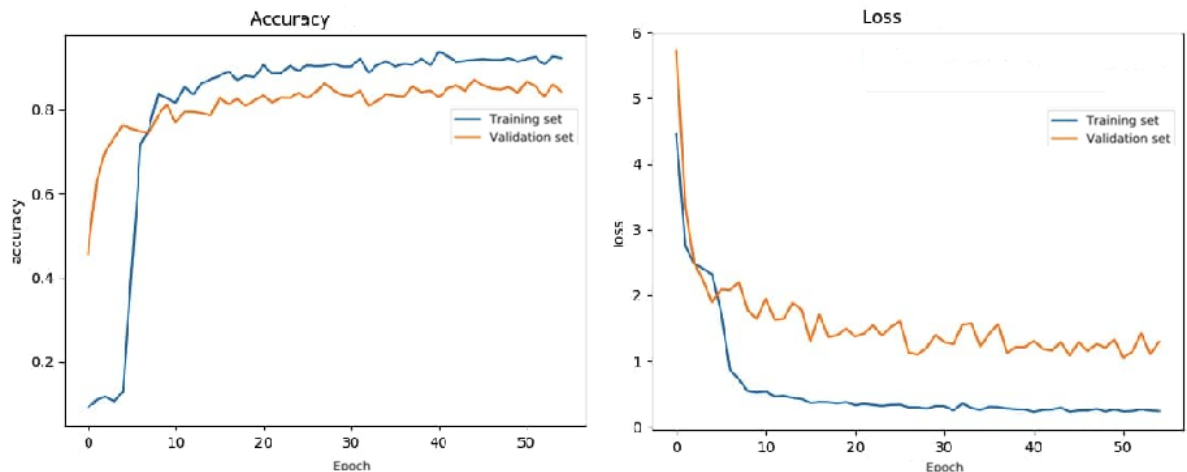


Рисунок 6.2 – Модифікований алгоритм HUGO

На рисунку 6.2 можна побачити модифікований метод HUGO, який краще працює в Training set, аніж в Validation set. Звісно, що вже проводиться додаткова робота з апгрейду модифікованого методу, щоб метод працював краще і в Validation set також.

6.2 Висновки за розділом 6

У розділі наведено методику проведення експериментів. Було зроблено опис та таблицю методів, за якими оцінюється точність навчання нейронних мереж. Також, в таблицях можна побачити час роботи методів. Та основні типи (LSTM та HUGO) наведено в чисельному (табличному виді) та в графічному вигляді також.

ВИСНОВКИ

Під час виконання проєкту було досліджено предметну область, вивчено проблематику поставленої задачі, наведено її актуальність доцільність.

Було проаналізовано стандартні еволюційні методи, що можуть бути застосовані для задачі класифікації та прогнозування. В процесі вирішення поставленої задачі було проаналізовано та обрано середовище програмування. Найкраще з усіх представлених для розробки підходить саме середовище “MatLab”, був аналіз на переваги та недоліки середовища програмування. Також, була обрана мова програмування, саме C++ найкраще підходить для реалізації задачі, бо мова відноситься до машинних мов та найкраще співпрацює з середовищем “MatLab”.

Звісно, що також серед всіх методів прогнозування потрібно було обрати найкращий метод. Найкраще з усіх підходить саме використання нейронних мереж в ході роботи. Також, під час використання нейронної мережі було вирішено реалізувати задачу за допомогою генетичного алгоритму, а саме генетичного алгоритму острівного методу. Перед цим був аналіз серед найвідоміших ГА, які допоможуть реалізувати задачу розробника.

Також, в ході реалізації задачі було проаналізовано багато літературних джерел та електронних ресурсів для підходів до модифікації та застосування програмних засобів для прогнозування фінансування для дітей – сиріт. В результаті чого виконано постановку завдання.

В результаті виконання роботи налагоджений спосіб навчання нейронної мережі за допомогою генетичного алгоритму, для якого запропоновано спосіб острівного методу та зв'язків та рис між ними. Дані, які мають на вході додаються до ГА острівної моделі та через деякий час маємо нові дані на виході. В результаті досліджень та експериментів

запропонованого паралельного генетичного алгоритму острівної моделі можна сказати, що спостерігається поліпшення за точністю.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Прогнозирование. Основные методы подсчета [Электрон. ресурс]. – Режим доступа: <https://dengi.ua/finance/1821714-skol-kog-osudarstvo-platit-za-studentov-bjudzhetnikov>.
2. Проблематика дітей – сиріт в Україні [Електрон. ресурс]. – Режим доступу: <https://www.ukrinform.ua/rubric-society/3311475-soroku-v-ukraini-staut-sirotami-blizko-10-tisac-ditej.html>.
3. Проблематика детей – сирот в Украине [Электрон. ресурс]. – Режим доступа: http://www.ukrstat.gov.ua/operativ/menu/menu_u/virdg.htm?fbclid=IwAR0LvnaXgjB1Hr3idLQnmp6H04NnFLSA91ORdqeWUsfHPZ1jeUS_Qp9K8IE.
4. Вирішувач математичних даних, редактор формул та баз даних [Електрон. ресурс]. – Режим доступу: <https://progsoft.net/ru/software/fixsolver?page=4>.
5. Краткое пособие по GNU Octave [Электрон. ресурс]. – Режим доступа: http://ru.wikibooks.org/wiki/GNU_Octave.
6. Abraham, A. Engineering evolutionary in “Matlab” / A. Abraham, C. Grosan, W. Pedrycz. – Berlin : Springer, 2008. – 444 p.
7. Souolver. Гибрид калькулятора и блокнота [Электрон. ресурс]. – Режим доступа: <https://appstudio.org/reviews/souolver-gibrid-kal-kulyatora-i-bloknota.html>.
8. Мюллер, Д. Программа и среда R для начинающих / Д. Мюллер. – Москва : Диалектика, 2016. – 480 с.
9. Мартин, Р. S. Подробное руководство / Р. Мартин. - Санкт-Петербург : Питер, 2013. – 820 с.
10. Методи прогнозування [Електрон. ресурс]. – Режим доступу: https://pidru4niki.com/12920522/menedzhment/metodi_prognozuvannya.
11. Груздев, В. Прогнозное моделирование и метод деревьев. Решения / В. Груздев. – Москва : ДМК Пресс, 2016. – 276 с.

12. Рашид, Т. Создаем нейронную сеть / Т. Рашид. – Москва : Диалектика, 2019. – 272 с.
13. Кадури́н, А. Обучение и погружение в мир нейронных сетей / А. Кадури́н. – Воронеж : Питер, 2018. – 446 с.
14. Гудфеллоу, Я. Глубокое обучение нейронных сетей / Я. Гудфеллоу. – Москва : ДМК, 2018. – 495 с.
15. Червяков, Н. Применение искусственных нейронных сетей / Н. Червяков. – Москва : Физматлит, 2012. – 269 с.
16. Ковтун, Н. Вибірковий метод. Теорія статистики / Н. Ковтун. – Київ : Знання, 2012. – 399 с.
17. Евдокимов, П. C++ на примерах / П. Евдокимов. – Санкт-Петербург : Наука и Техника, 2016. – 304 с.
18. Japikse, P. Pro C# with .NET and .NET Core / P. Japikse, A. Troelsen. – New York : Apress, 2016. – 1372 p.
19. Язык программирования C# / [А. Хейлсберг, М. Торгерсен, С. Вилтамут, П. Голд]. – Санкт-Петербург : Питер, 2012. – 784 с.
20. Лутц, М. Изучаем Python / М. Лутц. – Харьков : Диалектика, 2020. – 455 с.
21. Дьяконов, В. “MatLab”. Полный самоучитель / В. Дьяконов. – Москва : ДМК, 2012. – 768 с.
22. Ревинская, О. Основы программирования в “MatLab” / О. Ревинская. – Санкт – Петербург : БХВ, 2016. – 207 с.
23. Герберт, Ш. Книга C++ та її базовий курс / Ш. Герберт. – Харків : Диалектика, 2019. – 2019 с.
24. Аггарвал, Ч. Нейронні мережі та їх навчання з використанням / Ч. Аггарвал. – Київ : Знання, 2017. – 752 с.
25. Chaikin, S. Neural networks and Dlib tools in work / S. Chaikin. – New York : Apress, 2019. – 401 p.
26. Sedgwick, R. Algorithms on C++ / R. Sedgwick. – New York : Williams, 2019. – 1056 p.

27. Pattanajak, S. Deep learning and Tensor Flow / S. Pattanajak. – New York : Williams, 2017. – 450 p.
28. Сирота, О. Методи та алгоритми аналізу даних. Їх моделювання в “MatLab” / О. Сирота. – Санкт – Петербург : БХВ , 2016. – 384 с.
29. Потемкин, В. Нейронные сети в работе с “MatLab” / В. Потемкин. – Москва : ДМК, 2019. – 496 с.
30. Brownlee, J. Long Short-Term Memory Networks / J. Brownlee. – New York : Williams, 2018. – 246 p.
31. Костадинов, С. Рекуррентные нейронные сети / С. Костадинов. – Москва : ДМК, 2018. – 320 p.
32. Локашо, Н. Создание алгоритмов для обучения / Н. Локашо. – Москва : Фербер, 2017. – 304 с.
33. Хайкин, С. Нейронные сети. Полный курс / С. Хайкин. – Москва : Диалектика, 2018. – 1104 с.
34. Островной генетический алгоритм с динамическим распределением вероятностей выбора генетических операторов [Электрон. ресурс]. – Режим доступа: <https://ostrovnoy-geneticheskiy-algorithm-s-dinamicheskim-raspredeleniem-veroyatnostey-vybora-geneticheskikh-operatorov>.
35. Гладков, В.А. Генетические алгоритмы / В. Гладков. – Москва : ДМК, 2010. – 368 с.
36. Уилер, Т. Алгоритмы оптимизации / Т. Уилер. – Москва : ДМК, 2016. – 528 с.

ДОДАТОК А
Технічне завдання

Вступ

У роботі розглядається галузь статистики прогнозування. Програмний продукт пов'язаний з прогнозуванням фінансування для дітей – сиріт, які поступають до ЗВО, або у професійні коледжі на подальші роки.

А.1 Підстави для виконання роботи

Підставою для виконання роботи є завдання на магістерську дипломну роботу «Соціалізація сиріт у вищих навчальних закладах України. Соціотехнологічний аспект» відповідно до наказу №435 по Національному університету «Запорізька політехніка» від 05.11.2021 року.

А.2 Призначення для розробки методу

Призначенням розробки є програмний продукт, який би зміг реалізувати прогнозування фінансування для дітей – сиріт, які вступили та навчаються в ЗВО. Еволюційний алгоритм повинен реалізовувати представлені методи і дозволяти розв'язувати задачі аналізу даних для вирішення цілей, зокрема для підрахунку фінансування.

А.3 Основні вимоги до програми, що розроблюється

А.3.1 Вимоги до функціональних характеристик

Розроблюване програмне забезпечення повинно відповідати таким вимогам:

- відповідність темі проєкту;
- коректність роботи;
- можливість завантаження даних;
- можливість роботи у Online режимі;

- можливість виводу вихідних даних у графічному та текстовому вигляді.

А.3.2 Вимоги до коректної роботи

Програма повинна працювати в коректній стадії. Робота в середовищі не повинна мати глюків та незапланованих вилітів з програми. Вирішення цих проблем може тільки бути в правильності встановлення програми до РС. Не рекомендується завантаження окремих пакетів.

А.3.3 Умови для роботи

Для коректної роботи з програмою користувач повинен володіти достатніми знаннями для коректного вводу інформації та розбиратись в архітектурі програми. Для цього користувачеві потрібно ввести дані в програму. Робота здійснюється відповідно до програмної документації на розроблений метод, що відповідає стандартам, а також містить корисну інформацію, необхідну для освоєння.

А.3.4 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення написане мовою C++. Розробка і тестування проводилися на обладнанні з такими конфігураціями:

- процесор Intel Core i3 – 4005U, 1.7 GHz;
- 8 GB оперативної пам'яті;
- операційна система Windows 10 (x-64);
- мова програмування C++;
- нейромережева бібліотека Caffe;
- бібліотека машинного навчання <iostream>;

– виклик двох бібліотек для роботи в “MatLab” – “MatLabEngine.hpp” та “MatLabDataArray.hpp”.

A.3.5 Вимоги до програмної документації

Програма повинна поставлятися з технічним завданням та керівництвами програміста та користувача.

A.4 Стадії та етапи розробки

Програмний продукт повинен бути виконаний у 12-тижневий строк з моменту отримання завдання. Проєкт складається з таких етапів:

- аналіз предметної області;
- розробка технічного завдання;
- з’ясування моделі архітектури;
- реалізація програмного продукту та методу;
- тестування на всіх етапів та експериментальне дослідження з графічним відображенням.

A.5 Порядок контролю та приймання

Магістерська робота допускається до захисту керівником, консультантом та нормо контролером, затверджується завідувачем кафедри та захищається у присутності членів екзаменаційної комісії.

ДОДАТОК Б**Код програми**

Б.1 Файл «Main_mb»

```

#include <iostream>

#include <cstdlib>

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

using namespace std;

const int max_num_inp = 20;

const int max_num_out = 10;

const int numb_vect = 200;

const double temp = 0.75;

const double bound_error = 5.0e-4;

const int max_iterat = 1000;

const double init_weight = 5.0;

const double bias = 1.0;

typedef double matrix[max_num_inp][max_num_out + 1]; typedef double
inp_vec[max_num_inp];

typedef double out_vec[max_num_out];

struct perceptron{

int numb_inp;

int numb_out

inp_vec inp

out_vec out

matrix w;

} perc;

struct inp_out_vec{

inp_vec x;

out_vec y;

};

```

```

struct images{

int numb_images

inp_out_vec images[numb_vect];

}img;


double Vbound_error,

Vtemp,

Vdelta;

int Vmax_iterat;

double sigmoid(double Vtemp){

return (1.0/(1.0+exp(-Vtemp)));

}

void initial(){

srand((unsigned)time(NULL));

int i, j;

perc.numb_inp = 10;

perc.numb_out = 1;

for(i = 0; i <= perc.numb_inp; i++){

for(j = 1; j <= perc.numb_out; j++){

perc.w[i][j] = init_weight * ((double)rand()/RAND_MAX - 0.5);

}

}

}

void gener(){

srand((unsigned)time(NULL));

int i, j, k;

img.numb_images = 200;

for(i = 1; i <= img.numb_images; i++){

k = 0;

for(j = 1; j <= perc.numb_inp; j++){

```

```

img.images[i].x[j] = rand() > RAND_MAX / 2?1:0;

if(img.images[i].x[j] > 0) k = k + 1;

}

if(k > (perc.numb_inp - k)) img.images[i].y[1] = 1;
else img.images[i].y[1] = 0;

//cout << " " << img.images[i].y[1];

}

}

int main(){

setlocale(0, "");

double Vcorrect;      int i, j, k;

printf("-----
-----");

srand((unsigned)time(NULL));

unsigned int start_time = clock();

initial();

gener();

Vmax_iterat = 0;

do{

Vbound_error = 0.0;

for (i = 1; i <= img.numb_images; i++){

for (j = 1; j <= perc.numb_inp; j++){

perc.inp[j] = img.images[i].x[j];

}

for(k = 1; k <= perc.numb_out; k++){

Vtemp = bias * perc.w[0][k];                                for (j = 1; j <=
perc.numb_inp; j++){

Vtemp = Vtemp + perc.inp[j] * perc.w[j][k];                                }

perc.out[k] = sigmoid(Vtemp);

Vdelta = img.images[i].y[k] - perc.out[k];

Vbound_error = Vbound_error + 0.5 * Vdelta * Vdelta;

```

```

perc.w[0][k] = perc.w[0][k] + temp * bias * Vdelta;

for (j = 1; j <= perc.numb_inp; j++){

perc.w[j][k] = perc.w[j][k] + temp * perc.inp[j] * Vdelta;
}

}

}

Vmax_iterat = Vmax_iterat + 1;

}

while ((Vmax_iterat < max_iterat) && (Vbound_error > bound_error));

printf("\n| Кількість ітерацій = %4d
|", Vmax_iterat);

printf("\n| Кількість помилок = %6.5lf
|\n", Vbound_error);

printf("-----
-----\n");

printf("|
                                МОДЕЛЬ НЕЙРОННОЇ МЕРЕЖІ

|\n");

printf("-----
-----\n");

printf("| 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 | Відповідь| Вірна
відповідь |\n");

printf("-----
-----\n");

for (i = 1; i <= numb_vect/10; i++){

k = 0;

for (j = 1; j <= perc.numb_inp; j++){

perc.inp[j] = rand() > RAND_MAX / 2?1:0;

if (perc.inp[j] > 0)

k = k + 1;

}

if (k > (perc.numb_inp - k)) Vcorrect = 1.0;

else Vcorrect = 0.0;

for (k = 1; k <= perc.numb_out; k++){

Vtemp = bias * perc.w[0][k];

```

```

for (j = 1; j <= perc.numb_inp; j++){
Vtemp = Vtemp + perc.inp[j] * perc.w[j][k];
}

perc.out[k] = sigmoid(Vtemp);
}

for (j = 1; j <= perc.numb_inp; j++)
printf("|  %d ", (int)perc.inp[j]);
printf("|%6.2lf |", perc.out[1]);
printf("      %d      |\n", (int)Vcorrect);
}

printf("-----\n");

unsigned int end_time = clock();

unsigned int search_time = end_time - start_time;

printf(" Початковий час = %6.3lfc
|\n", start_time/1000.0);

printf(" Остаточний час = %6.3lfc
|\n", end_time/1000.0);

printf(" Час роботи програми = %6.3lfc
|\n", search_time/1000.0);

printf("-----\n");

system("pause");
}

```

Б.2 Файл «Easy_GA»

```

class Prim{
protected:
    Type _type;
    int _x;
    int _y;
    int _healthPointChange    Color _color;

```

```

public:

    Type GetType() ;

    void SetType(Type) ;

    int GetX() ;

    int GetY() ;

    void SetX(int) ;

    void SetY(int) ;

    Prim() ;

    Prim(Type) ;

    ~Prim() ;

    int getHPC() ;

    void setHPC(int) ;

    Color& getColor() ;

    void setColor(float, float, float) ;

};

class CreationC : public Prim
{

    enum direction { lU, up, uR, right, rD, down, dL, left };

    AreaCLA* _world;

    direction _head;

    short int _hp;

    vector<UNI> _commandList;

    UNI _itForCommand;

    Prim*** _area;

    boost::signal <void()> _eatingFood

    boost::signal <void()> _eatingPoison

    void _Step(UNI) ;

    bool _See(UNI) ;

    void _Roll(UNI) ;

    void _AddSlots() ;

    void _Eat(UNI) ;

```

```

    bool _isNextCellClose(UNI);

public:
    void Mutation();

    bool IsAlive();

    void Execute();

    short int GetHP();

    CreationC(vector<UNI>& gens, AreaCLA*, int x, int y);

    CreationC();

    vector<UNI>& GetCommandList();

    ~CreationC();

};

class AreaCLA
{
    Prim*** _map;

    int _heigh;

    int _width;

    UNI _countOfFood;

    UNI _countOfPoison;

    UNI _FPCount;

    CreationC** _crtns;

    int _crtnsCount;

    int _maxCrtnsCount;

    int _minCrtnsCount;

    void _cicle();

    void _refresh();

    void _Reborn();

    void _delCrt(int, int, int);

public:
    AreaCLA()

    ~AreaCLA();

    void SetFood(COORD coord);

```

```

void SetPoison(COORD coord);

void SetWall(COORD coord);

void SetVoid(COORD coord);

void Start();

void minFood();

void minPoison();

};

void AreaCLA::_cicle()
{
    for (int i = 0; i < _crtnsCount; i++)
    {
        if (!_crtns[i]) continue;
        _crtns[i]->Execute();
        if (!_crtns[i]->IsAlive())
            delCrt(i, _crtns[i]->GetX(), _crtns[i]->GetY());
        if (_crtnsCount == this->_minCrtnsCount)
            return;
        if (_countOfCtrnsChenged)
        {
            i--;
            _countOfCtrnsChenged = 0;
            continue;
        }
    }
}

```

Б.3 Файл «LSTM»

```

sequence_length = 30

x_data = []
y_data = []

for i in range(0, len(text) - sequence_length, 1):

```

```

x_sequence = text[i:i + sequence_length]

y_label = text[i + sequence_length]

x_data.append([char2idx[char] for char in x_sequence])

y_data.append(char2idx[y_label])

X = np.reshape(x_data, (data_length, sequence_length, 1))

X = X/float(vocab_length)

y = np_utils.to_categorical(y_data)

# Split into training and testing set, shuffle data

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2,
shuffle=False)

# Shuffle testing set

X_test, y_test = shuffle(X_test, y_test, random_state=0)

model = Sequential()

model.add(LSTM(256, input_shape=(X.shape[1], X.shape[2]),
return_sequences=True, recurrent_initializer='glorot_uniform',
recurrent_dropout=0.3))

model.add(LSTM(256, return_sequences=True,
recurrent_initializer='glorot_uniform', recurrent_dropout=0.3))

model.add(LSTM(256, recurrent_initializer='glorot_uniform',
recurrent_dropout=0.3))

model.add(Dropout(0.2))

model.add(Dense(y.shape[1], activation='softmax'))

```

ДОДАТОК В
Слайди презентації

Національний університет “Запорізька політехніка”
кафедра програмних засобів

Дипломна робота магістра
на тему

**ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
ПРОГНОЗУВАННЯ ФІНАНСУВАННЯ СОЦІАЛІЗАЦІЇ СИРІТ У
ЗАКЛАДАХ ВИЩОЇ ОСВІТИ**

Виконав
Ст. гр. КНТ-120м

Злиденний А.О.

Керівник
д.т.н., доцент

Олійник А.О.

1

Рисунок В.1 – Слайд №1

Об’єкт, предмет та мета дослідження

- Об’єкт дослідження – процес прогнозування витрат на основі інтелектуальних технологій.
- Предмет дослідження – методи для прогнозування фінансування для дітей – сиріт, які вступили у ЗВО України.
- Мета роботи – дослідження та модифікація еволюційних методів прогнозування.

2

Рисунок В.2 – Слайд №2

Постановка задачі

- Проблема фінансування дітей – сиріт, які вступають та навчаються до ЗВО дуже велика для України на сьогоднішній день. Кожен рік до учбових закладів поступає велика кількість абітурієнтів, задачею аналітиків та статистиків – спрогнозувати фінансування.
- Необхідно проаналізувати загальновідомі методи та розробити або модифікувати метод для прогнозування фінансування дітей – сиріт, які вступають до лав ЗВО та навчаються в ньому.

3

Рисунок В.3 – Слайд №3

Основні методи аналізу даних

Таблиця 1 – Методи аналізу даних

Метод	Сутність	Основні етапи побудови	Переваги	Використання (галузь)
Дерева рішень	Модель прогнозує значення цільової змінної на основі декількох змінних на вході.	Вибрати атрибут для здійснення розбиття в даному вузлі; визначити критерій зупинки навчання; вибрати метод відсікання гілок; оцінити точність побудованого дерева.	Можливість пов'язати ставлення цілі з діями, що підлягають реалізації	В галузі статистики та аналізу даних для прогнозних моделей
Нейронна мережа	Працюють як з кількісними, так і з якісними характеристиками, певним чином підготовленими, дають змогу встановити зв'язки.	Дані поступають на нейрон; обчислюється їх вага; результати обчислень передаються на наступний нейрон; процес повторюється.	Універсальність, дешевизна і низька швидкість навчання	Економіка, медицина, наукові дослідження, інформаційні технології
Вибірковий метод	З значенням характеристик вибіркової сукупності роблять певні висновки	Визначення основи вибірки; побудова дизайну вибірки; обчислення обсягу вибірки; відбір одиниць для безпосереднього обстеження.	Розповсюдити висновки отримані на основі вивчення частини вибірки	Соціальні дослідження, економіка
Логічна регресія	При запровадженні значення може знаходити застосування у класи	Логістичну регресію можна представити у вигляді одношарової нейронної мережі	Простота розрахунків і прийняття на точність прогнозу	Використовується як готові реалізації для вирішення прикладних задач

Таблиця 2 – Вибір інструменту для передбачення

Метод	Час при навчанні	Залежність вхідних та вихідних даних	Точність прогнозу
Дерева рішень	Дуже швидкий процес навчання	Параметри дерева необхідно налаштувати в залежності від вхідних даних	Мак велику точність, але поступається нейронній мережі
Нейронна мережа	Залежить від процесу (з вчителем, або без вчителя)	Здатна вирішувати завдання, в яких невідомі закономірності розвитку ситуації та залежності між вхідними та вихідними даними	Дуже велика точність прогнозу завдяки адаптації до середовища
Вибірковий метод	Значно програє дереву рішень та нейронній мережі	Спостереження не всіх елементів сукупності що вивчаються, а лише вибіраних	Залежить від вибірки

4

Рисунок В.4 – Слайд №4

Порівняння мов програмування

Таблиця 3 – Вибір мови програмування

Критерії порівняння	Мова програмування			
	C++	C#	Python	Matlab
Адаптація до PC та його середовища	+	+	+	-
Оновлення пакетів	-	+	+	-
Безкоштовність	+	+	+	-
Платформонезалежність	+	+	+	+
Наявність бібліотек для навчання	+	+	+	+
Продуктивність в процесі	+	-	+	-
Кросплатформність	-	+	+	-
Швидкість компілювання коду	+	+	+	-
Робота з ядром комп'ютера	+	-	+	+
Співпраця з іншими мовами	+	+	+	+

З таблиці 3 можна побачити, що проводився аналіз мов програмування, які здатні працювати з нейронними мережами для реалізації поставленої задачі.

5

Рисунок В.5 – Слайд №5

Основні рішення щодо реалізації компонентів системи

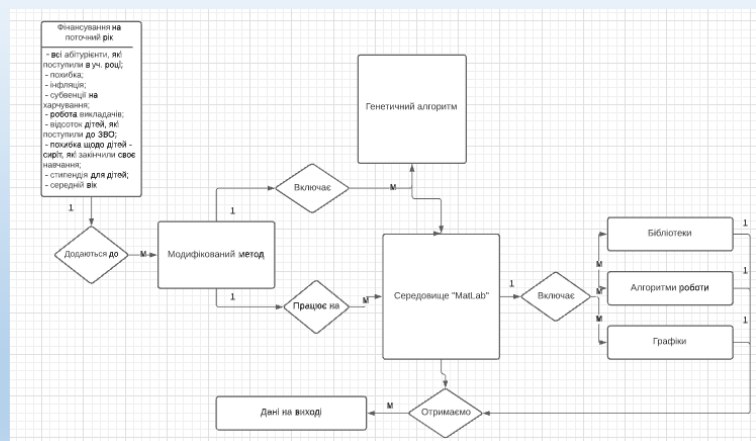


Рисунок 1 – ERP діаграма предметної області

6

Рисунок В.6 – Слайд №6

Огляд нейронних мереж для прогнозування

Таблиця 4 – Огляд та вибір типу мережі для роботи

Тип мережі	Помилка у відсотках та час обробки у секундах		
	Нейрони: 40 епохи : 60	Нейрони: 80 епохи : 60	Нейрони: 120 епохи : 60
LSTM	0.53 % 116 с.	0.34 % 156 с.	0.18 % 401 с.
LSTM+ Dropout	0.48 % 143 с.	0.28 % 163 с.	0.16 % 406 с.
RNN	0.58 % 69 с.	0.38 % 61 с.	0.24 % 71 с.
RNN+ Dropout	0.54 % 68 с.	0.35 % 63 с.	0.21 % 81 с.
FFNN	0.51 % 19 с.	0.42 % 19 с.	0.26 % 20 с.
FFNN+ Dropout	0.47 % 24 с.	0.37 % 24 с.	0.23 % 25 с.

З таблиці показано, що при вибірці великого розміру найкраще для реалізації заданої задачі підходить тип мережі LSTM.

Після чого було прийнято модифікувати саме LSTM нейронну мережу модифікованим алгоритмом.

Новий модифікований метод має назву “HUGO”.

7

Рисунок В.7 – Слайд №7

Порівняння та вибір ГА

Генетичний алгоритм	Час пошуку вирішення		Запит до функції		Кількість популяцій	
	Ср. Сек.	Дис. с ²	Ср.	Дис.	Ср.	Дис.
Стандартний ГА	29.9	592.4	1267039	1.017*1012	74.9	3584
Га + РВКМ (регуляція кросингвера та мутації)	26.4	670.3	1114035	9.152*1011	89.4	5600
Га + КМ (в поколіннях)	15.6	284	780658	6.156*1011	237	58600
ГА + ДСП (динамічна самоорганізація операторів)	54.2	1360	1925972	2.469*1012	68.4	1226
Острівний ГА (2 острова)	12	17	560247	2.113*1010	-	-
Острівний ГА (3 острова)	19	37	859917	5.173*1010	-	-
Острівний ГА (4 острова)	23	48	113007	6.433*1010	-	-
Острівний ГА (всі острова)	31	81	135768	9.944*1010	-	-

Рисунок 2 – Огляд ГА та вибір відповідного до задачі

Генетичний алгоритм острівного метода (ОГА) дозволяє в 3 рази швидше знаходити рішення, порівняно з ПГА (простим генетичний алгоритмом).

Слід також відзначити високу надійність рішень, отримуваних острівним методом – дисперсія часу пошуку рішення за допомогою острівного метода менше, ніж у решті ГА.

8

Рисунок В.8 – Слайд №8

Модифікація генетичного алгоритму

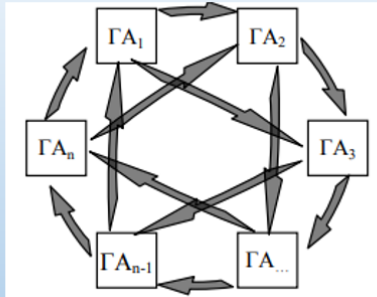


Рисунок 3 – Острівна модель
ГА

$$\bar{F}_{is}(\bar{x}_c, \bar{x}_r) = \sum_{k=1}^{Nk} \mu_k (\bar{Y}_k(\bar{x}_c, \bar{x}_r))^2$$

$\bar{F}_{is}$ - функція модифікованої острівної моделі;

Nk - кількість критерій якості;

μ_k - вага критерій якості;

$\bar{Y}_k$ - нормоване значення критерій якості;

$\bar{x}_c, \bar{x}_r$ - вектори шляху.

$$\bar{F}_{glob}(\bar{x}_c, \bar{x}_r) = \sum_{r=1}^{Nk} (\bar{F}_{is}(\bar{x}_c, \bar{x}_r))^2$$

9

Рисунок В.9 – Слайд №9

Експерименти та результати

Таблиця 5 – Зрівняння між LSTM та
HUGO

Назва	Точність розрахункова, %	Час, с.
LSTM	82	401
HUGO	86	397

З експериментів зрозуміло, що обрано та модифіковано було саме LSTM нейронну мережу, тому що саме вона найкраще та точніше працює з поставленою задачею.

Також, було модифіковано LSTM нейронну мережу під назвою "HUGO". Зрівняння можна побачити на рисунку 5.

10

Рисунок В.10 – Слайд №10

Результати

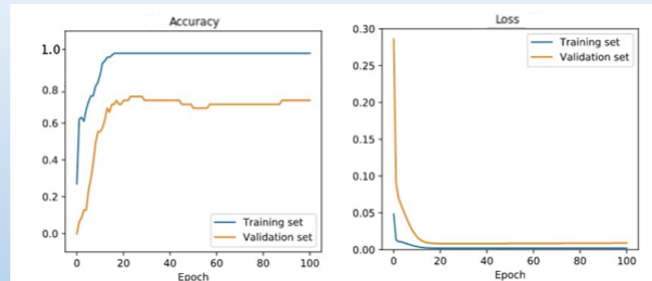


Рисунок 4 – LSTM нейронна мережа

11

Рисунок В.11 – Слайд №11

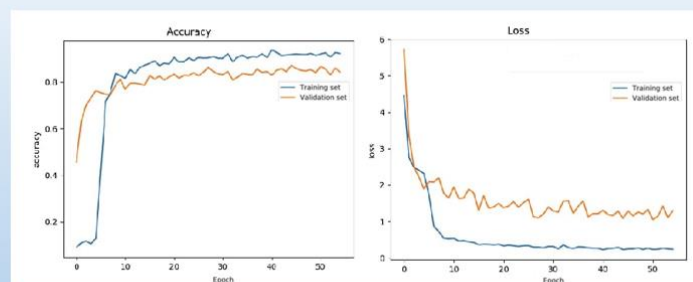


Рисунок 5 – Модифікація під назвою HUGO

12

Рисунок В.12 – Слайд №12

Висновки

Під час виконання дипломної роботи було:

- проаналізовано предметну область та методи, які можна використовувати для прогнозування фінансування для дітей – сиріт, які навчаються в ЗВО;
- наведено опис основних рішень, щодо реалізації системи;
- проаналізовано сучасні літературні джерел, щодо підходів до модифікації та застосування програмних засобів для прогнозування фінансування;
- модифіковано нейронну мережу шляхом генетичного алгоритму (острівної моделі).

В результаті виконання роботи запропоновано спосіб навчання нейронної мережі за допомогою ГА, для якого запропоновано спосіб острівного методу та зв'язків між ними.



13

Рисунок В.13 – Слайд №13