НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
(повна назва інституту/факультету)
кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ
(повна назва кафедри)

«До захисту допущено»

Завідувач кафедри БМК

_____ __Євген НАСТЕНКО_
(підпис) (ініціали, прізвище)

"_*20*_" __*червня*__2022 р.

ІНДИВІДУАЛЬНИЙ ДОСЛІДНИЙ ПРОЕКТ

на здобуття ступеня бакалавра

за освітньо-професійною програмою ***Комп'ютерні технології в біології та медицині***
зі спеціальності ***122 Комп'ютерні науки***
(код і назва)

на тему: **Логістичний ліс самоорганізованих дерев за критерієм якості прогнозу класифікації**

Виконала: студент **IV** курсу, групи ____БС-81_____
(шифр групи)

ГЛАДКИЙ ЯРОСЛАВ ВОЛОДИМИРОВИЧ _____
(прізвище, ім'я, по батькові) (підпис)

Керівник ІДП **доцент, к.т.н. Володимир ПАВЛОВ** _____

Консультант ІДП **ас. каф. БМК Олександр ДАВИДЬКО** _____

Нормоконтроль **ст.викл. каф. БМК Галина КОРНІЄНКО** _____

Засвідчую, що у звіті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
(повна назва)

Кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ
(повна назва)

Рівень вищої освіти – перший (бакалаврський)
спеціальність **122 Комп'ютерні науки**
спеціалізація **Комп'ютерні технології в біології та медицині**
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри БМК

_____ Євген НАСТЕНКО
(підпис) (ініціали, прізвище)

«_*18*» _*червня*_ 2022 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на індивідуальний дослідний проект (ІДП) студента

ГЛАДКОГО ЯРОСЛАВА ВОЛОДИМИРОВИЧА
(прізвище, ім'я, по батькові)

1. Тема ІДП ***Логістичний ліс самоорганізованих дерев за критерієм якості прогнозу класифікації***

Керівник ІДП
Павлов Володимир Анатолійович к.т.н, доцент, доцент каф. БМК
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені засіданням кафедри БМК від «*18*» _*квітня* 2022 р. №_17__

2. Термін подання студентом звіту ***20 червня 2022 року***

3. Вихідні дані до роботи ***База даних КТ зображень легенів пацієнтів ДУ «Національний інститут фтизіатрії і пульмонології ім. Ф.Г. Яновського НАМН України»***

4. Зміст роботи ***Анотації (на двох мовах); Вступ ;Оогляд літературних джерел, теоретична частина (основні поняття з теми побудови самоорганізованих дерев та логістичного лісу, формули що описують класифікатор на основі дерев прийняття рішень; існуючі методи згорткових нейронних мереж); аналітична чатина (аналіз існуючого***

методу випадкового лісу та проведення порівняння між ним (переваги /недоліки) тощо); практична частина.

Загальні висновки; список використаних джерел.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо)

11 рисунків, 5 таблиць, 1 презентацію.

6. Орієнтовний перелік публікацій ***Davydko, O., Hladkyi, Y., Linnik, M., Nosovets, O., Pavlov, V., & Nastenko, I. (2021, September). Hybrid Classifiers Based on CNN, LSOF, GMDH in COVID-19 Pneumonic Lesions Types Classification Task. In 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT) (Vol. 1, pp. 380-384). IEEE.***

7. Консультант з розділів роботи ***Давидько О.Б., ас. каф. БМК***

8. Дата видачі завдання ***25 квітня 2022р.***

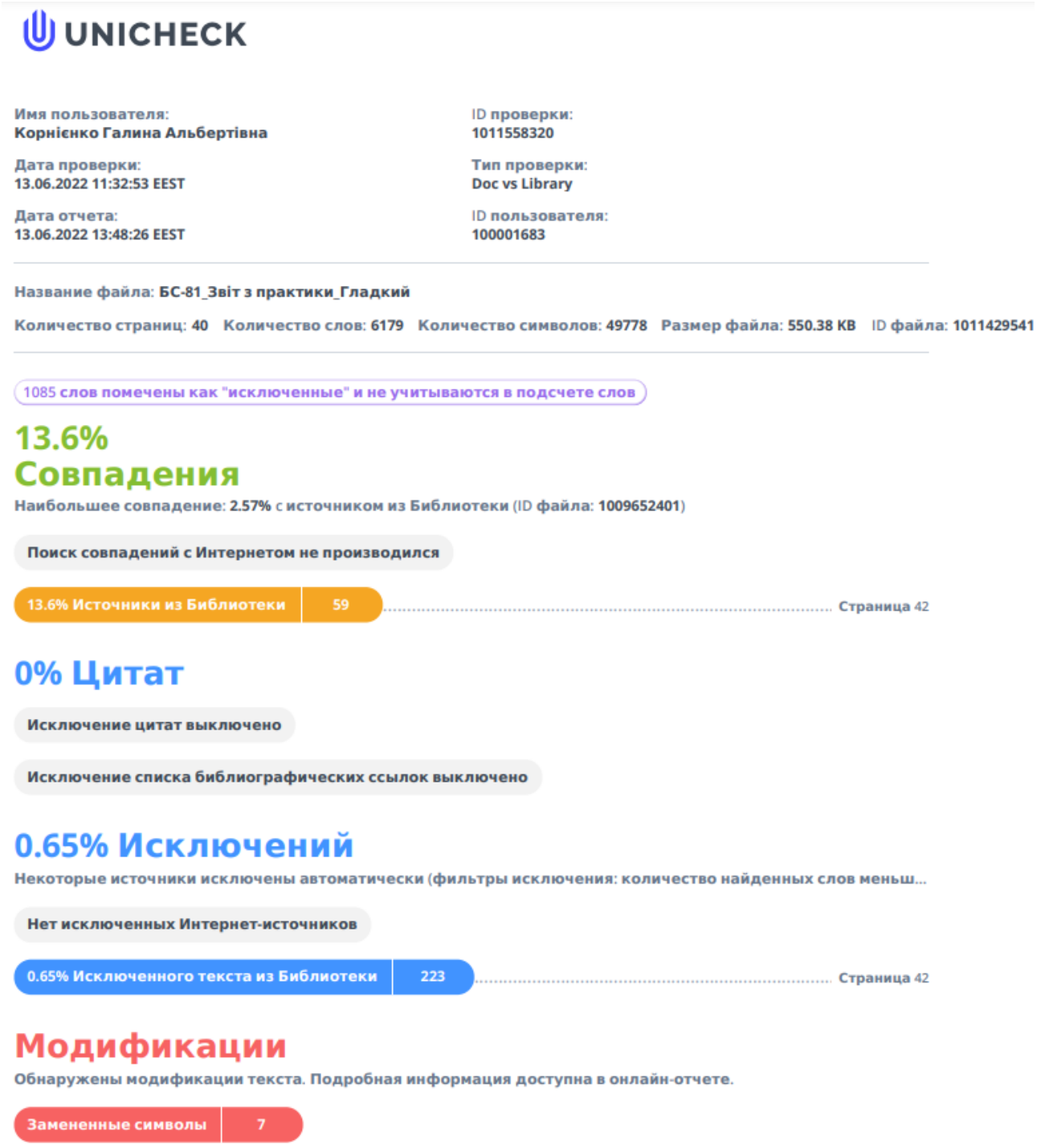
Календарний план

№ з/п	Назва етапів виконання ІДП	Термін виконання етапів ІДП	Примітка
1	Отримання завдання на переддипломну практику	25.04.2022р.	*виконано*
2	Захист практики	14-18.06.2022р.	*виконано*
3	Ухвалення остаточної теми ІДП	18.06.2022р.	*виконано*
4	Доопрацювати документи на комплексний атестаційний екзамен (доповідь по темі ІДП)	19-20.06.2022р.	*виконано*
5	Подання пакету документів по ІДП	20.06.2022р.	*виконано*
6	Захист ІДП	20-25.06.2022р.	*виконано*

Студент _____ ***Ярослав ГЛАДКИЙ***
(підпис)

Керівник за темою практики _____ ***Володимир ПАВЛОВ***
(підпис)

ЗВІТ З UNICHEK



UNICHECK

Имя пользователя:
Корнієнко Галина Альбертівна

ID проверки:
1011558320

Дата проверки:
13.06.2022 11:32:53 EEST

Тип проверки:
Doc vs Library

Дата отчета:
13.06.2022 13:48:26 EEST

ID пользователя:
100001683

Название файла: **БС-81_Звіт з практики_Гладкий**

Количество страниц: **40** Количество слов: **6179** Количество символов: **49778** Размер файла: **550.38 KB** ID файла: **1011429541**

1085 слов помечены как "исключенные" и не учитываются в подсчете слов

13.6%
Совпадения

Наибольшее совпадение: **2.57%** с источником из Библиотеки (ID файла: **1009652401**)

Поиск совпадений с Интернетом не производился

13.6% Источники из Библиотеки 59 ..... Страница 42

0% Цитат

Исключение цитат выключено

Исключение списка библиографических ссылок выключено

0.65% Исключений

Некоторые источники исключены автоматически (фильтры исключения: количество найденных слов меньш...

Нет исключенных Интернет-источников

0.65% Исключенного текста из Библиотеки 223 ..... Страница 42

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы 7

АНОТАЦІЯ

Індивідуальний дослідний проект (надалі – ІДП) за темою «*Логістичний ліс самоорганізованих дерев за критерієм якості прогнозу класифікації»* виконаний студентом кафедри біомедичної кібернетики ФБМІ *Гладким Ярославом Володимировичем* зі спеціальності *122 «Комп'ютерні науки»* за освітньо-професійною програмою *«Комп'ютерні технології в біології та медицині»* та складається зі: вступу; 4 розділів (літературного огляду, теоретичного, аналітичного та практичного); висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 11 джерел. Загальний обсяг роботи – 43 сторінки.

Актуальність теми ІДП: пандемія COVID-19 стала однією з найяскравіших глобальних реакцій. Однією з причин цього є безперервне зростання людської популяції. Тому ефективні діагностичні та аналітичні інструменти для боротьби з наслідками інфекції сьогодні дійсно потрібні. Проблема в тому, що, за даними ВООЗ, навіть найпопулярніші ПЛР-тести не можуть дати достовірних результатів і їх точність становить близько 66%. У той же час комп'ютерна томографія дозволяє виявити тип ураження легень з набагато вищим ступенем достовірності – до 99%. Крім того, КТ дозволяє лікарям отримати структуру ураження, яка може бути основною важливою інформацією про перебіг інфекції. В роботі розглядається застосування удосконаленого алгоритму самоорганізованого лісу для вирішення задачі класифікації типів уражень легень.

Мета ІДП: Вирішення задачі підвищення ефективності одного з найбільш розповсюджених і часто використовуваних алгоритмів класифікації з класу Random Forest.

Завдання ІДП:

• Розробка алгоритму та програмної реалізації лісу самоорганізованих дерев.

• Вирішення задачі класифікації типу уражень легень при захворюванні на

Covid-19.

Публікації: за результатами ІДП була опублікована 1 наукова стаття:

1. Davydko, O., Hladkyi, Y., Linnik, M., Nosovets, O., Pavlov, V., & Nastenko, I. (2021, September). Hybrid Classifiers Based on CNN, LSOF, GMDH in COVID-19 Pneumonic Lesions Types Classification Task. In 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT) (Vol. 1, pp. 380-384). IEEE.

Ключові слова: Random Forest, алгоритм самоорганізації, МГУА, логістична регресія, COVID-19, КТ легенів, ROI.

ABSTRACT

An individual research project (hereinafter – IRP) on the topic *" Logistic forest of self-organized trees with the criterion of quality of the forecast forecast"* was performed by a student of the Department of Biomedical Cybernetics FBME *Yaroslav Hladkyi* in specialty 122 *"Computer Science"* in the educational and professional program *"Computer Technology in Biology and Medicine"* and consists of: introduction; 4 sections (literature review, theoretical, analytical and practical); conclusions to each of these sections; general conclusions; a list of used sources, which includes 11 sources. The total volume of work is – 43 pages.

The urgency of the topic of IRP: the COVID-19 pandemic was one of the most powerful global reactions. One of the reasons for this is the continuous growth of the human population. Therefore, effective diagnostic and analytical tools to combat the effects of infection are really needed today. The problem is that, according to the WHO, even the most popular PCR tests can not give reliable results and their accuracy is about 66%. At the same time, computed tomography can detect the type of lung damage with a much higher degree of reliability - up to 99%. In addition, CT allows doctors to obtain the structure of the lesion, which can be the main important information about the course of infection. The paper considers the application of an advanced algorithm of self-organized forest to solve the problem of classification of types of lung lesions.

The purpose of IRP: Solving the problem of improving the efficiency of one of the most common and frequently used classification algorithms of the Random Forest class.

Tasks of IRP:

• Development of algorithm and software implementation of self-organized trees.

• Solving the problem of classifying the type of lung lesions in Covid-19.

Publications: according to the results of IRP 1 scientific article was published:

1. Davydko, O., Hladkyi, Y., Linnik, M., Nosovets, O., Pavlov, V., & Nastenko, I. (2021, September). Hybrid Classifiers Based on CNN, LSOF, GMDH in

COVID-19 Pneumonic Lesions Types Classification Task. In 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT) (Vol. 1, pp. 380-384). IEEE.

Key words: Random Forest, self-organization algorithm, GMDH, logistic regression, COVID-19, lung CT, ROI.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

пкс – піксель

МГУА – метод групового урахування аргументів

GLCM – Gray-Level Co-occurrence Matrix

ROI – region of interest (область інтересу)

ВСТУП

Тема роботи: Розробка алгоритму логістичного лісу самоорганізованих дерев за критеріям якості прогнозу класифікації.

Актуальність: пандемія COVID-19 стала однією з найяскравіших глобальних реакцій. Однією з причин цього є безперервне зростання людської популяції. Тому ефективні діагностичні та аналітичні інструменти для боротьби з наслідками інфекції сьогодні дійсно потрібні. Проблема в тому, що, за даними ВООЗ, навіть найпопулярніші ПЛР-тести не можуть дати достовірних результатів і їх точність становить близько 66%. У той же час комп'ютерна томографія дозволяє виявити тип ураження легень з набагато вищим ступенем достовірності – до 99%. Крім того, КТ дозволяє лікарям отримати структуру ураження, яка може бути основною важливою інформацією про перебіг інфекції. В роботі розглядається застосування удосконаленого алгоритму самоорганізованого лісу для вирішення задачі класифікації типів уражень легень.

Основні інструменти у даній роботі: логістична регресія як функція голосування лісу, згорткова нейронна мережа, алгоритми класу Random Forest, матриця текстурних характеристик GLCM.

Мета роботи: Вирішення задачі підвищення ефективності одного з найбільш розповсюджених і часто використовуваних алгоритмів класифікації з класу Random Forest.

Основні задачі практики:

• Розробка алгоритму та програмної реалізації лісу самоорганізованих дерев.

• Вирішення задачі класифікації типу уражень легень при захворюванні на Covid-19.

Період проходження практики: з 02.05.2022 р. по 19.06.2022 р.

Загальний обсяг годин: 405 годин.

Перелік основних матеріалів: База даних КТ зображень легенів пацієнтів ДУ «Національний інститут фтизіатрії і пульмонології ім. Ф.Г. Яновського НАМН України».

РОЗДІЛ 1
ЛІТЕРАТУРНИЙ ОГЛЯД

1.1 Випадковий ліс прийняття рішень

Вперше алгоритм випадкового лісу був запропонований китайським вченим на ім’я Тін Кам Хо у 1995 році [12]. Цей алгоритм є представником класу метаалгоритмів ансамблевого навчання для класифікації, регресії та вирішення інших задач, в основі якого лежить одночасна побудова скінченної множини незалежних дерев прийняття рішень. Випадковий ліс з допомогою детермінованої функції усереднення поєднує рішення ансамблю моделей, що дозволяє отримати результати прогнозування з більшою точністю.

Алгоритми випадкового лісу на сьогоднішній день залишаються одним з найпопулярніших рішень сучасних задач та широко застосовується у задачах класифікації різної складності, що підтверджує актуальність такого ансамблевого алгоритму[10].

1.2 МГУА

Метод групового урахування аргументів (МГУА) – метод, що належить о категорії підходів індуктивної самоорганізації [1]. Він вимагає невеликих вибірок даних і здатний об’єктивно оптимізувати структуру моделей. Індуктивний підхід подібний до нейронних мереж, але є необмеженим за своєю природою, де незалежні змінні системи зміщуються випадковим чином і активуються так, що в кінцевому підсумку вибирається найкраща відповідність залежним змінним.

Автори дослідження [9] пропонували шляхи удосконалення в класі алгоритмів Random Forest з використанням приципів методів групового урахування аргументів. Таким чином, кожне окреме дерево за результатами навчання мало більшу точність класифікації, що призвело до покращення результатів прогнозування ансамблевого алгоритму.

1.3 Логістична регресія

Модель логістичної регресії приймає натуральний логарифм шансів як функцію регресії від предикторів [8]. Використовується у випадку, коли залежна змінна є бінарною. Із використанням значення порогу може застосовуватись як метод класифікації.

Автори дослідження [3] застосували логістичну регресію як функцію агрегації прогнозів самоорганізованих моделей. За результатами дослідження, були отримані результати підвищеної точності у порівнянні із класичним алгоритмом випадкового лісу.

Висновки до розділу 1

У цьому розділі було проаналізовано опрацьовані літературні джерела та існуючі ансамблеві алгоритми в класі випадкового лісу та варіації їх удосконалення. Було оглянуто поширений метод самоорганізації МГУА, а також модель логістичної регресії.

РОЗДІЛ 2
ТЕОРЕТИЧНА ЧАСТИНА

2.1 Алгоритм випадкового лісу

Випадковий ліс — популярний алгоритм машинного навчання, який належить до класу навчання з учителем. Його можна використовувати як для задач класифікації та регресії. Він заснований на концепції ансамблевого навчання, що являє собою процес об’єднання кількох класифікаторів для вирішення складної проблеми та покращення продуктивності моделі [12].

Як випливає з назви, випадковий ліс — це класифікатор, який містить ряд дерев рішень для різних підмножин даного набору даних і усереднює значення, щоб підвищити точність прогнозування цього набору даних. Замість того, щоб покладатися на одне дерево рішень, випадковий ліс бере прогноз з кожного дерева на основі більшості голосів передбачень і прогнозує кінцевий результат.

Що більша кількість дерев у лісі, то більшу точність матиме класифікатор, що в свою чергу також запобігатиме проблеми перенавчання.

Діаграма нижче пояснює роботу алгоритму (рис. 2.1):

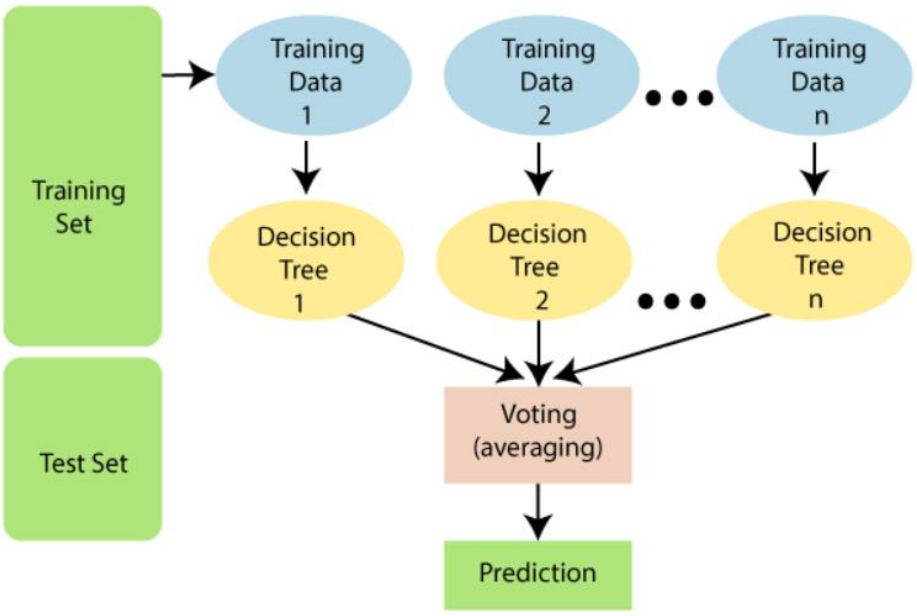


Рисунок 2.1 – Діаграма алгоритму випадкового лісу

Алгоритм випадкового лісу складається з двох фаз:

1. Створення випадкового лісу шляхом об’єднання N дерева рішень
2. Створення прогнозів для кожного дерева, створеного на першому етапі.

Принци роботи можна пояснити на наступних кроках:

Крок 1. Обрати K випадкових точок даних із навчального набору.

Крок 2. Побудувати дерево прийняття рішень на підмножині даних, що була обранана попередньому кроці.

Крок 3: Обрати число N, що позначатиме кількість дерев у лісі.

Крок 4: Повторювати кроки 1 і 2 допоки буде отримано N дерев.

Крок 5. Для точок даних із тестового набору необхідно отримати прогнози кожного дерева рішень і класифікувати кожну точку до тої категорії, для якої кількість прогнозів серед дерев виявилось найбільше.

2.2 Алгоритм МГУА

В загальному випадку, зв'язок між вхідними та вихідними змінними моделі можна апроксимувати функціональним рядом Вольтерра, дискретним аналогом якого є поліном Колмогорова-Габора (2.2):

$$y = a_0 + \sum_{i=1}^{m} a_i x_i + \sum_{i=1}^{m}\sum_{j=1}^{m} a_{ij} x_i x_j + \sum_{i=1}^{m}\sum_{j=1}^{m}\sum_{k=1}^{m} a_{ijk} x_i x_j x_k + \cdots \qquad (2.1)$$

де $x=(x_1, x_2, \ldots, x_m)$ – вектор вхідних змінних, $A=(a_0, a_1, a_2, \ldots, a_m)$ – вектор вагів. Поліном Колмогорова-Габора може апроксимувати будь-яку стаціонарну випадкову послідовність спостережень і може бути обчислений або адаптивними методами, або системою нормального рівняння Гаусса.

Автор [6], натхненний формою полінома Колмогорова-Габора, розробив новий алгоритм, який було названо «Методом групового урахування аргументів (МГУА)». Дотримуючись підходу евристичного та персептронного типу, він

намагався повторити поліном Колмогорова-Габора, використовуючи поліноми низького порядку для кожної пари вхідних змінних. Він довів, що поліном другого порядку (2.2) може відновити повний поліном Колмогорова-Габора за допомогою ітераційної процедури типу персептрона.

$$y = a_0 + a_1 x_i + a_2 x_j + a_3 x_i x_j + a_4 {x_i}^2 + a_5 {x_j}^2 \quad (2.2)$$

Під час процедури моделювання алгоритм МГУА включає чотири евристики, які представляють основні особливості теорії методу [5]:

1. Накопичення набору спостережень, який потенційно може має відношення до досліджуваного об’єкта.
2. Розділення набору даних на дві групи. Перша буде використовуватися для оцінки коефіцієнтів (вагів) моделі, а другий виділить корисну інформацію, що прихована у вхідних даних.
3. Створення набору елементарних функцій, складність яких буде зростати в результаті ітераційної процедури, що створюватиме більш комплексні моделі.
4. Застосування зовнішнього критерію для вибору оптимальної моделі.

Підхід МГУА може бути корисним з наступних причин:

- Знайходження оптимальної складності структури моделі, відповідно до рівня шуму в вибірці даних. Для реальних задач із зашумленими або короткими даними більш точними є спрощені оптимальні моделі.
- Кількість шарів і нейронів у прихованих шарах, структура моделі та інші оптимальні гіперпараметри визначаються автоматично.
- Гарантія того, що будуть знайдені найбільш точні та неперенавчені моделі – метод не пропускає найкраще рішення під час сортування всіх варіантів.
- В якості вхідних змінних можуть використовуватися нелінійні функції або ознаки, які можуть впливати на вихідну змінну.

- Метод автоматично знаходить інтерпретовані відносини в даних і вибирає ефективні вхідні змінні.
- Метод отримує інформацію безпосередньо із вибірки даних і мінімізує вплив апріорних припущень автора щодо результатів моделювання.
- Підхід дає можливість знайти об'єктивну фізичну модель об'єкта (закон або сегментацію).

2.3 Побудова матриці GLCM

Статистичним методом дослідження текстури, який враховує просторове співвідношення пікселів, є матриця спільного виникнення сірого рівня (Gray-Level Co-occurrence Matrix – GLCM), також відома як матриця просторової залежності рівня сірого. Функції GLCM характеризують текстуру зображення та несуть інформацію про те, як часто в зображенні зустрічаються пари пікселів із певними значеннями та в конкретній просторовій орієнтації.

Побудова матриці відбувається за алгоритмом, що буде описано нижче [4].

Для прикладу буде розглянуто синтетичне дрібне зображення (рис. 2.2). Значення на рис. 2.3 є рівнями сірого зображення. Чим нижче число рівня сірого, тим темніше зображення.

Рисунок 2.2 – тестове зображення 4x4 пкс для побудови матриці GLCM

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

Рисунок 2.3 – матриця рівнів сірого для тестового зображення на рис. 2.2.

Текстура GLCM розглядає відношення між двома пікселями одночасно, що мають назви опорного і сусіднього пікселя. За замовчуванням, сусідній піксель обирається праворуч від опорного. Кожен піксель на зображенні по черзі стає опорним пікселем. Пікселі вздовж правого краю не мають сусіда справа, тому вони не використовуються для побудови матриці в якості опорних пікселів. На рис. 2.4 показано кілька таких попарних зв’язків: червоний пікселів є опорним пікселем, а синій – сусіднім.

0	0	1	1
0	0	1	1
0	2	2	2
2	2	3	3

Рисунок 2.4 – демонстрація опорних та сусідніх значень матриці рівнів сірого

Основна ідея матриці GLCM полягає в підрахунку частоти появлення пар опорний-сусідній пікселі з однаковими відповідними значенням. Рядки матриці представляють собою значення, що можуть приймати опорні пікселі, а стовпчики – сусідні пікселі. Пара, що має опорний піксель з рівнем сірого i, та сусідній піксель з рівнем j, додаватиме одиницю у відповідну клітинку (i, j) матриці GLCM. Після обчислення кожної пари, отримаємо матрицю GLCM (рис. 2.5) для розглянутого зображення.

2	2	1	0
0	2	0	0
0	0	3	1
0	0	0	1

Рисуноок 2.5 – матриця GLCM для зображення на рис. 2.2.

Розглянута матриця була побудована для взаємного просторового відношення пікселів (1,0), тобто сусідній піксель знаходився строго праворуч на відстані одного пікселя. Проте можна застосовувати й інші просторові відношення, наприклад (-1,0) – сусідній піксель знаходитиметься ліворуч від опорного, (1,1) – по діагоналі справа знизу, (2,0) – відстань між пікселями дорівнюватиме 2 пкс. Для кожної конфігурації просторового відношення створюватиметься нова матриця GLCM.

2.4 Модель логістичної регресії

Цей тип статистичної моделі часто використовується для класифікації та прогнозної аналітики. Логістична регресія оцінює ймовірність настання події, наприклад, на основі заданого набору даних незалежних змінних. Оскільки результат є ймовірністю, залежна змінна обмежена між 0 і 1. У логістичній регресії застосовується до шансів, тобто ймовірність успіху, поділена на ймовірність невдачі. Це також широко відоме як логарифм шансів або натуральний логарифм шансів, і ця логістична функція представлена формулою (2.3) [11].

$$p(x) = \frac{1}{1 + e^{-(x-\mu)/s}} \tag{2.3}$$

де μ — параметр розташування (середина кривої, де p(μ)=1/2), а s — параметр масштабу.

Висновки до розділу 2

У цьому розділі було опрацьовано основні теоретичні положення щодо алгоритму випадкового лісу, методу групового урахування аргументів. Також, було оглянуто принципи побудови матриці GLCM та роботи моделі логістичної регресії.

РОЗДІЛ 3
АНАЛІТИЧНА ЧАСТИНА

3.1 Мова програмування

Мовою програмування для реалізації програмного продукту та алгоритму лісу самоорганізованих дерев буде обрано Python. Python – це проста, але потужна інтерпретована мова програмування, яка долає складнощі, що присутні в компільованих мовах та дозволяє швидке створення прототипів. Її синтаксис складається з конструкцій, запозичених з багатьох інших мов; найбільш помітними є впливи від ABC, C, Modula-3 та Icon. Інтерпретатор Python легко розширюється новими функціями та типами даних, реалізованими на C. Python також підходить як мова розширення для програм C для створення додаткового функціоналу, наприклад текстові редактори або віконні менеджери. До того ж, ця мова програмування доступна для різних операційних систем.

Python має велику кількість бібліотек машинного навчання, що мають простий та дружелюбний інтерфейс та пропоноють готову реалізацію великої кількості алгоритмів. Зокрема, найрозповсюдженіші бібліотеки – NumPy, Pandas, sklearn, SciPy, TensorFlow, PyTorch тощо.

Для реалізації алгоритму стануть в нагоді наступні бібліотеки:

- NumPy – бібліотека, що дозволяє проводити обчислення в галузі лінійної алгебри. Реалізація бібліотеки високооптимізована, що дозволяє швидко проводити бажані операції. Одним із важливих критеріїв оптимізації є векторизація. Векторизація описує відсутність будь-якого явного циклу в коді - ці речі відбуваються «за кадром» в оптимізованому, попередньо скомпільованому коді C. Векторизований код має багато переваг, серед яких:
    - векторизований код є більш стислим і легшим для читання
    - менше рядків коду допоможе уникнути більшої кількості помилок

- код більше нагадує стандартні математичні позначення (що полегшує, як правило, правильне кодування математичних конструкцій)
- векторізація призводить до більш «Pythonic» коду. Без векторизації код був би завалений неефективними і важкочитаними циклами.

Ще одним фактором оптимізації є трансляція — це термін, який використовується для опису неявної поелементної поведінки операцій. Взагалі кажучи, у NumPy всі операції, не тільки арифметичні, а й логічні, бітові, функціональні тощо, поводяться неявно поелементно, тобто вони транслюються.

- sklearn – один з найпоширеніших робочих інструментів в сфері машинного навчання серед розробників на Python. Це бібліотека з відкритим вихідним кодом, написана на основі таких бібліотек як NumPy, SciPy, Matplotlib. Вона містить великий набір інструментів для статистичного моделювання, машинного навчання та пропонує готові рішення до таких задач, як кластеризація, класифікація, регресія, зменшення розмірності. До того ж, можна скористатись інструментами для оцінки, синтезу та порівняння моделей та попередньої обробки даних. Усе це об’єднано в простий інтерфейс, що дозволяє розробнику за невеликий час розробити прототип власної моделі або скористатись готовим рішенням для вирішення прикладної задачі.
- Pandas – комплексна бібліотеки для зручної роботи з чисельними і не тільки таблицями. Широкий програмний інтерфейс дозволяє проводити складні операції з даними, що зібрані у відповідних структурах даних. Також бібліотека пропонує зручну взаємодію з індексами структур даних, що стане ключовим елементом в реалізації алгоритму.

3.2 Середовище розробки

В якості середовища розробки можна обрати одну із наступних альтернатив:

Pycharm — спеціальне кросплатформне інтегроване середовище розробки для мови програмування Python. Воно включає в себе широкий спектр унікальних функцій та інструментів, які забезпечують ефективність і зручність програмування на Python, зокрема в області науки про дані та веб-розробки. PyCharm підтримує версії Python 2 (2.7) і Python 3 (3.5 і вище) і сумісний з Windows, macOS і Linux. Pycharm легко налаштовується, тому кожен користувач може підлаштувати під себе користувацький інтерфейс та увімкнути/вимкнути опціональні можливості середовища. Також, PyCharm пропонує велику кількість плагінів, які можна встановити та увімкнути за кілька секунд.

Jupyter Notebook – це сервер-клієнтська програма, що дозволяє редагувати та запускати код у так званих блокнотах через веб-браузер. Програму можна запустити на ПК без доступу до Інтернету, або встановити її на віддалений сервер, та користуватись через мережу. Jupyter має два основні компоненти - ядро та панель інструментів.

1. Ядро - це програма, яка запускає та інтроспектує код користувача. У застосунку Jupyter Notebook є ядро для інтерпретування Python коду, але є ядра і для інших мов програмування.
2. Панель інструментів програми має не лише документи блокнота, що були створені розробником раніше, але також надає можливість управління ядрами – контроль за їх запуском, зупинкою, конфігурацією тощо.

Jupyter Notebook дозволяє «по клаптиках» виконувати код, що полегшує дослідницьку роботу, оскільки можна розбити усе дослідження на кілька частин, та розділити код, що відповідає кожному розділу. Також це полегшує роботу з візуалізацією даних, і в одному блокноті можна створити та зберегти бажану кількість корисних графіків.

Visual Studio Code — це простий та водночас потужний редактор коду від компанії Microsoft, який являється кросплатформовим. Він встановлюється з підтримкою багатьох мов програмування, серед яких є Python. Так само як і PyCharm IDE, Visual Studio Code має у своїй базі велику кількість розширень та доповнень, що можуть вільно використовуватись користувачами цього текстового редактору. Величезною перевагаю цього редактору є вбудована підтримка Jupyter Notebook. Тож Visual Code дозволяє поєднувати роботу зі звичайними файлами Python коду з розширенням «.py» та інтерактивними блокнотами, для яких як було згадано раніше потрібен запуск локального сервера, за що відповідає вбудована інтеграція програми Jupyter в редакторі. Інтегроване середовище PyCharm також має аналогічний плагін, проте він доступний в комерційній версії застосунку.

Орієнтуючись на вище згадані аргументи, практична частина буде виконуватись в редакторі Visual Studio Code.

3.3 Аналіз існуючих рішень

3.3.1 XGBoost

XGBoost — це ансамблевий алгоритм машинного навчання на основі дерев прийняття рішень, який використовує інфраструктуру підвищення градієнта. У задачах прогнозування, що включають неструктуровані дані (зображення, текст тощо), штучні нейронні мережі мають тенденцію перевершувати всі інші алгоритми чи методи. Однак, коли справа доходить до структурованих/табличних даних, алгоритми на основі дерева прийняття рішень зараз вважаються найкращими у своєму класі.

Нижче продемонстрована еволюція алгоритмів на основі дерев рішень:

1. Звичайне дерево прийняття рішень – представляє собою структуру, подібну до блок-схеми. Шлях від кореня дерева до листових вузлів представляють собою правила класифікації, а листовий вузол являється результатом та містить у собі мітку прогнозованого класу.
2. Беггінг – ансамблевий мета-алгоритм, що комбінує прогнози кількох дерев прийняття рішення з допомогою механізму голосування.

3. Випадковий ліс – яскравий представник беггінгу, де для побудови дерев прийняття рішень або їх колекцій використовується випадкова підмножина навчальної множини даних.
4. Бустинг – мета-алгоритм, який припускає послідовне створення моделей, де кожна наступна має ціль мінімізувати помилку попередників. Алгоритм також збільшує вплив високоточних моделей.
5. Градієнтний бустинг – мінімізація помилки ансамблевого алгоритму виконується з допомогою методу градієнтного спуску.
6. XGBoost – оптимізований алгоритм градієнтного бустингу, що включає в себе паралельний синтез дерев, обрізка дерев прийняття рішень, обробка відсутніх значень, а також регуляризація для уникнення перенавчання.

Алгоритм XGBoost був розроблений як дослідницький проект в Університеті Вашингтона. Тяньці Чен і Карлос Гестрін представили свою роботу на конференції SIGKDD у 2016 році і вразили світ машинного навчання [2]. З моменту запровадження цей алгоритм не лише переміг у численних конкурсах Kaggle, але й був рушійною силою для кількох передових галузевих додатків. Алгоритм вирізняється за такими причинами:

Широкий спектр застосувань: може використовуватися для вирішення завдань регресії, класифікації, ранжирування та визначених користувачем завдань прогнозування.

Кросплатформовість: працює на Windows, Linux і OS X.

Мови: підтримує найпоширеніші мови програмування, в тому числі C++, Python, R, Java, Scala і Julia.

Хмарна інтеграція: підтримує кластери AWS, Azure і Yarn і добре працює з Flink, Spark та іншими екосистемами.

3.3.2 GMDH Shell

Програмне забезпечення GMDH Shell – інструмент для прогнозування, розроблений на основі штучних нейронних мереж. Користувач може легко і швидко сторювати прогнозні моделі, а також проводити процедури попердньої обробки даних.

GMDH Shell має наступні переваги (рис. 3.1):

1. Створення прогнозу в кілька кліків

GMDH Shell дозволяє вирішувати завдання прогнозування різної складності або аналізу даних з мінімальними зусиллями. Завдяки методу прогнозування GMDH і сучасній технології паралельних обчислень, що лежить в основі програми, вона здатна надавати надзвичайно точні прогнози часових рядів, і робить це набагато швидше, ніж звичайні штучні нейронні мережі.

2. Перевірені алгоритми всередині

GMDH Shell базується на методі групового урахування аргументів, який є вдосконаленою версією класичного методу регресійного аналізу, розробленого в 60-х роках. Автори вдосконалили алгоритм, зробивши його справді передовою технологією, яка підходить для широкого кола застосувань.

3. Універсальний і легко налаштовуваний

GMDH Shell на основі розширеної математики є універсальним рішенням, ідеальним для прогнозування «під ключ» і легкого аналізу часових рядів. У той же час, повний спектр параметричних налаштувань і вільний вибір методів прогнозування, а також безліч унікальних опцій дозволяють налаштувати програму під будь-яку конкретну задачу від біологічного та хімічного аналізу до прогнозування фондового ринку та аналізу часових рядів.

4. Швидкість

GMDH Shell повністю використовує можливості вашого комп’ютера. Він використовує всі можливі процесори та їхні ядра для паралельного виконання обчислень, щоб пришвидшити отриманні результатів.

GMDH Shell можна налаштувати як автономний прогнозів. Легкий і простий інтерфейс не займе для цього багато часу.

5. Безкоштовний період

Будь-який користувач може завантажити програмне забезпечення та спробувати роботу в ньому впродовдж безкоштовного періоду. Таким чином можна перевірити та оцінити продуктивність GMDH Shell навіть з використання власних даних, а не випадкової вибірки.

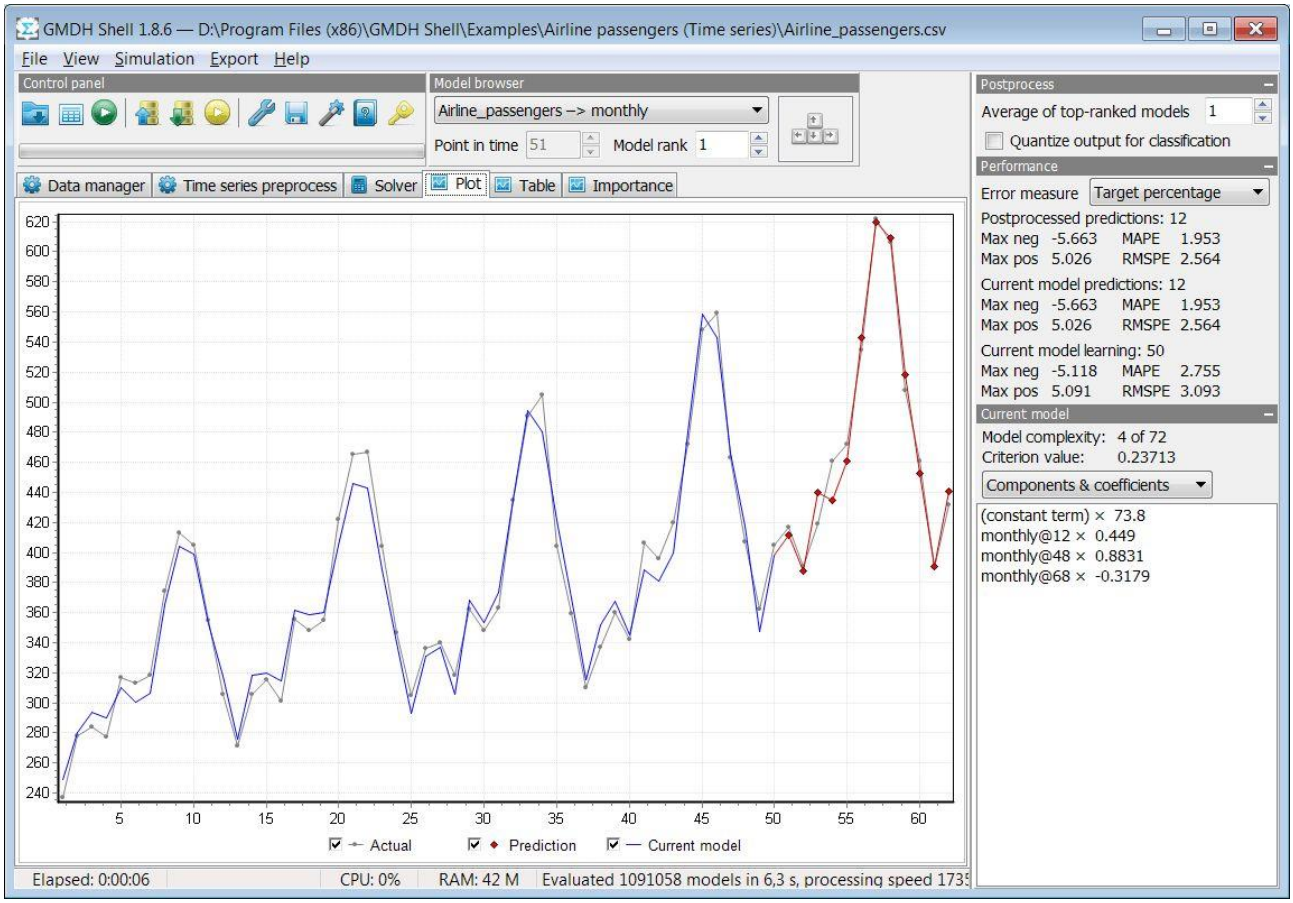


Рисунок 3.1 – Інтерфейс GMDH Shell

3.4 Алгоритм логістичного лісу самоорганізованих дерев

Кожне дерево лісу представляє собою бінарне дерево, у вузлах якого обчислюються ознаки із відповідними порогами та знаками. Класифікація здійснюється шляхом порівняння значення ознаки об'єкта з пороговим значенням та знаком. Якщо значення ознаки X_i більше порогу P_i зі знаком «>», то шлях об’єкта, що класифікується пролягає через правого нащадка, інакше – через лівого. У випадку знака «<» вибір шляху інвертується. Процедура виконується до тих пір, поки об’єкт не досягне листового вузла, який і визначатиме клас об’єкта. Саме цей листовий вузол стане вирішальним у прогнозі. Далі йтиме алгоритм побудови дерева:

Попередній етап. В процесі передобробки даних збільшується простір об’єктів від X до f та розбивається вибірка.

Набір даних R розбивається на множини A, B, C. Тут нехай $n_{AB} = 0,8 * n_R$, $C = 0,2 * n_R$. Кожне дерево будується на наборі AB, згідно з (3.1):

$$n_A = 0,7 * n_{AB} \pm k_i,\ n_B = 0,3 * n_{AB} \mp k_i \quad (3.1)$$

тут ki – випадкове число від [0, 0,5]. Співвідношення (3) забезпечує мінливість ознак і порогів у вузлах при побудові і-го дерева лісу. Набір даних розподілений стратифікованим способом рівномірно по дисперсії відносно центрів класів у просторі ознак f.

Побудова вузла. Пошук кращого порога (рис. 3.2) для ознаки *i* з *f* у вузлі відбувається за F^{sc}-метрикою на A (далі F^{scA}) множини P_i (3.2) за виключенням діапазонів, де значення ознак на варіаційному ряді об'єктів за номером класу слідують підряд.

$$\{p_i \in P_i \mid p_i \in \{\frac{x_{j+1}+x_j}{2}, j = 1,..,n_i, x_j \in [c_{0i}; c_{1i}]\}\} \quad (3.2)$$

де p_i – значення порогу для ознаки *i*, *xj* – *j*-те значення ознаки із множини порогів, n_i – кількість елементів множини P_i, c_{0i}, c_{1i} – центри класів за ознакою *i*.

Вибір *F* кращих ознак f_{best} для вузла відбувається за комбінованою F-метрикою на A + B відповідно (3.3).

$$F^{sc*} = w \cdot F^{scB} + (w - 1) \cdot F^{scA} \quad (3.3)$$

Для кожної ознаки з f_{best} будуються лівий та правий нащадки. Оскільки нащадки класифікують точки даних на підмножинах, що не перетинаються, це дозволяє скористатись наступним фактом: F^*-метрика суми нащадків дорівнює сумі F^*-метрик. Кращою вважається ознака, для якої максимізується (3.4)

$$F^{sc*}_{total} = F^{sc*}_{left} + F^{sc*}_{rigth} \quad (3.4)$$

, де F^{sc*}_{left} – F^{*}-метрика лівого нащадка, F^{sc*}_{rigth} – F^{*}-метрика правого нащадка. Краща ознака зберігається в корені разом із порогом та знаком.

Таким чином, краща ознака на поточному рівні p обирається не тільки за власною , а й за найкращими результатами на рівні p+1, що дозволяє отримати кращі результати в цілому.

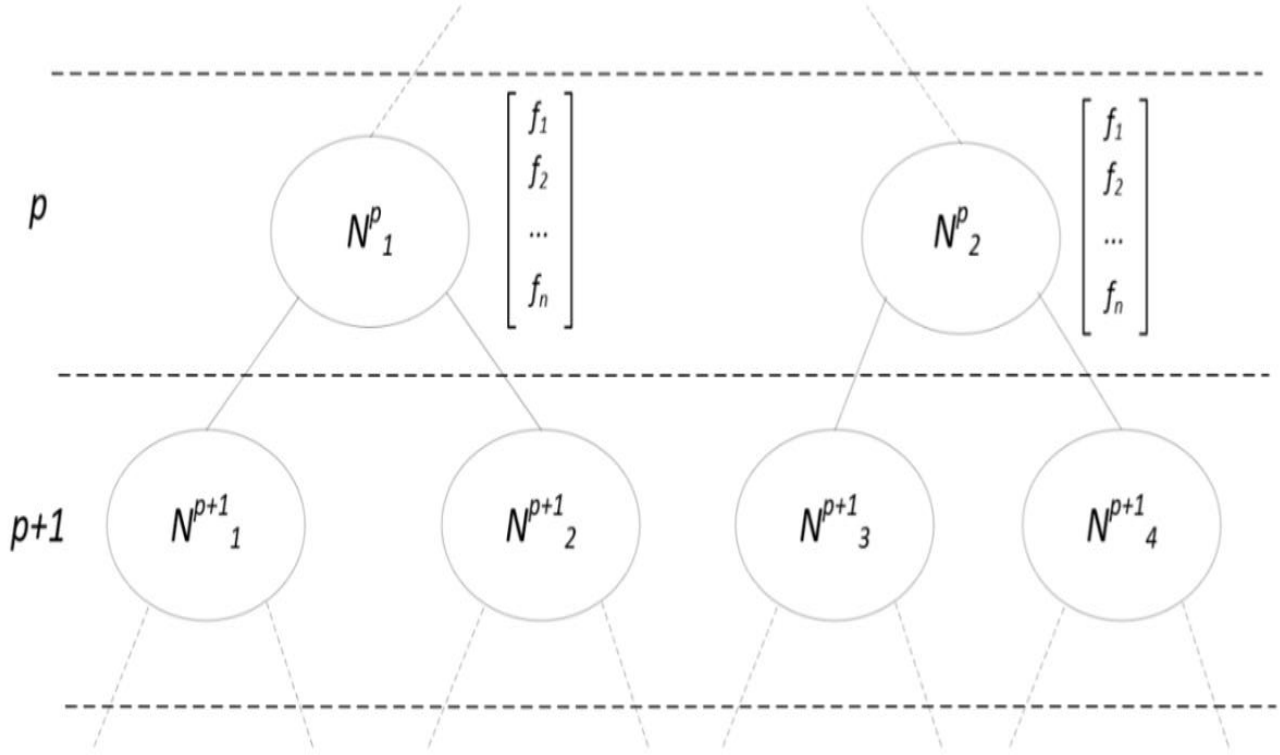


Рисунок 3.2 – Два послідовних рівня дерева в самоорганізаційному лісі

Побудова усього дерева починається з вузла-кореню та проводиться в ширину та в висоту. Критерій зупинки у розростанні дерева визначається максимальною глибиною дерева *d*, або ж за умови, що F^{sc*} усього дерева після побудови поточного вузла не покращилась. У разі досягнення критерію зупинки, поточний вузол залишає за собою ознаку із порогом, що найкраще класифікує дані. Кожне дерево після завершення навчання класифікує вхідні дані із множини AB та повертає вектор прогнозів. Останній складає множину ознак для логістичної регресії, де кожна ознака індексується порядковим номером дерева.

Навчання логістичної регресії відбувається на множині AB, а тестування – на множині C. Кількість дерев в лісі збільшується поки метрика якості логістичної регресії зростає.

3.5 Покращення алгоритму лісу

В підрозділі 3.4 розглядається алгоритм лісу самоорганізованих дерев. У ньому при побудові вузла дерева виконується пошук найкращої ознаки разом із порогом та знаком, яка гарантуватиме найкращу продуктивність в нащадках вузла, оскільки вирішується задача максимізації (3.4).

Для покращення прогнозів моделі самоорганізованого дерева, пропонується універсалізувати число q, що позначає кількість рівнів дерева в глибину, де відбувається порівняння F^{sc*}-метрик для пранащадків.

Таким чином формула (3.4) приймає новий вигляд (3.5)

$$F^{sc*}_{total} = \begin{cases} F^{sc*}_{left} + F^{sc*}_{right}, & q = 1 \\ F^{sc*}_{total,left} + F^{sc*}_{total,right}, & q \geq 2 \end{cases} \quad (3.5)$$

, де $F^{sc*}_{total,left}$, $F^{sc*}_{total,right} - F^{sc*}_{total}$ для лівого та правого нащадків відповідно.

Покращений алгоритм потенційно може мати кращі результати, оскільки в процесі побудови кожного вузла дерева відбувається пошук такої умови, що дасть найкращі результати на q рівнів уперед, що призведе до покращення результатів прогнозування усього дерева.

Висновки до розділу 3

У цьому розділі було проаналізовано та обрано мову програмування для реалізації програмного застосунку, порівняно середовища розробки та обрано найбільш сприятливу альтернатив. Також було оглянуто існуючі рішення за темою практики, та виділені їхні переваги. Основною частиною цього розділу став розглянутий та детально описаний алгоритм логістичного лісу

самоорганізованих дерев за критерієм якості прогнозу класифікації. Також, було запропоновано покращення вищезгаданого алгоритму.

РОЗДІЛ 4
ПРАКТИЧНА ЧАСТИНА

4.1 Постановка задачі

Передбачається, що існує скінченна кількість класів D^*_i, $i = 1, ..., K$, K=3, уражень легеневої тканини пацієнтів з захворюванням на COVID-19 що представляють собою наступні типи:

1. «ground-glass opacity»
2. «crazy-paving»
3. «consolidation»

Класи відображаються у вигляді наборів зображень (об’єктів класифікації d*), які представлені у вигляді областей інтересу (ROI) КТ-зображень легенів пацієнтів. Кожен клас D^*_i, $i = 1, ..., K$ є скінчнним або нескінченним будь-яким набором зображень ROI d^*. Окрім того, передбачається (4.1).

$$D^*_i \cap D^*_j, i \neq j \tag{4.1}$$

Такі класи дають скінченні навчальні підмножини D_i ступеня n_i, $i = 1,..., K$, представлені об’єктами $(ROI)_{ij}$, де $j = 1,..., n_i$. Кожен об’єкт d_{ij} є фрагментом КТ-зображення легенів людини $(ROI)_{ij}$, який позначений як патологічний. На основі наведених навчальних підмножин D_i, $i = 1, ..., K$ необхідно створити механізм найкращої класифікації об'єктів d^*_{ij} з D_i, $i = 1, ..., K$ в обраному класі.

4.2 Дані для практичної задачі

Анонімні дані для розробки класифікатора надано ДУ «Національний інститут фтизіатрії та пульмонології імені Ф.Г. Яновського НАМН України». Набір даних складається з 1831 позначеної області інтересу. Дані були отримані у форматі Nifti1, щоб зберегти оригінальні величини одиниць Хаунсфілда.

Розбиття вибірки даних на тренувальну, тестову та екзаменаційну вибірки представлені у таблиці 4.1.

Таблиця 4.1.

Розбиття вибірки

Клас	**Розмір тренувальної вибірки**	**Розмір тестової вибірки**	**Розмір екзаменаційної вибірки**	**Загальна кількість**
«Ground-glass opacity»	585	33	33	650
«Crazy-paving»	585	33	32	650
«Consolidation»	477	27	26	531

4.3 Підготовка даних та генерація ознак

Для класифікації типів ураження легень при COVID-19 будемо використовувати текстурні характеристики, які відображають характеристики залежності значень інтенсивності сусідніх пікселів зображення.

В цій роботі для кожної області інтересу були розраховані матриці GLCM з наступними просторовими співвідношеннями:

- (0, 1)
- (1, -1)
- (-1, 0)

Для генерації ознак був використаний конструктор ознак – сконфігурована згорткова нейронна мережа, розроблена авторами [3]. У роботі [14] схожу задачу вирішували за допомогою автокодера та аналізу головних компонент, проте для даної робота було застосовано саме перше рішення, оскільки основна перевага такого підходу полягає в тому, щоб отримати оптимальне формування ознак для найкращого вирішення розглянутої задачі класифікації.

Структура конструктору ознак продемонстрована на рис. 4.1.

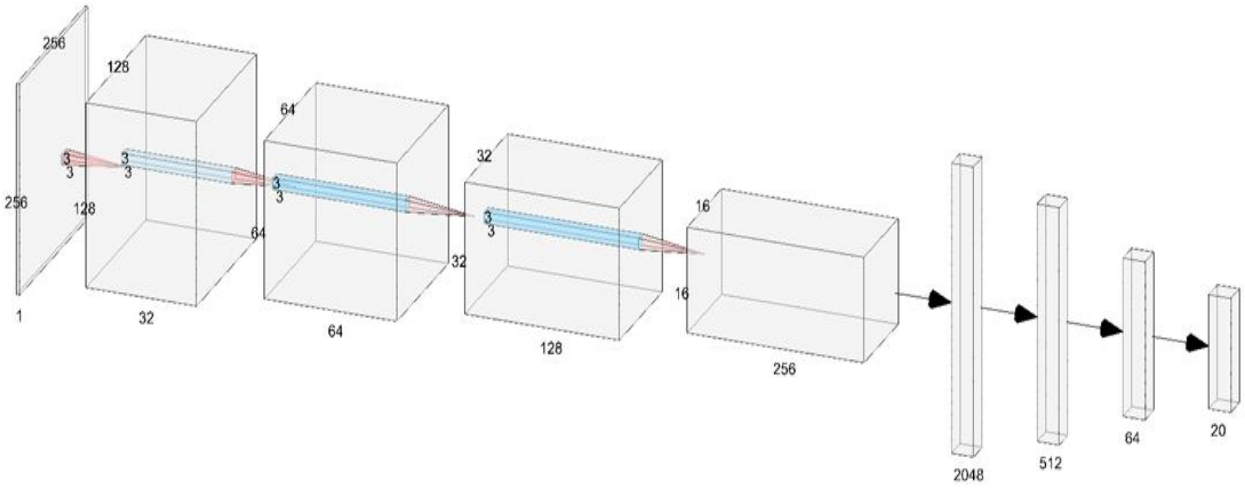


Рисунок 4.1 – Структура конструктору ознак для обробки GLCM

4.4 Застосування логістичного лісу самоорганізованих дерев

Отримані в результаті конструювання ознаки, ставили вхідними даними для самоорганізованого логістичного лісу. Також була застосована техніка “skip connections”, за якою голоси дерев з моделі LSOF поєднуються з ознаками, отриманими від конструктору ознак з метою оптимізації функції логістичного голосування. Такий принцип поєднання ознак в структурах використовується в ResNet [7].

Поєднавши конструктор ознак та розроблений алгоритм, було отримано гібридний класифікатор, що здатний вирішувати задачу класифікації типів уражень легеней пацієнтів хворих на COVID-19 за областями інтересу. Отримана структура гібридного класифікатору зображена на рис. 4.2.

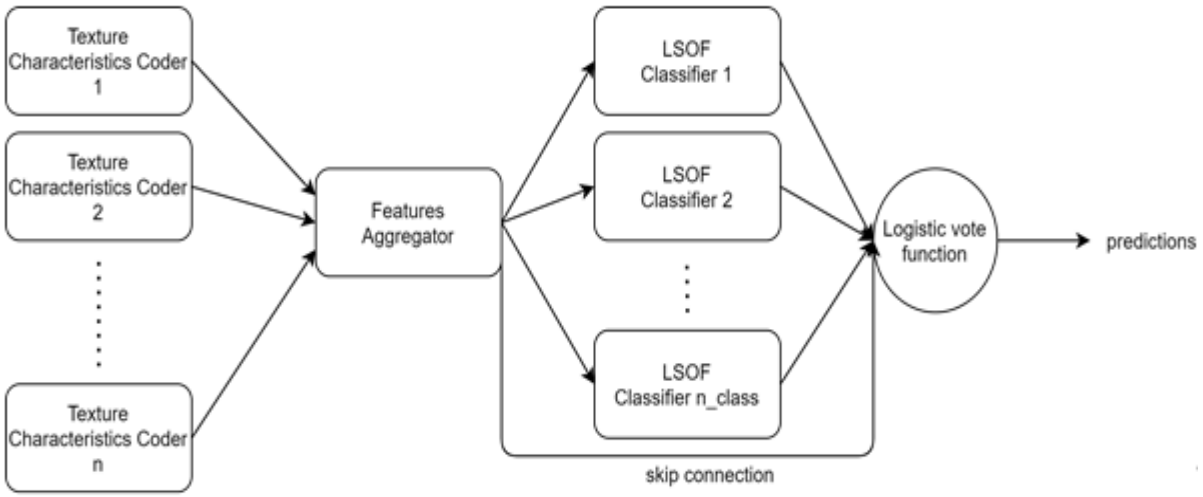


Рисунок 4.2 – структура розробленого гібридного класифікатора, де “Texture Characteristics Coder” – матриця GLCM, “Feature Aggregator” – конструктор ознак, “LSOF Classifier” – самоорганізоване дерево, “Logistic vote function” – логістична функція голосування

4.5 Приклади побудованих дерев

На рисунку 4.3 наведено структуру одного із побудованих дерев самоорганізації для q=1.

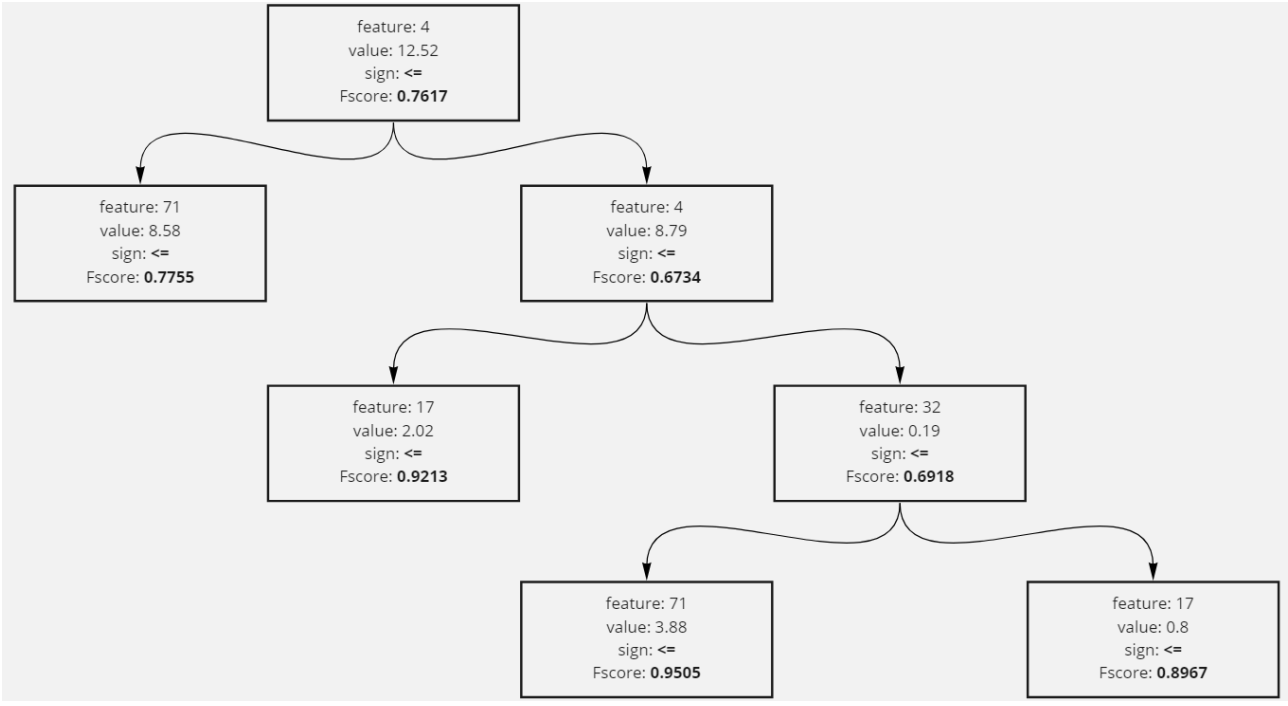


Рис. 4.3 – структура побудованого самоорганізованого дерева для q=1

На рисунку 4.4 наведено структуру одного із побудованих дерев самоорганізації для q=2.

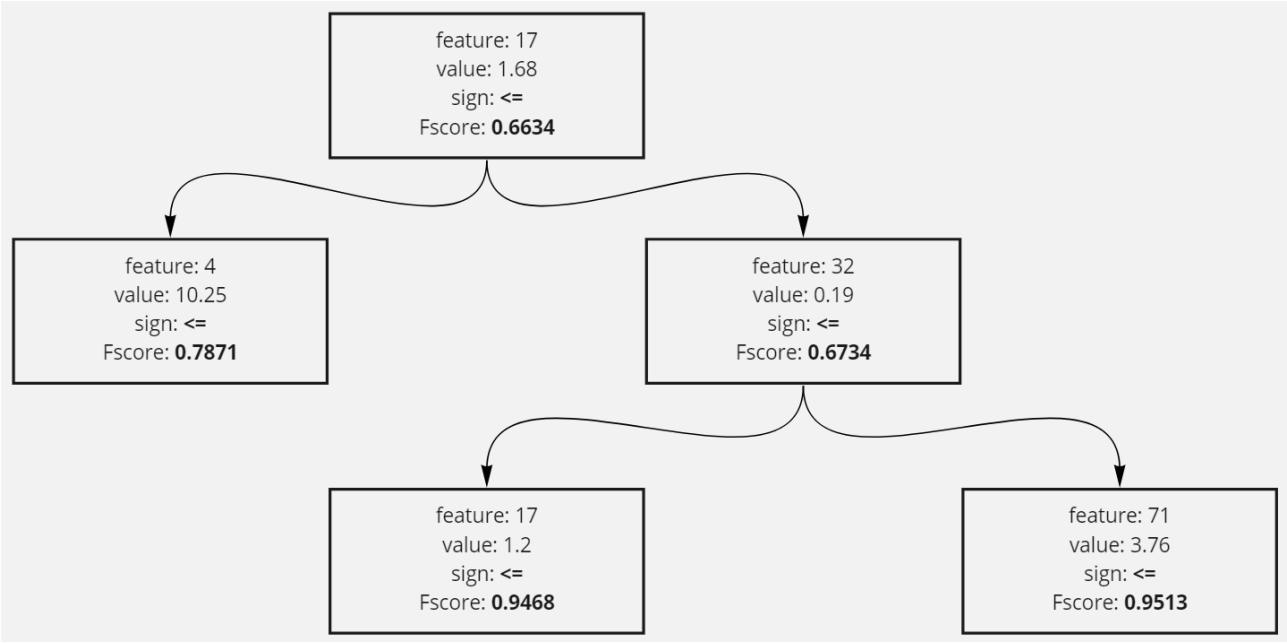


Рис. 4.4 – структура побудованого самоорганізованого дерева для q=2

4.6 Порівняння результатів

Для оцінки роботи класифікатора, порівняємо наступні три результати прогнозування:

1. Гібридний класифікатор з конструктором ознак та базовою реалізацією випадкового лісу.
2. Гібридний класифікатор з конструктором ознак та алгоритмом логістичного лісу самоорганізованих дерев.
3. Гібридний класифікатор з конструктором ознак та покращеним алгоритмом логістичного лісу самоорганізованих дерев з універсалізацією глибини q, де q=2.

Використання першого класифікатору дали результати, наведені в таблиці 4.2.

Таблиця 4.2.

Результати першого класифікатора

Клас	**Точність**	**Повнота**	**F-метрика**
«ground-glass opacity»	0.92	0.92	0.92
«crazy-paving»	0.94	0.94	0.94
«consolidation»	0.91	0.91	0.91
Загальна точність	0.92		

Використання першого класифікатору дали результати, наведені в таблиці 4.3.

Таблиця 4.3.

Результати другого класифікатора

Клас	**Точність**	**Повнота**	**F-метрика**
«ground-glass opacity»	1	1	1
«crazy-paving»	0.97	0.93	0.95
«consolidation»	0.91	0.96	0.93
Загальна точність	0.96		

Використання першого класифікатору дали результати, наведені в таблиці 4.4.

Таблиця 4.4.

Результати третього класифікатора

Клас	**Точність**	**Повнота**	**F-метрика**
«ground-glass opacity»	1	1	1
«crazy-paving»	0.97	0.94	0.95
«consolidation»	0.92	0.96	0.94
Загальна точність	0.97		

Порівняння результатів загальної точності прогнозування наведені в таблиці 4.5.

Таблиця 4.5.

Порівняння моделей прогнозування

Модель	**Загальна точність**
Логістичний ліс самоорганізованих дерев з q=2	0.97
Логістичний ліс самоорганізованих дерев	0.96
Випадковий ліс	0.92

Висновки до розділу 4

У цьому розділі була вирішена практична задача класифікації типів уражень легеней з допомогою розробленого гібридного класифікатору. Проведено порівняння точностей моделей.

ЗАГАЛЬНІ ВИСНОВКИ

У ході виконання переддипломної практики було:

1. Проаналізовано, розроблено та реалізовано алгоритм логістичного лісу самоорганізованих дерев за критерієм якості прогнозу класифікації. Було покращено цей алгоритм шляхом універсалізації параметра q для пошуку оптимальної ознаки у вузлі дерева, що гарантуватиме найкращий результат класифікації у вузлах на q рівнів глибше.

2. Побудовано гібридний класифікатор за участі логістичного лісу самоорганізованих дерев для вирішення практичної задачі класифікації типів уражень легень за областями інтересу на базі анонімних даних, що були надані ДУ «Національний інститут фтизіатрії та пульмонології імені Ф.Г. Яновського НАМН України». Порівняно результат точності прогнозування з різними класифікаторами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

2. Anastasakis, L.; Mort, N. The development of self-organization techniques in modelling: a review of the group method of data handling (GMDH). RESEARCH REPORT-UNIVERSITY OF SHEFFIELD DEPARTMENT OF AUTOMATIC CONTROL AND SYSTEMS ENGINEERING, 2001.
3. Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
4. Davydko, O., Hladkyi, Y., Linnik, M., Nosovets, O., Pavlov, V., & Nastenko, I. (2021, September). Hybrid Classifiers Based on CNN, LSOF, GMDH in COVID-19 Pneumonic Lesions Types Classification Task. In 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT) (Vol. 1, pp. 380-384). IEEE.
5. Hall-Beyer, Mryka. GLCM texture: a tutorial. National Council on Geographic Information and Analysis Remote Sensing Core Curriculum, 2000, 3.1: 75.
6. Ivakhnenko, A. G. “Heuristic self-organization in problems of engineering cybernetics”. Automatica, 1970, 6.2: 207-219.
7. Ivakhnenko A.G. “The group method of data handling – a rival of the method of stochastic approximation”, Soviet Automatic Control c/c of Avtomatika, vol.1, no.3, pp.43-55, 1968.
8. K. He, X. Zhang, S. Ren, J. Sun, “Deep residual learning for image recognition,” In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778) (2016).
9. Lavalley, Michael P. Logistic regression. Circulation, 2008, 117.18: 2395-2399.
10. Nastenko I., Maksymenko V., Galkin A., Pavlov V., Nosovets O., Dykan I., Tarasiuk B., Babenko V., Umanets V., Petrunina O., Klymenko D. Liver Pathological States Identification with Self-organization Models Based on Ultrasound Images Texture Features. Advances in Intelligent Systems and Computing V. Cham:Springer International Publishing, 2021. pp. 401–418.

11. Prediction of Lung Cancer Risk using Random Forest Algorithm Based on Kaggle Data Set. International Journal of Recent Technology and Engineering. 2020. Vol. 8, No. 6. pp. 1623–1630.
12. Pregibon, D. (1981). Logistic regression diagnostics. The annals of statistics, 9(4), 705-724.
13. Tin Kam Ho Random decision forests. IEEE Comput. Soc. Press, 1995.
14. VanRossum, G. (1995). Python reference manual. Department of Computer Science [CS], (R 9525).
15. Ş. Öztürk, U. Özkaya, M. Barstuğan, "Classification of Coronavirus (COVID-19) from X-ray and CT images using shrunken features", International journal of imaging systems and technology, 10.1002/ima.22469. Advance online publication (2020).