Міністерство освіти і науки України
Національний університет «Києво-Могилянська академія»
Факультет інформатики
Кафедра інформатики

Магістерська робота

освітній ступінь – магістр

на тему: **«ПОБУДОВА СИСТЕМИ ПРОГНОЗУВАННЯ ЕПІДЕМІЧНОЇ СИТУАЦІЇ, ВИКЛИКАНОЇ COVID-19, НА ОСНОВІ МЕТОДІВ АНАЛІЗУ ЧАСОВИХ РЯДІВ»**

Виконав: студент 2-го року навчання,
Спеціальності
6.050103 Інженерія програмного забезпечення
Киян Максим Євгенович

Керівник Ковалюк Т. В.,
доцент, кандидат технічних наук

Рецензент _____
(прізвище та ініціали)

Магістерська роботи захищена
з оцінкою _____

Секретар ЕК _____
«____» _____ 20___р.

Київ – 2021

Міністерство освіти і науки України

Національний університет «Києво-Могилянська академія»

Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ

доцент, кандидат технічних наук

_____ Ковалюк Т. В.
(підпис)

20 жовтня 2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

на магістерську роботу

студенту Кияну М.Є. факультету інформатики 2 курсу 2 (магістерського) рівня

ТЕМА Побудова системи прогнозування епідемічної ситуації, викликаної COVID-19, на основі методів аналізу часових рядів

Зміст текстової частини до магістерської роботи:

Індивідуальне завдання

Вступ

1. Теоретичні відомості

2. Розробка функціональної і візуальної частини застосунку

3. Аналіз отриманих результатів

Висновки

Список використаних джерел

Дата видачі 20 жовтня 2020 р. Керівник _____
(підпис)

Завдання отримав _____
(підпис)

Тема: Побудова системи прогнозування епідемічної ситуації, викликаної COVID-19, на основі методів аналізу часових рядів

Календарний план виконання роботи:

№ п/п	Назва етапу дипломної роботи	Термін виконання	Примітка
1	Отримання завдання на дипломну роботу.	20.10.2020	
2	Огляд теоретичного матеріалу за темою роботи.	30.11.2020	
3	Вибір алгоритмів прогнозування.	05.02.2021	
4	Теоретичний опис алгоритмів.	25.02.2021	
5	Визначення функціональних вимог до застосунку.	30.02.2021	
6	Розробка дизайну застосунку.	01.03.2021	
7	Пошук статистичних даних та розробка застосунку.	03.04.2021	
8	Написання пояснювальної роботи.	07.05.2021	
9	Створення слайдів для доповіді та написання доповіді.	12.05.2021	
10	Попередній захист роботи.	14.05.2021	
11	Аналіз та корегування роботи з науковим керівником.	15.05.2021	
12	Захист дипломної роботи.	16.06.2021	

ЗМІСТ

АНОТАЦІЯ

- Об'єм роботи – 75 сторінок (26 рисунків, 18 таблиць, 37 формул);
- Кількість використаних джерел – 31;
- Мета роботи – розробка застосунку для прогнозування епідемічної ситуації викликаної захворюванням COVID-19 у різних країнах, у тому числі і України;
- Основні результати роботи – розроблено кінцевий застосунок, у якому реалізовано прогнозування епідемічної ситуації на основі сингулярного спектрального аналізу, поліноміального та лінійного методів найменших квадратів на мові програмування C# з використанням фреймворків ML.NET та Accord.NET, технології WPF для реалізації візуальної частини застосунку;
- Ключові слова: аналіз часових рядів, сингулярний спектральний аналіз, метод найменших квадратів, ML.NET, Accord.NET, прогнозування епідемічних ситуацій, C#, WPF.

ВСТУП

Вже півтора року перед людством стоїть глобальний виклик – пандемія COVID-19, спричинена вірусом SARS-CoV-2, що відноситься до сімейства коронавірусів. Назва пов'язана з будовою віруса, оскільки його шиповидні відростки нагадують сонячну корону.

Спалах хвороби почався у китайському місті Ухань (провінція Хубей). І хоча перші клінічні прояви у хворих з’явились ще 8 грудня 2019 року, перше повідомлення про «пневмонію невідомого походження» китайська влада зробила лише 30 грудня, а на наступний день про це отримала сповіщення Всесвітня організація охорони здоров'я [1]. Також повідомлялось про те, що задля запобігання поширення хвороби влада у місті закрила місцевий оптовий ринок морепродуктів, який згодом буде названий джерелом вірусу. Хоча питання виникнення підіймається й досі. Так, ще 26 травня 2021 року, президент США Джо Байден вже не вперше дав американським спецслужбам 90 днів на те, щоб останні підготувати ще один звіт про походження коронавірусу. 30 травня 2021 року інтерес до теми підігріла британська газета «The Sunday Times», повідомивши про те, що розвідка Великобританії тепер вважає правдоподібною версію про витік коронавірусу з Уханьского інституту вірусології [2].

Але повернувшись у початковий перебіг подій після двох засідань Комітету Всесвітньої організації охорони здоров'я, голова Організації доктор Тедрос Адханом Гебрейесус оголосив епідемію надзвичайною ситуацією міжнародного значення (тоді масштаби захворювання не відповідали означенню пандемії, про яку почнуть говорити пізніше).

13 січня було виявлено перший випадок зараження за межами Китаю – у Тайланді, у жінки з Ухані. Згодом коронавірус почав різко поширюватися в Європі та світі. Одна за одною країни оголошували про виявлення перших заражених на новий тип вірусу. З тих пір COVID-19 заразилися більше 170 млн людей, і майже 3,5 млн померли.

11 березня 2020 року Всесвітня організація охорони здоров'я охарактеризувала світовий масштаб поширення хвороби як пандемію. Цього ж дня українська влада ухвалила рішення про перший карантин на території країни, який буде продовжений декілька разів поспіль, а Україна у цілому перейде у режим посилення та послаблення карантинних обмежень.

З початку світової пандемії перед владою кожної держави щодня постають серйозні питання: чи вводити/послаблювати/знімати карантинний режим, за яких умов можна це робити, якого навантаження лікарень очікувати, коли можна вважати, що громадяни у безпеці, а ризик виникнення нового спалаху епідемії мінімальний тощо. Показовою є ситуація у Китаї: на відміну від інших країн Китай подолав пандемію дуже швидко, і до 30 травня 2021 року у країні діагностувались поодинокі випадки захворювання (що у порівнянні з кількістю населення країни неймовірний показник). Але 30 травня китайська влада помітила підозрілу статистику у місті Гуанчжоу і одразу ж відреагувала. У цей же день мегаполіс з 15-ти мільйонним населенням закрили на карантин.

Іншими словами, для усього світу прогнозування епідемічної ситуації стало дуже актуальним та важливим питанням, адже це необхідна процедура для прийняття рішення щодо запобігання або виходу з такої критичної ситуації як епідемія.

Кожен день оприлюднюються статистичні дані щодо кількості виявлених хворих на COVID-19, кількості летальних випадків та пацієнтів, що одужали. З математичної точки зору ці дані є часовим рядом. Саме завдяки аналізу часових рядів ми можемо описати дані (відобразити на графіку), створити модель, спрогнозувати дані через деякий проміжок часу, що може послужити основою прийняття багатьох рішень.

Метою роботи є розробка застосунку для прогнозування епідемічної ситуації викликаної захворюванням COVID-19 у різних країнах, у тому числі і України.

Об’єктом дослідження є часові ряди.

Методи дослідження – аналіз та комп’ютерне моделювання.

Робота складається зі вступу, 3 розділів та списку використаних джерел з 31 джерела.

У першому розділі описані поняття часового ряду та його аналізу, було обґрунтовано вибір алгоритмів, на основі яких у кінцевому застосунку відбувається прогнозування епідемічної ситуації викликаної захворюванням COVID-19 у різних країнах, у тому числі і України, надано опис цих алгоритмів.

У другому розділі були описані фреймворки для машинного навчання ML.NET та Accord.NET, принципи їх роботи, проаналізовані існуючі рішення для прогнозу COVID-19. Також було визначено функціональні вимоги для майбутнього застосунку, процес отримання та перетворення набору даних для подальшої роботи з ними, опис реалізації функціональної та візуальної частини застосунку згідно до вимог.

У третьому розділі був проведений аналіз отриманих результатів, а саме: описані основні характеристики прогнозування та метрики, які були використані при розрахунку прогнозу в застосунку, надані практичні поради при прогнозуванні. Також було проведено тестування, та приведені метрики прогнозу для таких країн світу як Україна, Індія та Канада, надано прогноз для України на майбутнє.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ВІДОМОСТІ

Прогнозування інфекційної захворюваності активно розвивається з початку XX століття. Число робіт на цю тему стрімко зростає завдяки розгортанню інформаційних систем та появи великих обсягів статистичних даних, доступних для аналізу. Епідемічні прогнози обраховуються для різних термінів, залежно від яких, використовуються для різних цілей. Так, короткостроковий прогноз на кілька тижнів вперед застосовується в оперативному управлінні і при виявленні епідемічних спалахів захворюваності. У випадку пандемії 2020 року, враховуючи, що за думкою науковців вакцина буде розроблюватись та тестуватись впродовж як мінімум 2 років, актуальним є як короткостроковий прогноз, так і довгостроковий [1].

Розглянемо доцільність застосування методів аналізу часових рядів для даних, що характеризують епідемічну ситуацію. Даний підхід полягає в ідентифікації моделі ряду, що описує кореляційну залежність між значеннями та дозволяє врахувати [1]:

1) стохастичний характер вхідних даних;
2) одночасно епідемічний період і період між епідеміями;
3) взаємозв’язок показників та розгляд епідемічної ситуації як єдиного процесу.

Усі реальні процеси у природі, як і епідемічний процес, безумовно є нелінійними динамічними процесами. Проте на практиці частіше використовують лінійне наближення через простоту та високу точність результатів у багатьох ситуаціях. Значення часового ряду, що прогнозуються, у даному випадку представляє собою лінійну функцію поточних та попередніх значень процесу [1].

Отже, можемо зробити висновок, що використовувати часові ряди та їх аналіз для прогнозування епідемічної ситуації є доцільним.

1.1 Поняття часового ряду та його аналізу

Часовий ряд (або ряд динаміки) – зібраний у різні моменти часу статистичний матеріал про значення будь-яких параметрів досліджуваного процесу. Кожна одиниця статистичного матеріалу називається виміром або відліком, також допустимо називати його рівнем у пов’язаний з ним момент часу. У часовому ряді для кожного відліку має бути зазначено час вимірювання або номер вимірювання за порядком. Часовий ряд істотно відрізняється від простої вибірки даних, оскільки при аналізі враховується взаємозв'язок вимірювань з часом, а не тільки статистична різноманітність і статистичні характеристики вибірки [3].

У якості показника часу можуть бути використані або певні моменти часу (дати), або окремі періоди (дні, тижні, місяці, квартали, роки тощо). Таким чином, часові ряди за характером часового параметру можуть бути моментні та інтервальні відповідно [1].

Аналіз часових рядів — сукупність математико-статистичних методів аналізу, призначених для виявлення структури часових рядів і для їх прогнозування. Виявлення структури часового ряду необхідно для того, щоб побудувати математичну модель того явища, яке є джерелом часового ряду [1].

На даний момент розроблено і обґрунтовано багато різних методів для аналізу часових рядів: методи інтерполяції, двох крайніх точок, середніх групових точок, Холта-Уінтерса, Трігга-Ліча, Чоу, з використанням експертних оцінок тощо [4].

Історично першими були розроблені методи, у яких на основі статистичного аналізу пропонувалося використовувати авторегресії, згладжування тощо. Пізніше в рамках нелінійної динаміки були розроблені нові практичні методики [1]:

- сингулярний спектральний аналіз (SSA);
- локальна апроксимація (LA);
- поєднання SSA-LA.

1.2 Вибір методів для прогнозу

При аналізі даних по країнам було виявлено, що часові ряди поводять себе по-різному для різних країн, а також за умов їх різної інтерпретації. Наприклад у Китаї на початку поширення інфекції був великий стрибок кількості хворих, а з плином часу графік стає лінійним; при аналізі поточної кількості всіх хворих часовий ряд загалом змінюється лінійно, а при аналізі кількості виключно нових хворих кожного дня – періодично. У зв’язку з нестандартною природою і поведінкою даних було вирішено використовувати алгоритми для прогнозу універсальних одновимірних часових рядів, без прив’язки до конкретної моделі тренду та періодичності.

Перший метод який було обрано – сингулярний спектральний аналіз (SSA). Суттєва різниця між SSA та більшістю методів аналізу часових рядів полягає в тому, що SSA не вимагає апріорної моделі для тренду, а також апріорних знань про кількість періодичностей та значення періоду. Крім того, періодичність може модулюватися різними способами, і тому тип моделі, адитивний чи мультиплікативний, необов’язково зберігати та враховувати [5]. Ще одним аргументом на його користь є те що спеціалісти з компанії Microsoft, розробники бібліотеки машинного навчання ML.NET рекомендують використовувати для аналізу одновимірних часових рядів саме його [6]. Також він показав досить гарні результати при тестуванні [1].

Другий алгоритм, який використовується в роботі – поліноміальний метод найменших квадратів. Його було обрано за універсальність, популярність, відсутність жорстких вимог до характеристик даних та гарну пристосованість до одновимірних часових рядів.

Третій обраний метод аналізу часових рядів – лінійний метод найменших квадратів. Це, напевно, найпростіший і найпопулярніший з методів, так що без нього система була б неповною. Окрім того звичайний МНК може похизуватися тим, що для його використання не потрібно підбирати жодного параметру, він

ідеально підходить для одновимірних часових рядів і показує стабільний, як правило не сильно відхилений, результат не залежно від характеристик даних.

Також були протестовані наступні методи:

1. Нейромережевий підхід з даними «рухомого-вікна» (з англ. Neural network approach with rolling-window data) [7]. Він виявився не дуже ефективним через характер поведінки даних, сильно залежить від декількох параметрів які дуже складно налаштовувати, відносно довго працює.
2. Алгоритми для загального регресійного аналізу ML. NET [8]:
    - FastForest
    - LbfgsPoissonRegression
    - SDCA
    - FastTreeTweedie
    - FastForest
    - Gam

Вони виявилися неефективними саме для одновимірних часових рядів. Ці методи пристосовані для багатовимірної регресії, тобто коли на цільову функцію впливають багато (більше одного) чинників, які вижарені у відповідних стовпчиках даних в дата сеті.

3. Прогнозування за допомогою авторегресійної моделі ARIMA (з англ. Autoregressive Integrated Moving Average). Було вирішено не використовувати цю модель, тому що вона потребує на вхід 2 параметра, якісний підбір яких є нетривіальною задачею для людини, немає безкоштовних бібліотек для роботи з нею, відносно довго працює.

1.3 Метод найменших квадратів

1.3.1 Загальні положення

Метод найменших квадратів (МНК, англ. LS – Least Squares) – математичний метод, застосовуваний для вирішення різних завдань, заснований на мінімізації

суми квадратів відхилень деяких функцій від шуканих змінних. Він може використовуватися для «вирішення» перевизначених систем рівнянь (коли кількість рівнянь перевищує кількість невідомих), для пошуку рішення в разі звичайних (не перевизначених) нелінійних систем рівнянь, для апроксимації точкових значень деякої функції. МНК є одним з базових методів регресійного аналізу для оцінки невідомих параметрів регресійних моделей за вибірковими даними [9].

Суть методу:

Нехай x – набір n невідомих змінних (параметрів), а $f_i(x), i = 1, \ldots, m, m > n$ – це сукупність функцій від цього набору змінних.

Завдання полягає в підборі таких значень x, щоб значення цих функцій були максимально близькі до деяких значень y_i. По суті мова йде про «вирішення» перевизначеної системи рівнянь $f_i(x) = y_i, i = 1, \ldots, m$ в зазначеному сенсі максимальній близькості лівої і правої частин системи. Суть МНК полягає у виборі в якості «міри близькості» суми квадратів відхилень лівих і правих частин $|f_i(x) - y_i|$. Таким чином, сутність МНК може бути виражена таким чином:

$$\sum_i e_i^2 = \sum_i (y_i - f_i(x))^2 \to \min_x.$$

У разі, якщо система рівнянь має рішення, то найменше значення суми квадратів дорівнюватиме нулю, і можуть бути знайдені точні рішення системи рівнянь аналітично або, наприклад, різними чисельними методами оптимізації. Якщо система перевизначена, тобто, кажучи нестрого, кількість незалежних рівнянь більше кількості шуканих змінних, то система не має точного рішення і метод найменших квадратів дозволяє знайти певний «оптимальний» вектор x в сенсі максимальній близькості векторів y і $f(x)$ або максимальної близькості вектора відхилень e до нуля [9].

МНК у регресійному аналізі

Нехай є n значень деякої змінної y (це можуть бути результати спостережень, експериментів і т. д.) І відповідних змінних x. Завдання полягає в тому, щоб

взаємозв'язок між y і x апроксимувати деякою функцією $f(x,b)$, відомої з точністю до деяких невідомих параметрів b, тобто фактично знайти найкращі значення параметрів b, що максимально наближають значення $f(x,b)$ до фактичних значень y. Фактично це зводиться до випадку «рішення» перевизначеної системи рівнянь щодо b:

$$f(x_t, b) = y_t, t = 1, \ldots, n.$$

У регресійному аналізі використовуються імовірнісні моделі залежності між змінними

$$y_t = f(x_t, b) + \varepsilon_t,$$

де ε_t – так звані випадкові помилки моделі.

Відповідно, відхилення спостережуваних значень y від модельних $f(x,b)$ передбачається вже в самій моделі. Сутність МНК (звичайного, класичного) полягає в тому, щоб знайти такі параметри b, при яких сума квадратів відхилень e_t буде мінімальною:

$$\hat{b}_{OLS} = \arg\min_b RSS(b),$$

де RSS – англ. Residual Sum of Squares визначається як:

$$RSS(b) = e^T e = \sum_{t=1}^{n} e_t^2 = \sum_{t=1}^{n} (y_t - f(x_t, b))^2.$$

У загальному випадку рішення цього завдання може здійснюватися чисельними методами оптимізації (мінімізації). У цьому випадку говорять про нелінійний МНК (NLS або NLLS – англ. Non-Linear Least Squares). У багатьох випадках можна отримати аналітичне рішення. Для вирішення завдання мінімізації необхідно знайти стаціонарні точки функції $RSS(b)$, продиференціювавши її за невідомими параметрами b, прирівнявши похідні до нуля і вирішивши отриману систему рівнянь [9]:

$$\sum_{t=1}^{n} (y_t - f(x_t, b)) \frac{\partial f(x_t, b)}{\partial b} = 0.$$

1.3.2 Поліноміальний метод найменших квадратів

1.3.2.1 Загальний опис

Поліноміальний метод найменших квадратів на сьогоднішній день є найбільш широко застосовуваним методом моделювання. Це те, що має на увазі більшість людей, коли кажуть, що використовували "регресію", "лінійну регресію" або "найменші квадрати", щоб підібрати модель до своїх даних. Лінійна регресія найменших квадратів не тільки є найбільш широко застосовуваним методом моделювання, але вона була адаптована до широкого кола ситуацій, що виходять за межі її прямого обсягу. Поліноміальний МНК відіграє важливу основну роль у багатьох інших методах моделювання, включаючи нелінійний метод найменших квадратів, зважена регресія найменших квадратів та LOESS (з англ. locally estimated scatterplot smoothing) [10].

Використовуючи безпосередньо, з відповідним набором даних, лінійну регресію найменших квадратів можна використовувати для узгодження даних з будь-якою функцією форми

$$f(\vec{x};\vec{\beta}) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots$$

У якій:

1) кожна пояснювальна змінна у функції помножується на невідомий параметр

2) є щонайбільше один невідомий параметр без відповідної пояснювальної змінної

3) всі окремі доданки підсумовуються для отримання кінцевого значення функції.

У статистичному плані будь-яку функцію, яка відповідає цим критеріям, називали б "лінійною функцією". Використовується термін "лінійна", хоча функція може бути не прямою, оскільки якщо невідомі параметри вважаються змінними, а пояснювальні змінні – відомими коефіцієнтами, що відповідають цим "змінним", то проблема може бути представлена у вигляді системи лінійних рівнянь, яка може

бути вирішена для значень невідомих параметрів. Для розмежування різних значень слова "лінійний", лінійні моделі часто називають "лінійними за параметрами" або "статистично лінійними" [10].

Як було сказано вище, лінійні моделі не обмежуються лише прямими або площинами, а включають досить широкий діапазон форм. Наприклад, проста квадратична крива:

$$f(x;\vec{\beta}) = \beta_0 + \beta_1 x + \beta_{11} x^2,$$

прямолінійна модель log(x):

$$f(x;\vec{\beta}) = \beta_0 + \beta_1 \ln(x),$$

поліном sin(x):

$$f(x;\vec{\beta}) = \beta_0 + \beta_1 \sin(x) + \beta_2 \sin(2x) + \beta_3 \sin(3x),$$

є лінійними у статистичному сенсі, оскільки вони є лінійними за параметрами, хоча і не стосовно спостережуваної пояснювальної змінної, x.

1.3.2.2 Матрична форма та розрахунок оцінок

Модель поліноміальної регресії

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \cdots + \beta_m x_i^m + \varepsilon_i \ (i = 1, 2, \ldots, n)$$

може бути виражена у матричній формі через матрицю проектування $\mathbf{X}$, вектор відповіді $\vec{y}$, вектор параметрів $\vec{\beta}$, та вектор $\vec{\varepsilon}$ випадкових помилок. І-й рядок $\mathbf{X}$, та $\vec{y}$ міститиме значення *x* та *y* для і-ї вибірки даних. Тоді модель можна записати як систему лінійних рівнянь

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^m \\ 1 & x_2 & x_2^2 & \cdots & x_2^m \\ 1 & x_3 & x_3^2 & \cdots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^m \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_n \end{bmatrix},$$

яка при використанні чисто матричних позначень записується як

$$\vec{y} = \mathbf{X}\vec{\beta} + \vec{\varepsilon}.$$

Вектор коефіцієнтів оцінок поліноміальної регресії (за допомогою оцінки звичайних найменших квадратів) становить

$$\hat{\vec{\beta}} = (\mathbf{X}^{\mathsf{T}}\mathbf{X})^{-1}\,\mathbf{X}^{\mathsf{T}}\vec{y},$$

Робимо припущення $m < n$, яке потрібно, щоб матриця була оборотною; тоді оскільки $\mathbf{X}$ є матрицею Вандермонде, умова оберненості гарантовано буде виконуватися, якщо всі значення x_i різні, тоді це буде унікальне рішення за допомогою методу найменших квадратів [11].

1.3.2.3 Лінійний метод найменших квадратів

Лінійний МНК за своєю суттю є поліноміальним МНК з поліномом першого ступеню.

Його функція моделі:

$$y = \alpha + \beta x,$$

яка описує пряму з нахилом β та y-відхиленням α. Взагалі такий зв’язок може не виконуватися точно для значною мірою неспостережуваної сукупності значень незалежних та залежних змінних; ми спостерігаємо непомічені відхилення від наведеного рівняння помилками. Припустимо, ми спостерігаємо n пар даних і називаємо їх $\{(x_i, y_i),\ i = 1,\ \ldots,\ n\}$. Ми можемо описати основний зв’язок між y_i та x_i, що включає цей термін помилки ε_i, за допомогою функції

$$y_i = \alpha + \beta x_i + \varepsilon_i.$$

Цей взаємозв'язок між істинними (але неспостережуваними) базовими параметрами α та β та точками даних називається моделлю лінійної регресії.

Мета – знайти приблизні значення $\widehat{\alpha}$ та $\widehat{\beta}$ для параметрів α та β, які підходили б «найкраще» в певному сенсі для точок даних. У методі МНК під "найкращим" підходом розуміється як підхід з найменшими квадратами: рядок, що мінімізує

суму квадратних залишків $\hat{\varepsilon}_i$ (різниці між фактичними та передбачуваними значеннями залежної змінної *y*), кожне з яких задано для будь-яких значень параметрів-кандидатів α та β.

$$\hat{\varepsilon}_i = y_i - \alpha - \beta x_i .$$

Іншими словами, $\widehat{\alpha}$ та $\widehat{\beta}$ вирішають таку проблему мінімізації [12]:

$$\text{Find } \min_{\alpha,\,\beta} Q(\alpha, \beta), \quad \text{for } Q(\alpha, \beta) = \sum_{i=1}^{n} \hat{\varepsilon}_i^{\,2} = \sum_{i=1}^{n} (y_i - \alpha - \beta x_i)^2 .$$

1.3.2.4 Переваги методу найменших квадратів

Лінійна регресія найменших квадратів завоювала своє місце як основний інструмент моделювання процесів завдяки своїй ефективності та повноті.

Хоча існують типи даних, які краще описуються функціями, нелінійними за параметрами, багато процесів у науці та техніці добре описуються лінійними моделями. Це пояснюється або тим, що процеси за своєю суттю є лінійними, або тим, що в коротких межах будь-який процес може бути добре апроксимований лінійною моделлю.

Оцінки невідомих параметрів, отримані за допомогою лінійної регресії найменших квадратів, є оптимальними оцінками з широкого класу можливих оцінок параметрів за звичайних припущень, що використовуються для моделювання процесів. Практично кажучи, лінійна регресія найменших квадратів дуже ефективно використовує дані. Хороші результати можна отримати за відносно невеликих наборів даних.

Нарешті, теорія, пов’язана з лінійною регресією, добре зрозуміла і дозволяє побудувати різні типи легко інтерпретованих статистичних інтервалів для прогнозування, калібрування та оптимізації. Потім ці статистичні інтервали можна використовувати для чітких відповідей на наукові та технічні питання [10].

1.3.2.5 Недоліки методу найменших квадратів

Основними недоліками лінійних найменших квадратів є обмеження у формах, які лінійні моделі можуть приймати на великих відстанях та чутливість до екстремальних значень.

Лінійні моделі з нелінійними членами у прогнозованих змінних змінюються порівняно повільно, тому для нелінійних процесів, що є невід'ємними, стає все важче знайти лінійну модель, яка відповідає даним, оскільки діапазон даних збільшується. Оскільки пояснювальні змінні стають екстремальними, результат лінійної моделі також завжди буде більш екстремальним. Це означає, що лінійні моделі можуть бути неефективними для екстраполяції результатів процесу, для якого неможливо зібрати дані в регіоні, що цікавить. Звичайно, екстраполяція потенційно небезпечна незалежно від типу моделі.

Нарешті, хоча метод найменших квадратів часто дає оптимальні оцінки невідомих параметрів, він дуже чутливий до наявності екстремальних точок у даних, що використовуються для підгонки моделі. Один або два екстремальні значення іноді можуть серйозно перекосити результати аналізу найменших квадратів. Це робить валідацію моделі, особливо стосовно стрибків графіку, критично важливою для отримання обґрунтованих відповідей на питання, що спонукають до побудови моделі [10].

1.4 Сингулярний спектральний аналіз

Сингулярний спектральний аналіз (у російськомовній літературі можна зустріти назву «Гусениця», а з англійської Singular spectrum analysis, далі – SSA) є непараметричний спектральний метод оцінки. Він поєднує у собі елементи класичного аналізу часових рядів, багатовимірного статистичного аналізу, багатовимірної геометрії, динамічних систем і обробки сигналів [13]. Основа методу полягає у перетворенні одновимірного часового ряду у багатовимірний за допомогою однопараметричної зсувної процедури та дослідження отриманої багатовимірної траєкторії за допомогою сингулярного розкладання та відновлення

(апроксимації) часового ряду за вибраним головним компонентом. Метою методу є розробка часового ряду на адитивні складові, що інтерпретуються [1].

При цьому метод не вимагає стаціонарності часового ряду, що аналізується, знання моделі тренду, а також інформації про наявність у часових рядів періодичних складових та їх періодів. При таких слабких припущеннях метод SSA дозволяє розв'язувати різні завдання, наприклад, виділити тренд, виявити періодику, згладити часовий ряд, побудувати повне розкладання часового ряду у суму тренду, періодик і шума. Важливо, що цей непараметричний метод дозволяє отримати результати, часто лише незначно менш точні, ніж багато параметричні методи, у яких використовуються відомі моделі часових рядів [5].

Варто відзначити, що даний метод у 80-х роках минулого століття активно використовувався для аналізу часових рядів, що характеризують стан різних біологічних і гідрологічних систем. Області, у яких SSA може застосовуватися дуже широкі: кліматологія, морська наука, геофізика, машинобудування, обробка зображень, медицина, економетрика тощо. Навіть відомі приклади успішного застосування методу SSA в астрономічних задачах [5]. Науковцями були запропоновані різні модифікації та різні методики SSA, що зараз використовуються у практичних застосуваннях, такі як екстракція тренду, виявлення періодичністей, сезонне коригування, згладжування, зменшення шуму. У той же час, незважаючи на активні дослідження даного методу різними авторами, ряд питань, пов'язаних з вибором його параметрів і інтерпретацією отриманих результатів, залишаються недослідженими [1].

1.4.1 Базовий алгоритм методу

Нехай N > 2. Розглянемо часовий ряд $F = (f_0, \ldots, f_{N-1})$ довжини N. Будемо припускати, що ряд F ненульовий, тобто існує, принаймні, одне i, таке що $f_i \neq 0$. Зазвичай вважається, що $f_i = f(i\Delta)$ для деякої функції f (t), де t – час, а Δ – певний часовий інтервал, однак це не буде грати особливу роль в подальшому, більш того,

числа $0, \ldots, N-1$ можуть бути інтерпретовані не тільки як дискретні моменти часу, але і як деякі мітки, які мають лінійно-впорядковану структуру.

Нумерація значень часового ряду починається з i = 0, а не стандартно з i = 1 тільки через зручності позначень.

Базовий алгоритм складається з двох етапів, що доповнюють один одного, розкладання і відновлення [14].

Перший етап: розкладання

Крок 1. Вкладення

Процедура вкладення переводить вихідний часовий ряд в послідовність багатовимірних векторів.

Нехай L – деяке ціле число (довжина вікна), 1 < L < N.

Процедура вкладення утворює K = N – L + 1 векторів вкладення

$$X_i = (f_{i-1}, \ldots, f_{i+L-2})^{\mathrm{T}}, \quad 1 \le i \le K,$$

які мають розмірність L. Якщо нам потрібно буде підкреслити розмірність X_i, то ми будемо називати їх векторами L-вкладення.

L-Траєкторна матриця (або просто траєкторна матриця) ряду F

$$\mathbf{X} = [X_1 : \ldots : X_K]$$

складається з векторів вкладення в якості стовпців.

Іншими словами, траєкторна матриця – це матриця

$$\mathbf{X} = (x_{ij})_{i,j=1}^{L,K} = \begin{pmatrix} f_0 & f_1 & f_2 & \cdots & f_{K-1} \\ f_1 & f_2 & f_3 & \cdots & f_K \\ f_2 & f_3 & f_4 & \cdots & f_{K+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ f_{L-1} & f_L & f_{L+1} & \cdots & f_{N-1} \end{pmatrix}$$

Очевидно, що $x_{ij} = f_{i+j-2}$ і матриця X має однакові елементи на діагоналях i + j = const. Таким чином, траєкторна матриця є ганкелевою. Існує взаємно-однозначна відповідність між ганкелевими матрицями розмірності L x K і рядами довжини N = L + K – 1.

Крок 2. Сингулярне розкладання

Результатом цього кроку є сингулярне розкладання (SVD = Singular Value Decomposition) траєкторної матриці ряду.

Нехай $\mathbf{S} = \mathbf{X}\mathbf{X}^{\mathrm{T}}$. Позначимо $\lambda_1, \ldots, \lambda_L$ власні числа матриці S, взяті в порядку неспадання $(\lambda_1 \geq \ldots \geq \lambda_L \geq 0)$ і $U_1, \ldots, U_L$ – ортонормовану систему власних векторів матриці S, відповідних власним числам.

Нехай $d = \max\{i : \lambda_i > 0\}$. Якщо позначити $V_i = \mathbf{X}^{\mathrm{T}} U_i / \sqrt{\lambda_i}, \ i = 1, \ldots, d,$, то сингулярне розкладання матриці X може бути записано як

$$\mathbf{X} = \mathbf{X}_1 + \ldots + \mathbf{X}_d, \tag{1.1}$$

де $\mathbf{X}_i = \sqrt{\lambda_i} U_i V_i^{\mathrm{T}}$. Кожна з матриць $\mathbf{X}_i$ має ранг 1. Тому їх можна назвати елементарними матрицями.

Набір $(\sqrt{\lambda_i}, U_i, V_i)$ ми будемо називати і-ю власною трійкою сингулярного розкладання (1.1).

Другий етап: відновлення

Крок 3. Угруповання

На основі розкладання (1.1) процедура угруповання ділить всю множину індексів $\{1, \ldots, d\}$ на m непересічних підмножин $I_1, \ldots, I_m$.

Нехай $I = \{i_1, \ldots, i_p\}$. Тоді результуюча матриця $\mathbf{X}_I$, відповідна групі I, визначається як

$$\mathbf{X}_I = \mathbf{X}_{i_1} + \ldots + \mathbf{X}_{i_p}.$$

Такі матриці обчислюються для $I = I_1, \ldots, I_m,$, тим самим розкладання (1.1) може бути записано в згрупованому вигляді

$$\mathbf{X} = \mathbf{X}_{I_1} + \ldots + \mathbf{X}_{I_m}. \tag{1.2}$$

Процедура вибору множин $I_1, \ldots, I_m$ і називається угрупованням власних трійок.

Крок 4. Діагональне усереднення

На останньому кроці базового алгоритму кожна матриця згрупованого розкладання (1.2) перетворюється на новий ряд довжини N.

Нехай Y – деяка L x K матриця з елементами y_{ij}, де $1 \le i \le L, 1 \le j \le K.$. Покладемо $L^* = \min(L,K),\ K^* = \max(L,K)$ і $N = L + K - 1.$. Нехай $y^*_{ij} = y_{ij}$, якщо L < K, і $y^*_{ij} = y_{ji}$ інакше.

Діагональне усереднення переводить матрицю Y в ряд $g_0, \dots, g_{N-1}$ за формулою

$$g_k = \begin{cases} \dfrac{1}{k+1} \displaystyle\sum_{m=1}^{k+1} y^*_{m,k-m+2} & \text{для } 0 \le k < L^* - 1, \\ \dfrac{1}{L^*} \displaystyle\sum_{m=1}^{L^*} y^*_{m,k-m+2} & \text{для } L^* - 1 \le k < K^*, \\ \dfrac{1}{N-k} \displaystyle\sum_{m=k-K^*+2}^{N-K^*+1} y^*_{m,k-m+2} & \text{для } K^* \le k < N. \end{cases}$$

Цей вираз відповідає усередненню елементів матриці уздовж діагоналей i + j = k + 2: вибір k = 0 дає $g_0 = y_{11}$, для k = 1 отримуємо $g_1 = (y_{12}+y_{21})/2$ і т. д. Зауважимо, що якщо матриця Y є траєкторної матрицею деякого ряду $(h_0, \dots, h_{N-1})$ (іншими словами, якщо матриця Y є ганкелевою), то $g_i = h_i$ для всіх i.

Застосовуючи діагональне усереднення до результуючих матриць $\mathbf{X}_{I_k}$, ми отримуємо ряди $\widetilde{F}^{(k)} = (\tilde{f}_0^{(k)}, \dots, \tilde{f}_{N-1}^{(k)})$, і, отже, вихідний ряд $(f_0, \dots, f_{N-1})$ розкладається в суму m рядів:

$$f_n = \sum_{k=1}^{m} \tilde{f}_n^{(k)}.$$

1.4.2 Роздільність та вибір параметрів

Дуже важливим питанням є те, як вибрати параметри для побудови правильного розкладу спостережуваних часових рядів і коли це можливо. Поняття розділеності дайте відповідь на це питання. Розділеність двох часових рядів $\mathbb{X}_N^{(1)}$ та $\mathbb{X}_N^{(2)}$ означає можливість видобутку $\mathbb{X}_N^{(1)}$ від спостережуваної суми $\mathbb{X}_N^{(1)}$ + $\mathbb{X}_N^{(2)}$. SSA може приблизно відокремлювати, наприклад, сигнал і шум, синусоїди з різною частотою, тенденцією та оригінальністю та інші [15].

Якщо два часові ряди приблизно роздільні, виникає проблема ідентифікації виразів у (1.1), що відповідають $\mathbb{X}_N^{(1)}$. Компоненти часових рядів можна ідентифікувати на основі наступного принципу: форма власного вектора повторює форму компонента часового ряду, який виробляє цей власний вектор. Таким чином, графіки власних векторів можуть допомогти в процесі ідентифікації. Більше того, синусоїда генерує, точно або приблизно, дві складові синусоїди з однаковою частотою та фазовим зсувом π / 2. Отже, діаграма розсіювання пари власних векторів, яка утворює більш-менш регулярний многокутник T-вершин, може допомогти ідентифікувати синусоїду періоду T. Для проблем вилучення сигналу, згладжування та зменшення шуму обрано кілька провідних власних трикутників [5].

Дуже корисна інформація для поділу міститься у так званій матриці w-кореляції. Це матриця, що складається з зважених кореляцій між реконструйованими компонентами часових рядів. Вагові коефіцієнти відображають кількість записів термінів часового ряду в його матрицю траєкторії. Добре розділені компоненти мають малу кореляцію, тоді як погано розділені компоненти мають велику кореляцію. Отже, розглядаючи w-кореляційну матрицю, можна знайти групи корельованих елементарно відновлених рядів і використовувати цю інформацію для подальшого групування. Одне з правил – не включати до різних груп корельовані компоненти [5].

Умови (приблизної) роздільності дають рекомендації щодо вибору довжини вікна L: вона повинна бути досить великою (L ~ N / 2), і якщо ми хочемо витягнути періодичну складову з відомим періодом, то довжини вікон, які діляться на період, забезпечують кращу відокремлюваність. Вибір параметрів більш детально обговорюється в [16]. SSA з малим L виконує згладжування серії фільтром порядку 2L – 1 [17], якщо ми вибрали кілька провідних власних трійок. Як правило, вибір довжини вікна важливий, але результат стабільний щодо невеликих змін L [16].

Якщо часовий ряд має складну структуру, рекомендується так званий послідовний SSA. Послідовний SSA складається з двох етапів, на першому етапі

тенденція витягується з невеликою довжиною вікна, а потім виявляються та витягуються періодичні компоненти із залишку з L ~ N / 2.

Якщо ми використовуємо SSA як без модельну та дослідницьку техніку, то обґрунтування розкладання не є формальним; він базується на теорії розділеності та інтерпретації результатів. Обробка в реальному часі або пакетна обробка за допомогою SSA можлива, якщо клас серій спеціалізований в достатній мірі, що дозволяє нам виправити правило вибору належних параметрів. Для проведення статистичного тестування слід вказати конкретну модель [5].

1.4.3 Алгоритми прогнозування SSA

Далі надано формальний опис алгоритму прогнозування [15].

LLR – лінійна рекурентна формула (з англ. Linear recurrence relation).

1.4.3.1 Рекурентне прогнозування

Нехай I буде обраним набором власних трикутників, $P_i \in \mathsf{R}^L, i \in I,$ – відповідні власні вектори, P_i – їх перші $L-1$ координати, π_i буде останньою координатою для P_i, $\nu^2 = \sum_i \pi_i^2$. Визначимо $R = (a_{L-1}, \ldots, a_1)^{\mathrm{T}}$ як

$$R = \frac{1}{1-\nu^2} \sum_{i \in I} \pi_i \underline{P_i}. \tag{1.3}$$

Алгоритм періодичного прогнозування можна сформулювати наступним чином:

1) Часовий ряд $\mathbb{Y}_{N+M} = (y_1, \ldots, y_{N+M})$ визначається як

$$y_i = \begin{cases} \widetilde{x}_i & \text{for } i = 1, \ldots, N, \\ \sum\limits_{j=1}^{L-1} a_j y_{i-j} & \text{for } i = N+1, \ldots, N+M. \end{cases} \tag{1.4}$$

2) Числа $y_{N+1}, \ldots, y_{N+M}$ утворюють M значень рекурентного прогнозу.

Таким чином, періодичне прогнозування виконується безпосереднім використанням LRR з коефіцієнтами $\{a_j, j = 1, \ldots, L-1\}$.

Визначимо лінійний оператор $\mathcal{P}_{\text{Rec}} : \mathsf{R}^L \mapsto \mathsf{R}^L$ за формулою

$$\mathcal{P}_{\text{Rec}} Y = \begin{pmatrix} \overline{Y} \\ R^{\mathrm{T}} \overline{Y} \end{pmatrix}.$$

Встановимо

$$Z_i = \begin{cases} \widetilde{X}_i & \text{for } i = 1, \ldots, K, \\ \mathcal{P}_{\text{Rec}} Z_{i-1} & \text{for } i = K+1, \ldots, K+M. \end{cases} \tag{1.5}$$

Легко помітити, що матриця $\mathbf{Z} = [Z_1 : \ldots : Z_{K+M}]$ – траєкторна матриця ряду $\mathbb{Y}_{N+M}$. Отже, (1.5) можна розглядати як векторну форму (1.4).

1.4.3.2 Векторне прогнозування

Позначимо $\mathcal{L}_r = \text{span}(P_i, i \in I)$, $\widehat{X}_i$ проекція відсталого вектора X_i на $\mathcal{L}_r$.

Розглянемо матрицю:

$$\Pi = \underline{\mathbf{V}} \underline{\mathbf{V}}^{\mathrm{T}} + (1 - \nu^2) R R^{\mathrm{T}},$$

Де $\underline{\mathbf{V}} = [\underline{P_1} : \ldots : \underline{P_r}]$ та R визначено в (1.3). Матриця Π - це матриця лінійного оператора, який виконує ортогональну проекцію $\mathsf{R}^{L-1} \mapsto \underline{\mathcal{L}_r}$, де $\underline{\mathcal{L}_r} = \text{span}(\underline{P_i}, i \in I)$.

Нарешті, визначимо лінійний оператор $\mathcal{P}_{\text{Vec}} : \mathsf{R}^L \mapsto \mathcal{L}_r$ по формулі

$$\mathcal{P}_{\text{Vec}} Y = \begin{pmatrix} \Pi \overline{Y} \\ R^{\mathrm{T}} \overline{Y} \end{pmatrix}.$$

Сформулюємо векторний алгоритм прогнозування:

1) У позначеннях вище визначте вектори Z_i таким чином:

$$Z_i = \begin{cases} \widehat{X}_i & \text{for } i = 1, \ldots, K, \\ \mathcal{P}_{\text{Vec}} Z_{i-1} & \text{for } i = K+1, \ldots, K+M+L-1. \end{cases}$$

2) Побудувавши матрицю $\mathbf{Z} = [Z_1 : \ldots : Z_{K+M+L-1}]$, і зробивши її усереднення по діагоналі, ми отримаємо ряд $y_1, \ldots, y_{N+M+L-1}$.
3) Числа $y_{N+1}, \ldots, y_{N+M}$ утворюють M значень векторного прогнозу.

У періодичному прогнозуванні ми виконуємо усереднення по діагоналі, щоб отримати реконструйований ряд, а потім застосовуємо LRR. У векторному прогнозуванні ці кроки в певному сенсі взаємозамінні. Як правило, векторний прогноз трохи стабільніший, але він має набагато більші обчислювальні витрати, ніж періодичний прогноз [5].

Якщо складова часового ряду відокремлена від залишку і регулюється LRR, як періодичне, так і векторне прогнозування збігаються і забезпечують точне продовження. У випадку приблизної відокремлюваності ми отримуємо приблизне продовження [5].

1.4.3.3 Модель та вибір параметрів для прогнозування

Хоча для аналізу SSA зазвичай не потрібна модель, для прогнозування SSA модель потрібна. Модель детермінованого ряду, що допускає прогнозування SSA, є сигналом, який приблизно регулюється лінійним відношенням рекурентності. Прогнозування SSA стосується суми сигналу та залишку (можливо, шуму), які повинні бути приблизно розділені за допомогою SSA. Нам не слід точно вказувати модель перед проведенням аналізу SSA; розмірність сигналу та керуюча LRR можуть бути побудовані за допомогою самого аналізу SSA. Пов'язане статистичне тестування побудованої моделі може проводитися методами, які не є специфічними для SSA [5].

Основні правила вибору параметрів при прогнозуванні, як правило, такі ж, як і для реконструкції. Істотна різниця полягає в тому, що для прогнозування більш стабільна реконструкція може бути навіть важливішою, ніж точність

реконструкції. Крім того, моделювання та теорія показують, що краще вибирати довжину вікна L менше половини довжини часового ряду N [16]. Одне з рекомендованих значень – N / 3 [1].

Як правило, рекомендації є дійсними, якщо ряд приблизно відповідає моделі шумних часових рядів, що регулюється LRR. Реальні часові ряди завжди потребують додаткового аналізу [5].

Якщо часовий ряд $\mathbb{X}_N$ довгий і має стабільну структуру, можна застосувати техніку ковзаючих (sliding) прогнозів. Ми можемо вибрати довжину N_s ковзних підрядів, зафіксувати довжину вікна L та групу індексів I, вибрати горизонт прогнозування, а потім виконати прогнози підряду $\mathbb{X}_{i,i+N_s-1} = (x_i, \ldots, x_{i+N_s-1})$, $i = 1, \ldots, N - N_s$. Правильний вибір L та I відповідає невеликій середній середньоквадратичній похибці (MSE) прогнозів. Вибір параметрів, що дозволяють отримати мінімальну точність, є не найкращим, оскільки, наприклад, стабільність щодо невеликих змін довжини вікна може бути більш важливою. Для перевірки стабільності прогнозів також можуть бути корисними інтервали впевненості (confidence intervals) [5].

Якщо часовий ряд довгий, але його структура може змінюватися в часі, то оцінка точності прогнозу може допомогти зрозуміти, скільки останніх значень часового ряду слід враховувати для прогнозування [1].

РОЗДІЛ 2. РОЗРОБКА ФУНКЦІОНАЛЬНОЇ І ВІЗУАЛЬНОЇ ЧАСТИНИ ЗАСТОСУНКУ

2.1 Обрані технології

Для реалізації програмного застосунку, що має прогнозувати, а також відображати епідемічну ситуацію, що викликана захворюванням COVID-19 у різних країнах, було обрано мову програмування C#, з інтерфейсом на технології WPF (Windows Presentation Foundation), а також модульний фреймворк для розробки з відкритим вихідним кодом – .NET Core. Для роботи з часовими рядами було убрано популярні фреймворки ML.NET (Machine Learning .NET) та Accord.NET.

2.1.1 ML.NET та принципи роботи

У 2018 році компанія Microsoft вперше представила ML.NET – безкоштовний, кросплатфомений і відкритий фреймворк машинного навчання. Звіт Microsoft про машинне навчання за допомогою ML.NET продемонстрував, що він здатний навіть тренувати моделі аналізу настроїв, використовуючи великі набори даних, досягаючи високої точності. Результати свідчать про 95% точність на даних 9GB огляду Amazon. Оскільки, однією з можливостей ML.NET є саме реалізація сингулярного спектрального аналізу, було вирішено використати даний фреймворк [1].

Як було сказано вище, ML.NET – це кросплатформна середа машинного навчання з відкритим вихідним кодом (Windows, Linux, macOS) для розробників .NET. Вона дозволяє додавати у застосунки .NET можливості машинного навчання в автономному і підключеному режимах. Використовуючи цю функцію, можна отримувати автоматичні прогнози на основі даних, що доступні застосунку. Додатки машинного навчання використовують для прогнозування закономірностей, знайдених у даних, не будучи явно запрограмованими [18].

У основі ML.NET лежить модель машинного навчання. Ця модель визначає кроки, які необхідно виконати для отримання прогнозів на основі вхідних даних. За допомогою ML.NET можна навчити призначену для користувача модель, вказавши відповідний алгоритм, а також імпортувати попередньо навчені моделі TensorFlow і ONNX [1].

Створену модель можна додати в застосунок і використовувати її для отримання прогнозів.

ML.NET працює у Windows, Linux і macOS з .NET Core або в Windows з .NET Framework. 64-розрядна версія підтримується на всіх платформах, а 32-розрядна версія підтримується у Windows, за винятком функцій, пов'язаних з TensorFlow, LightGBM та ONNX.

ML.NET пропонує Model Builder (простий інструмент для користувача інтерфейсу) та інтерфейс командного рядка, створені для того, щоб спростити створення призначених для користувача моделей ML з використанням AutoML [19]. За допомогою ML.NET можна отримувати прогнози наступних типів [1]:

- Класифікація/категоризація (автоматичний поділ відгуків клієнтів на позитивні і негативні);
- Регресія/прогноз безперервних значень (прогноз ціни на будинки, залежно від їх розміру і місцезнаходження);
- Виявлення аномалій (виявлення шахрайських банківських операцій);
- Рекомендації (пропозиція продуктів, які онлайн-покупці можуть захотіти купити, на основі їх попередніх покупок);
- Часові ряди/послідовності (прогнози погоди і обсягів продажів);
- Класифікація зображень (класифікація патологій на медичних зображеннях).

2.1.1.1 Методи ML.NET для роботи з часовими рядами

1) DetectAnomalyBySrCnn – виявляє аномалії часових рядів за допомогою алгоритму SRCNN.

2) DetectChangePointBySsa – прогнозує точки зміни в часових рядах за допомогою сингулярного спектрального аналізу (SSA).
3) DetectEntireAnomalyBySrCnn – виявляє аномалії часових рядів для всього входу даних за допомогою алгоритму SRCNN.
4) DetectIidChangePoint – прогнозує точки змін у незалежному ідентично розподіленому (i.i.d.) часовому ряді на основі адаптивних оцінок щільності ядра та балів Мартінгала.
5) DetectIidSpike – прогнозує стрибки в незалежних ідентично розподілених (i.i.d.) часових рядах на основі адаптивних оцінок щільності ядра та балів Мартінгала.
6) DetectSeasonality – У даних часових рядів сезонність (або періодичність) – це наявність змін, які відбуваються через певні регулярні проміжки часу, наприклад, щотижня, щомісяця чи кварталу.

   Цей метод виявляє цей передбачуваний інтервал (або період), використовуючи методи аналізу Фур'є. Припускаючи, що вхідні значення мають однаковий інтервал часу (наприклад, дані датчиків, зібрані кожну секунду, упорядковані за мітками часу), цей метод бере список даних часових рядів і повертає регулярний період для вхідних сезонних даних, якщо передбачувана коливання або можна знайти шаблон, який повторюється або повторюється протягом цього періоду протягом вхідних значень.

   Повертає -1, якщо такого шаблону не знайдено, тобто введені значення не відповідають сезонним коливанням.
7) DetectSpikeBySsa – прогнозує стрибки часових рядів за допомогою аналізу сингулярного спектра (SSA)
8) ForecastBySsa – використовує модель сингулярного спектрального аналізу (SSA) для одновимірного прогнозування часових рядів. Цей метод ідеально підходить для випадку прогнозування даних з COVID-19, адже вони представляють собою одновимірні часові ряди.
9) LocalizeRootCause – локалізує першопричини за допомогою алгоритму дерева рішень.

10) LocalizeRootCauses – виводить упорядкований список RootCauses. Порядок відповідає тому, яка підготовлена причина, швидше за все, буде основною.

2.1.2 Accord.NET та принципи роботи

Accord.NET – це фреймворк для наукових обчислень у .NET. Вихідний код проекту доступний на умовах Gnu Lesser Public License, версія 2.1.

Фреймворк містить набір бібліотек, доступних як вихідний код, а також через виконувані інсталятори та пакети NuGet. Основні сфери охоплюють числову лінійну алгебру, чисельну оптимізацію, статистику, машинне навчання, штучні нейронні мережі, обробку сигналів та зображень та бібліотеки підтримки (такі як побудова графіків та візуалізація). Проект спочатку був створений для розширення можливостей AForge.NET Framework, але з тих пір включив AForge.NET всередину себе. Новіші випуски об’єднали обидва фреймворки під назвою Accord.NET.

Accord.NET Framework був представлений у багатьох книгах, таких як «Mastering.NET Machine Learning» від публікації PACKT та «F# для програм машинного навчання», представлений у QCON San Francisco, і наразі накопичує понад 1500 форків у GitHub. Багато наукових публікацій було опубліковано з використанням цього фреймворку [20].

Оскільки, однією з можливостей Accord.NET є саме реалізація поліноміального методу найменших квадратів, було вирішено використати даний фреймворк.

Фреймворк можна використовувати в Microsoft Windows, Xamarin, Unity3D, додатках Windows Store, Linux або мобільних пристроях.

Після об’єднання з проектом AForge.NET фреймворк тепер пропонує уніфікований API для моделей машинного навчання, який є розширюваним та простим у використанні. Він базується на наступній схемі [21]:

1) Оберіть алгоритм навчання, який забезпечує метод Learn (x, y) або Learn (x);
2) Використовуйте функцію Learn(x, y) для створення моделі машинного навчання, вивченої з даних;

3) Використовуйте моделі Transform, Decide, Scores, Probabilities або методи LogLikelihoods.

Для роботи з часовими рядами призначена частина фреймворка, яка називається *Accord.Statistics*.

2.2 Аналіз існуючих рішень

Кожна країна світу зараз перебуває у напрузі від постійного моніторингу епідемічної ситуації, встановлення чи змінення карантинних режимів тощо. Не дивлячись на те, що з моменту як у Китаї був виявлений новий тип вірусу SARS-CoV-2 пройшло вже півтора року, тим не менш Всесвітня організація охорони здоров'я (далі ВООЗ) й досі підкреслює, що недоступність даних ускладнює роботу з вивчення вірусу, його поширення та можливої мутації. Так, 28 грудня 2020 року, на онлайн-прес-конференції, що транслювалася з Женеви, генеральний директор ВООЗ Тедрос Адханом Гебрейесус наголосив, що ВООЗ співпрацює з науковцями з усього світу, і закликав до прозорого обміну інформацією між окремими країнами, а також з органом, координуючим міжнародну роботу в галузі охорони здоров'я у рамках системи ООН [22].

Отже, на жаль, питання доступності як мінімум статистичних даних є відкритим не тільки для пересічних людей, представників бізнесу, для яких данні можуть бути орієнтиром для руху, державних діячів, а навіть й для представників всесвітньої наукової спілки. Розглянемо найбільш популярні та доступні ресурси, що надають інформацію про стан коронавірусної інфекції.

Більшість популярних ресурсів надають лише обмежену вибірку даних. Так, наприклад Google, використовуючи дані з репозиторію «JHU CSSE COVID-19 Data», що належить приватному дослідницькому університету Джона Хопкінса (Балтимор, США), графічно показує дані по кожній країні, але лише або нові випадки, або кількість померлих [23]. До того ж, часові періоди також не є універсальними. Користувач може обрати лише один з наступних часових періодів «Весь час / Один тиждень / Два тижні / 30 днів». Але, найбільш великим недоліком

є повна відсутність прогнозування. Тобто, навіть такий лідер у цифрових рішеннях як Google представляє дуже обмежений функціонал (зображено на Рис. 2.1 справа)

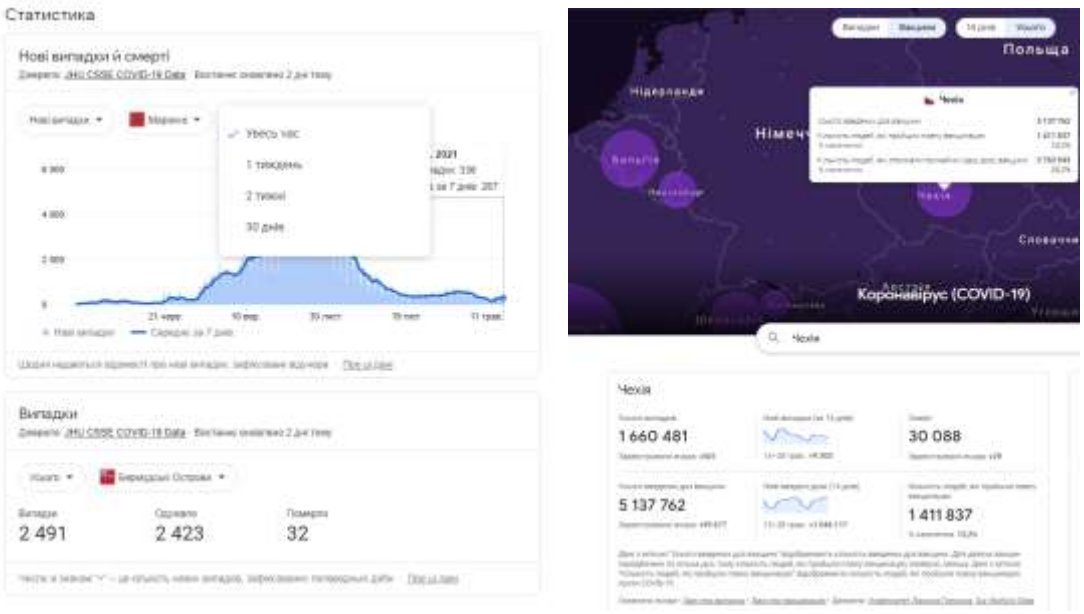

Рисунок 2.1 Приклад візуалізації статистичних даних від Google

Проте, варто зазначити, що серед багатьох недоліків Google, це один з небагатьох сервісів, що показує дані про кількість вакцинованих (див. Рис. зліва).

Мабуть, найбільш повним корисним конкурентноздатним сервісом є рішення від «Our World in Data» [24]. Загалом, це наукове інтернет-видання, яке зосереджується на великих глобальних проблемах, таких як бідність, хвороби, голод, зміни клімату, війни тощо. Це один з проектів «Global Change Data Lab», зареєстрований благодійною організацією у Англії та Уельсі, і заснований Максом Розером, соціальним істориком та економістом розвитку. Команда дослідників базується в Оксфордському університеті.

Розділ присвячений коронавірусу є дуже насиченим, можна подивитись окремі статистики за кількістю захворювань, госпіталізацій, смертей, можна повну статистику по країні, також окрема сторінка присвячена вакцинації, кількісті зроблених тестів кожного дня тощо. Насправді, для пересічного громадянина зрозуміти ці графіки досить складно. Перевагами цього ресурсу є можливість подивитись кожне подання даних у вигляді таблиці та графіку, завантажити дані,

сортувати дані, обрати дані для довільного часового періоду, і навіть вставити будь-який графік на особистий сайт. Але не дивлячись на велику кількість реалізованого функціоналу, ресурс «Our World in Data» не має жодних елементів прогнозування (Рис. 2.2):

Рисунок 2.2 Приклад роботи сервісу «Our World in Data»

Насправді, є ще дуже багато веб-ресурсів, які хоч і у значно менших масштабах, але відображають статистичні дані поточної пандемії. З прогнозуванням ситуація на ринку значно гірша. Є лише декілька рішень, що пропонують користувачам не тільки зручне подання статистичних даних, а й прогнозування.

Так, одне з рішень є продукт COVID-19 Pojections від Інституту метрик і оцінки здоров'я, що працює у галузі глобальної статистики охорони здоров'я та оцінки впливу при Вашингтонському університеті в Сіетлі, США, створеному завдяки фінансовому гранту Фонду Білла і Мелінди Гейтс [25].

Розробники застосунку відображають 4 окремих графіка для статистичних даних відносно загальної та щоденної кількості летальних випадків через коронавірусну інфекцію, кількості госпіталізованих, та заражених. Для кожного графіка наводиться прогнозовані дані на наступні 4 місяці від поточної дати, але користувач може обрати будь-який часовий період менший за 4 місяці.

У наукових публікаціях дослідники з Інституту метрик і оцінки здоров'я вказують, що розглянули 384 опублікованих та неопублікованих моделей

прогнозування COVID-19 та оцінили сім моделей, для яких можна було завантажити загальнодоступні оцінки смертності у різних країнах. Сюди входили моделі, змодельовані: DELPHI-MIT (Delphi), Youyang Gu (YYG), Національна лабораторія Лос-Аламоса (LANL), Імперський коледж Лондона (Imperial) та три власні розроблені моделі, а також модель підгонки кривої (IHME-CF), модель підгонки гібридної кривої та епідеміологічного відсіку (IHME-CF SEIR) та модель сплайну гібридної смертності та епідеміологічного відсіку (IHME – MS SEIR). Але, математичне обґрунтування методів та безпосередньо методи, за яким проводиться прогнозування у COVID-19 Pojections не вказується (Рис. 2.3):

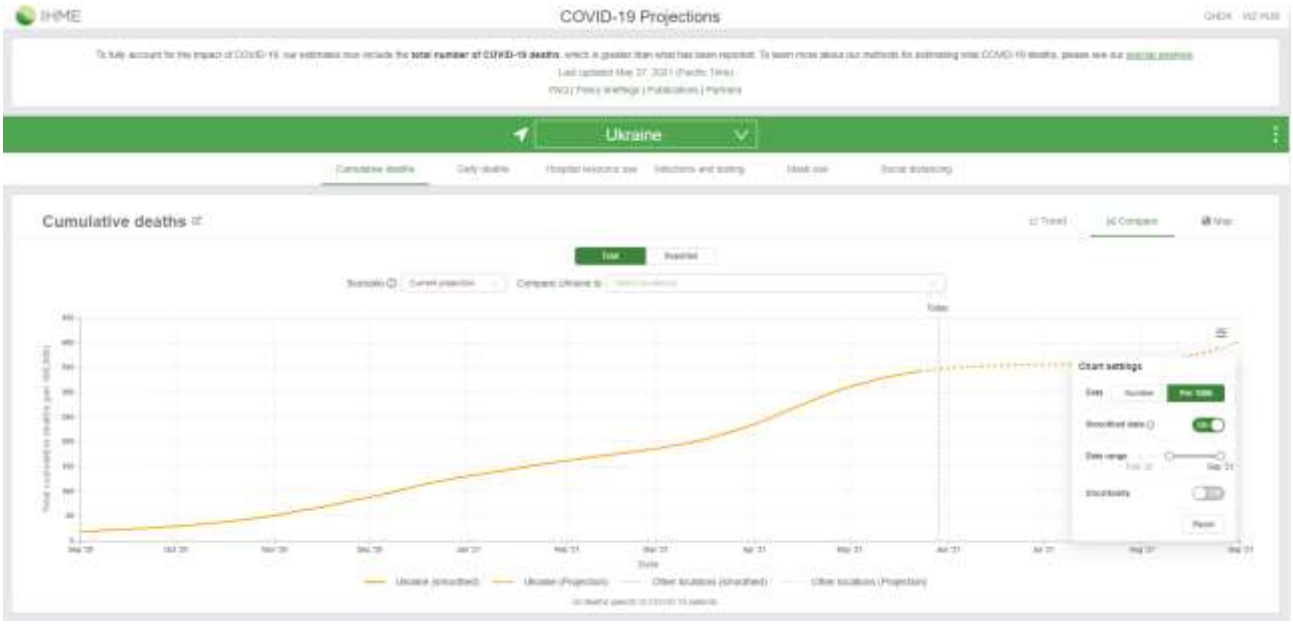

Рисунок 2.3 Приклад роботи COVID-19 Pojections

Ще одним досить популярним ресурсом є рішення від світової компанії IEM. Загалом компанія займається визначенням та створенням інноваційних рішень та інструментів для підвищення реакції на медичні наслідки катастроф, біологічних атак та пандемій, а також прагне допомагати федеральним, місцевим та територіальним урядам, органам охорони здоров'я. Рішення має дуже обмежений функціонал – відображає лише деякі дані і тільки у межах США [26]. Проте є прогнозування на 7 днів вперед за допомогою штучного інтелекту, але математичне обґрунтування роботи не вказується (Рис. 2.4):

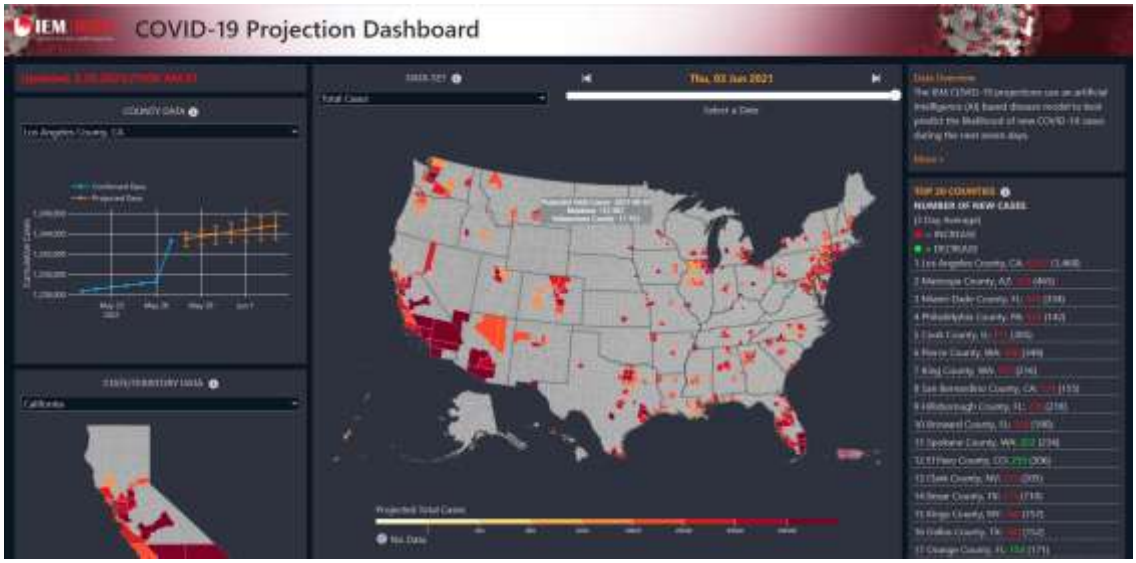

Рисунок 2.4 Приклад роботи рішення від IEM

Якщо брати до уваги опенсорсні аналоги, то не можна не відмітити один популярний застосунок, розташований на Kaggle (платформа для змагань з аналітики та прогнозування, у рамках якого статисти та аналітики даних конкурують у створенні найкращих моделей для прогнозування та опису даних, запропонованих компаніями або користувачами). Відмінність цього застосунку є наведені моделі даних, за допомогою яких система робить прогнозування, а саме: метод опорних векторів, поліноміальна регресія та Баєсова лінійна регресія [27]. Сам застосунок написаний мовою програмування Python, у якому реалізація даних методів вже є вбудованою. Недоліками даного рішення є відсутність зручного інтерфейсу для користування та насправді більшості можливостей, що є у попередніх прикладах, а також у рішеннях, що пропонують лише статистичне відображення (відображення інфографіки за країною, сортування у межах типу даних – нові випадки, хворі, летальні випадки, відображення за часовим інтервалом тощо).

Отже, аналіз існуючих рішень показав, що переважна більшість застосунків пов’язаних зі статистичними даними щодо коронавірусної інфекції у найкращому випадку лише частково відображають дані у їх зручному виді (зазвичай, сервіси не об’єднують статистичні дані з різних джерел, тому часто буває, коли наприклад по

одній країні на одному ресурсі є повна інформація, а на іншому дані зовсім відсутні, це пояснюється саме джерелом даних). Рішення з функціями прогнозування майже не представлені на ринку, і навіть наявні мають недоліки – або прогнозування лише у дуже обмеженому часовому періоді (7 днів як у рішенні від IEM, чи не більше 4 місяців як у COVID-19 Pojections). Мабуть, найкращим рішенням з доступними функціями прогнозування є COVID-19 Pojections від Інституту метрик і оцінки здоров'я, створеному завдяки гранту Фонду сім'ї Гейтса, проте дане рішення не є дуже зручним для користувача (неможливо об'єднати дані з таблиць, відобразити дані в будь-якому іншому форматі). З математичної точки зору оцінити застосунки також важко, оскільки переважно вони пропонують готові рішення, не пояснюючи, які моделі були використані та які параметри впливають на прогнозування (не говорячи вже про можливість зміни цих параметрів), про оцінку прогнозування не йде навіть мова.

Можна зробити висновок, що розробка застосунку, що об'єднає існуючі зручні застосунки з відображенням статистичних даних із сортуванням, зміною типу графічного подання, фільтрацією за багатьма факторами, з елементами прогнозування на не лише короткочасні періоди, а також можливістю впливати на математичне прогнозування, можливо мати декілька моделей для прогнозування задля більшого розуміння отриманих даних є дуже актуальною задачею.

2.3 Визначення функціональних вимог

На даному етапі роботи було визначено наступні вимоги до функціоналу:

1. Автоматичне оновлення статистичних даних з публічних інтернет ресурсів.
2. Завантаження користувачем статистичних даних у .csv форматі.
3. Відображення на карті світу даних щодо кількості хворих, одужавших та померлих за останній день з набору даних по кожній країні.

4. Вибір представлення даних, з варіантами вибору: «Нові випадки кожного дня», «Накопичувальні данні» та «Накопичувальні данні з кількістю поточних хворих».
5. Вибір алгоритму для прогнозування, з варіантами вибору: «Сингулярний спектральний аналіз», «Поліноміальний метод найменших квадратів» та «Лінійний метод найменших квадратів».
6. Вибір країни для подальших обрахунків та прогнозування.
7. Відображення на графіку кількості хворих, одужавших та померлих за увесь час вибраної країни.
8. Можливість відображати лише дані окремо хворих, окремо одужавших, окремо померлих на графіку.
9. Можливість обрізати вибрану кількість днів від початку пандемії у вибраної країни.
10. Можливість подивитися кількість днів між обраними двома точками даних на графіку.
11. Побудова кругової діаграми за вибраний день.
12. Прогноз даних з наступними введеними параметрами: кількість днів та параметр розміру вікна для алгоритму SSA.
13. Прогноз даних з наступними введеними параметрами: кількість днів та параметр розміру полінома для алгоритму поліноміальний МНК.
14. Прогноз даних за введеною кількістю днів для лінійного МНК.
15. Тестування прогнозу на основі порівняння з реальними даними з датасету (параметри аналогічні попереднім вимогам для кожного з алгоритмів).
16. Можливість автоматичного знаходження найкращого значення параметру прогнозу для тестування для алгоритмів які вимагають параметр.
17. Відображення реальних, спрогнозованих, а також спрогнозованих для тестування даних на графіку.
18. Обрахунок метрик тестового прогнозу.

19.Можливість очистити поточні прогнозовані данні на графіку.

2.4 Обробка статистичних даних

Одним з важливих етапів роботи над створенням кінцевого застосунку був пошук релевантних даних щодо кількості виявлених хворих на COVID-19, кількість летальних випадків та пацієнтів, що одужали у кожній країні світу. Було обрано набір даних на одному з найпопулярніших порталів у сфері Data Science – Kaggle [28]. Загалом, Kaggle містить більше ніж 19 000 датасетів.

Прямого шляху завантаження набору даних, на жаль, немає. Kaggle дозволяє підключитись до даних через публічний API. Для цього треба бути зареєстрованим користувачем платформи та авторизуватись через API (Authentication Header) [1].

Kaggle дозволяє завантажити потрібний набір даних лише у .zip архіві. Тому потрібно програмно розархівувати файл, як результат буде отриманий .csv файл. На рисунку 2.5 приведений метод DownloadDataFromKaggle(), у якому описані вищенаведені кроки отримання даних з платформи Kaggle:

```
public static void DownloadDataFromKaggle()
{
    const string dataFileName = "covid_19_clean_complete.csv";
    var auth = new { Username = string.Empty, Key = string.Empty };
    auth = JsonConvert.DeserializeAnonymousType(File.ReadAllText("kaggle.json"), auth);
    var authToken = Convert.ToBase64String(Encoding.ASCII.GetBytes(string.Format($"{auth.Username}:{auth.Key}", auth)));
    var client = new HttpClient();
    client.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Basic", authToken);
    var archiveStreamTask = client
        .GetStreamAsync("https://www.kaggle.com/api/v1/datasets/download/imdevskp/corona-virus-report/"
        + dataFileName);
    const string archiveName = "covid_19_clean_complete.zip";
    using (var stream = archiveStreamTask.Result)
    using (var output = new FileStream(archiveName, FileMode.OpenOrCreate))
    {
        stream.CopyTo(output);
    }

    using (ZipArchive archive = ZipFile.OpenRead(archiveName))
    {
        archive.Entries.First().ExtractToFile(dataFileName, true);
    }
}
```

Рисунок 2.5 – Метод DownloadDataFromKaggle

Наступним етапом роботи було перетворення (розпарс) даних з отриманого .csv файлу. Для цього було використано бібліотеку TinyCsvParser.

Її було використано в обгортці відомого патерну програмування – фабрика, для створення класу (Рисунок 2.6) [1]:

```
public class CsvCoronaParserFactory
{
    1 reference
    public static CsvParser<CoronaData> CreateParser()
    {
        var parserOptions = new CsvParserOptions(true, ',');
        var mapper = new CsvCoronaMapper();
        var parser = new CsvParser<CoronaData>(parserOptions, mapper);

        return parser;
    }

    2 references
    private class CsvCoronaMapper : CsvMapping<CoronaData>
    {
        1 reference
        public CsvCoronaMapper()
        {
            MapProperty(1, m => m.Country);
            MapProperty(4, m => m.CoronaDate, new DateTimeConverter("M/d/y"));
            MapProperty(5, m => m.CoronaConfirmed);
            MapProperty(6, m => m.CoronaDeaths);
            MapProperty(7, m => m.CoronaRecovered);
        }
    }
}
```

Рисунок 2.6 – Клас CsvCoronaParserFactory

Дані з Kaggle були надані Центром Системної Науки та Інженерії при Університеті ім. Джона Хопкінса [23], та гарно відформатовані для роботи з ними користувачами Kaggle [1].

Нажаль, датасет з Kaggle перестав оновлюватися починаючи з 07.08.2020. Для того щоб вирішити цю проблему, було реалізовано метод для завантаження даних напряму з Github Університета ім. Джона Хопкінса. Там данні розбиті на окремі файли, а також представлені у вигляді для мінімізації їх об’єму – за кожен новий день дані додаються окремим стовпцем, і тому їх довилося досить важко перетворювати.

Алгоритм завантаження та перетворення даних з Github Університета ім. Джона Хопкінса:

1) Завантажуємо дані за хворими, померлими та одужавшими розбиті на 3 окремих файли (Рисунок 2.7).

```
2 references | Maxim Kiyan, 9 days ago | 1 author, 1 change
public static void DownloadDataFromGithub()
{
    using var webClient = new WebClient();
    webClient.DownloadFile("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/
      csse_covid_19_time_series/time_series_covid19_confirmed_global.csv", "Confirmed.csv");
    webClient.DownloadFile("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/
      csse_covid_19_time_series/time_series_covid19_deaths_global.csv", "Deaths.csv");
    webClient.DownloadFile("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/
      csse_covid_19_time_series/time_series_covid19_recovered_global.csv", "Recovered.csv");
}
```

Рисунок 2.7 – Функція DownloadDataFromGithub

2) Через те, що дати в датасеті використовуються як окремі стовпці, автоматично перетворити його можна тільки в масив рядків (Рисунок 2.8):

```
CsvParserOptions csvParserOptions = new CsvParserOptions(false, ',');
CsvStringArrayMapping csvMapper = new CsvStringArrayMapping();
CsvParser<string[]> csvParser = new CsvParser<string[]>(csvParserOptions, csvMapper);

var confirmedParsed = csvParser
    .ReadFromFile(confirmedFileName, Encoding.UTF8)
    .ToArray();
var deathsParsed = csvParser
    .ReadFromFile(deathsFileName, Encoding.UTF8)
    .ToArray();
var recoveredParsed = csvParser
    .ReadFromFile(recoveredFileName, Encoding.UTF8)
    .ToArray();
```

Рисунок 2.8 – Код перетворення .csv файлів в масиви рядків

3) Через те що в файлах різна кількість рядків (бо не всім містам надані всі 3 типи даних), їх треба нормалізувати окремо (Рисунок 2.9):

```
3 references | Maxim Kiyan, 7 days ago | 1 author, 2 changes
private static Dictionary<string, CoronaData[]> NormalizeCoronaData(
    CsvMappingResult<string[]>[] parsedData, string propertyToSetName)
{
    var propertyToSet = typeof(CoronaData).GetProperty(propertyToSetName);
    var header = parsedData.First();
    var coronaData = new List<CoronaData>((parsedData.Length - 1) * (header.Result.Length - 4));
    for (int i = 1; i < parsedData.Length; i++)
    {
        var parsedResult = parsedData[i].Result;
        for (int j = 4; j < parsedResult.Length; j++)
        {
            var cd = new CoronaData
            {
                Country = parsedResult[1],
                CoronaDate = DateTime.Parse(header.Result[j])
            };
            propertyToSet.SetValue(cd, float.Parse(parsedResult[j]));
            coronaData.Add(cd);
        }
    }

    var countries = coronaData.Select(c => c.Country).Distinct().ToArray();
    Array.Sort(countries);
    var countryToCoronaDataDic = new Dictionary<string, CoronaData[]>(countries.Length);
    foreach (var country in countries)
    {
        var coronaDataSquashedByDate = coronaData
            .Where(c => c.Country == country)
            .GroupBy(data => data.CoronaDate)
            .Select(groupedData =>
            {
                var cd = new CoronaData
                {
                    CoronaDate = groupedData.Key,
                    Country = groupedData.First().Country
                };
                propertyToSet.SetValue(cd, groupedData.Sum(item => (float)propertyToSet.GetValue(item)));
                return cd;
            })
            .ToArray();

        Array.Sort(coronaDataSquashedByDate, (el1, el2) => el1.CoronaDate.CompareTo(el2.CoronaDate));

        countryToCoronaDataDic.Add(country, coronaDataSquashedByDate);
    }

    return countryToCoronaDataDic;
}
```

Рисунок 2.9 – Функція NormalizeCoronaData

4) І тільки потім нормалізовані данні можна зібрати в єдину структуру даних, зручну для оперування ними – словник (Dictionary) у якого в якості ключей – країни, а в якості значень – масив даних по кількості хворих, померлих та одужавших за кожен день пандемії.

2.5 Реалізація прогнозування часових рядів

Згідно вимог до функціональної частини застосунку на мові програмування C# [29], з використанням платформи .NET Core і фреймворку ML.NET було реалізовано ряд методів.

Для того, щоб статистичні дані взаємодіяли з моделями ML.NET і Accord.Statistics, та надавали зручний API для прогнозування даних з COVID-19, було написано клас TimeSeriesAnalyzer з методом підгрузки даних до моделі, а також найважливішими методами створення та тренування моделей за наявними даними і прогнозування на їх основі.

Завантаження даних до моделі було реалізовано у конструкторі класу (Рисунок 2.10):

```
4 references | Maxim Kiyan, 3 days ago | 1 author, 3 changes
public TimeSeriesAnalyzer(CoronaData[] coronaDataForTraining, string predictionAlgorithm, int? daysToIgnore = null)
{
    _predictionAlgorithm = predictionAlgorithm;
    CoronaDataForTraining = daysToIgnore.HasValue ?
        coronaDataForTraining.Take(coronaDataForTraining.Length - daysToIgnore.Value).ToArray()
        : coronaDataForTraining;

    if (predictionAlgorithm == AlgConstants.SSA)
    {
        _mlContext = new MLContext();
        _dataToTrainLength = CoronaDataForTraining.Length;
        _coronaDataView = _mlContext.Data.LoadFromEnumerable(CoronaDataForTraining);
    }
}
```

Рисунок 2.10 – Конструктор класу TimeSeriesAnalyzer

Для організації єдиного зручного інтерфейсу для прогнозування даних пандемії, клас має публічний метод PredictCoronaUnits (Рисунок 2.11), який дозволяє спрогнозувати обраний тип даних за обраним алгоритмом на введену кількість днів, використовуючи відповідні методи прогнозу за обраним алгоритмом:

```
public CoronaTimeSeriesPrediction PredictCoronaUnits(int daysToPredict, string columnToAnalyze, int predictionParameter)
{
    switch (_predictionAlgorithm)
    {
        case AlgConstants.SSA:
            return PredictBySingularSpectrumAnalysis(daysToPredict, columnToAnalyze, predictionParameter);
        case AlgConstants.PLS:
            return PredictByPolynomialLeastSquares(daysToPredict, columnToAnalyze, predictionParameter);
        case AlgConstants.LLS:
            return PredictByPolynomialLeastSquares(daysToPredict, columnToAnalyze, 1);
        default:
            return null;
    }
}
```

Рисунок 2.11 – Метод PredictCoronaUnits

Створення моделі та її тренування на основі алгоритму сингулярного спектрального аналізу було реалізовано у методі PredictBySingularSpectrumAnalysis (Рисунок 2.12):

```
private CoronaTimeSeriesPrediction PredictBySingularSpectrumAnalysis(
    int daysToPredict, string columnToAnalyze, int windowSize)
{
    IEstimator<ITransformer> forecastEstimator = _mlContext.Forecasting.ForecastBySsa(
        outputColumnName: nameof(CoronaTimeSeriesPrediction.PredictedCoronaUnits),
        inputColumnName: columnToAnalyze,
        windowSize: windowSize,
        seriesLength: _dataToTrainLength,
        trainSize: _dataToTrainLength,
        horizon: daysToPredict,
        confidenceLowerBoundColumn: nameof(CoronaTimeSeriesPrediction.PredictedCoronaLowerBound),
        confidenceUpperBoundColumn: nameof(CoronaTimeSeriesPrediction.PredictedCoronaUpperBound));

    ITransformer forecastTransformer = forecastEstimator.Fit(_coronaDataView);

    var coronaPredictEngine = forecastTransformer
        .CreateTimeSeriesEngine<CoronaData, CoronaTimeSeriesPrediction>(_mlContext);

    var coronaPrediction = coronaPredictEngine.Predict();

    RemoveNegativeValues(coronaPrediction.PredictedCoronaUnits);

    return coronaPrediction;
}
```

Рисунок 2.12 – Метод PredictBySingularSpectrumAnalysis

Аналогічно створення та тренування моделі на основі алгоритму поліноміального методу найменших квадратів було реалізовано у методі PredictByPolynomialLeastSquares (Рисунок 2.13):

```
private CoronaTimeSeriesPrediction PredictByPolynomialLeastSquares(
    int daysToPredict, string columnToAnalyze, int polynomialDegree)
{
    var propToPredict = typeof(CoronaData).GetProperty(columnToAnalyze);

    var days = Enumerable.Range(0, CoronaDataForTraining.Length).Select(v => (double)v).ToArray();
    var values = CoronaDataForTraining.Select(el => (double)(float)propToPredict.GetValue(el)).ToArray();

    var polynomialLeastSquares = new PolynomialLeastSquares()
    {
        Degree = polynomialDegree
    };

    var polyRegressionTransformer = polynomialLeastSquares.Learn(days, values);

    var daysToPredictArr = Enumerable.Range(days.Length, daysToPredict)
        .Select(v => (double)v)
        .ToArray();

    var prediction = polyRegressionTransformer.Transform(daysToPredictArr);

    var predictionFloat = prediction.Select(v => (float)v).ToArray();
    RemoveNegativeValues(predictionFloat);
    return new CoronaTimeSeriesPrediction
    {
        PredictedCoronaUnits = predictionFloat
    };
}
```

Рисунок 2.13 – Метод PredictByPolynomialLeastSquares

Для прогнозу методом лінійного МНК достатньо викликати прогноз поліноміальним МНК з поліномом першого ступеню.

Важливо зазначити, що у випадку даних з COVID-19, динаміка яких може значно змінюватися з часом, варто окремо тренувати модель під кожен прогноз, а не зберігати і використовувати вже натреновану модель.

Крім цього потрібно розуміти, що алгоритми самі по собі не враховують природи даних і тому можуть прогнозувати і від’ємні значення. Отже для випадку прогнозування даних пандемії має сенс видалити від’ємні значення з прогнозу, якщо такі будуть.

Нарешті треба враховувати, що на практиці алгоритми як правило працюють з дробовими числами і тому перед прогнозом цілі значення кількості хворих, померлих і одужавших слід перевести у дробові.

2.6 Реалізація візуальної частини застосунку

Після реалізації усіх функціональних вимог до кінцевого застосунку, було розроблено візуальну частину для відображення даних. Для розробки візуальної частини застосунку було обрано WPF технології [30].

Для того, щоб оновити статистичні дані з платформи Kaggle або завантажити свої статистичні дані у форматі .csv було додано дві кнопки на верхню панель (зображено на рисунку 2.14) [1].

Рисунок 2.14 – UI оновлення та завантаження статистичних даних

При оновленні було додано додаткове модальне вікно для запобігання випадкового натискання (зображено на рисунку 2.15) [1]:

Рисунок 2.15 – Модальне вікно про оновлення статистичних даних

На вкладці Coronavirus Map Chart користувач може побачити відображення на карті світу даних щодо кількості хворих, одужавших та померлих за останній день з набору даних по кожній країні. Так, наприклад, на рисунку 2.16 користувач вибрав фільтр за померлими, та навів мишкою на карті на Україну [1]:

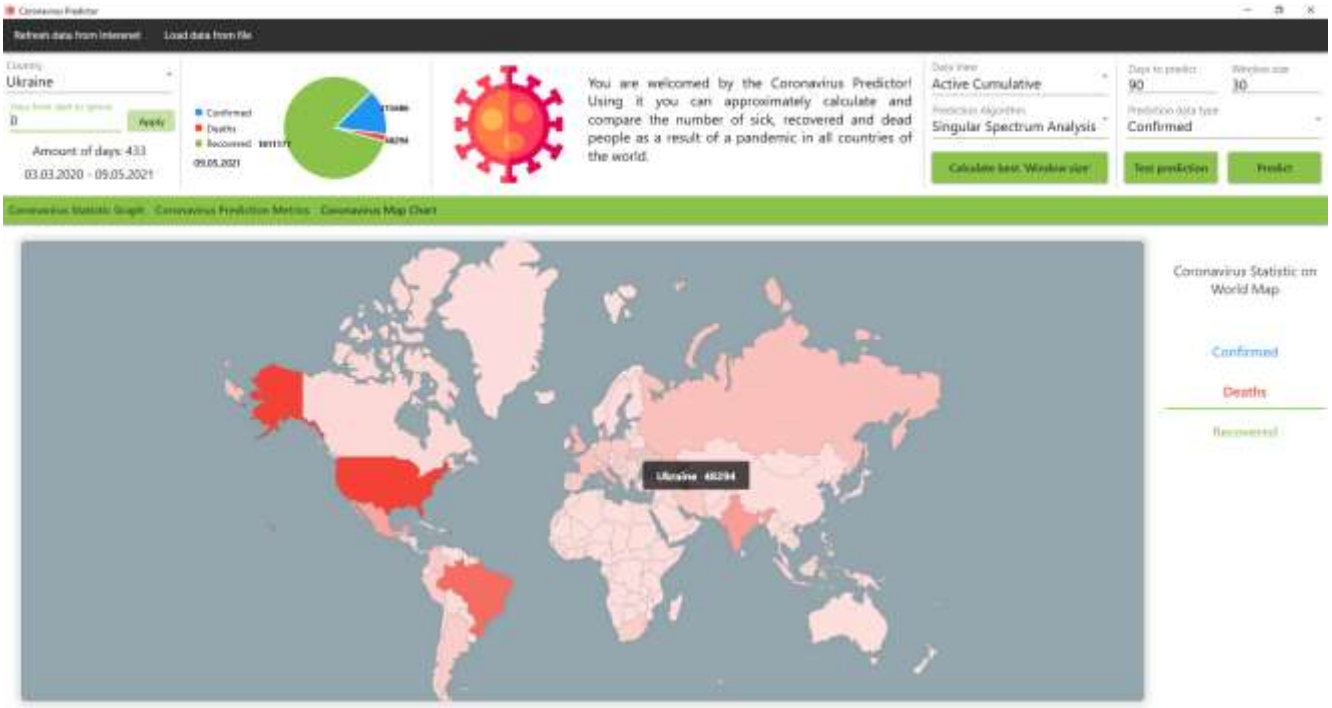

Рисунок 2.16 – Вкладка Coronavirus Map Chart

У верхньому лівому куту користувач може обрати країну з випадаючого списку, у якому відображаються усі країни з завантаженого або автоматично підгруженого набору даних. Після вибору країни вкладка Coronavirus Statistic Graph автоматично оновлюється: на ній відображаються дані щодо вибраної країни. Також оновлюється кругова діаграма. Аналогічно роботі карти, користувач може фільтрувати дані (вибирати графіки яких типів даних будуть показані), крім того, якщо користувач наведе мишкою на точку – відобразяться дані за вказаний день. Вищеописаний сценарій показаний на Рисунку 2.17 [1]:

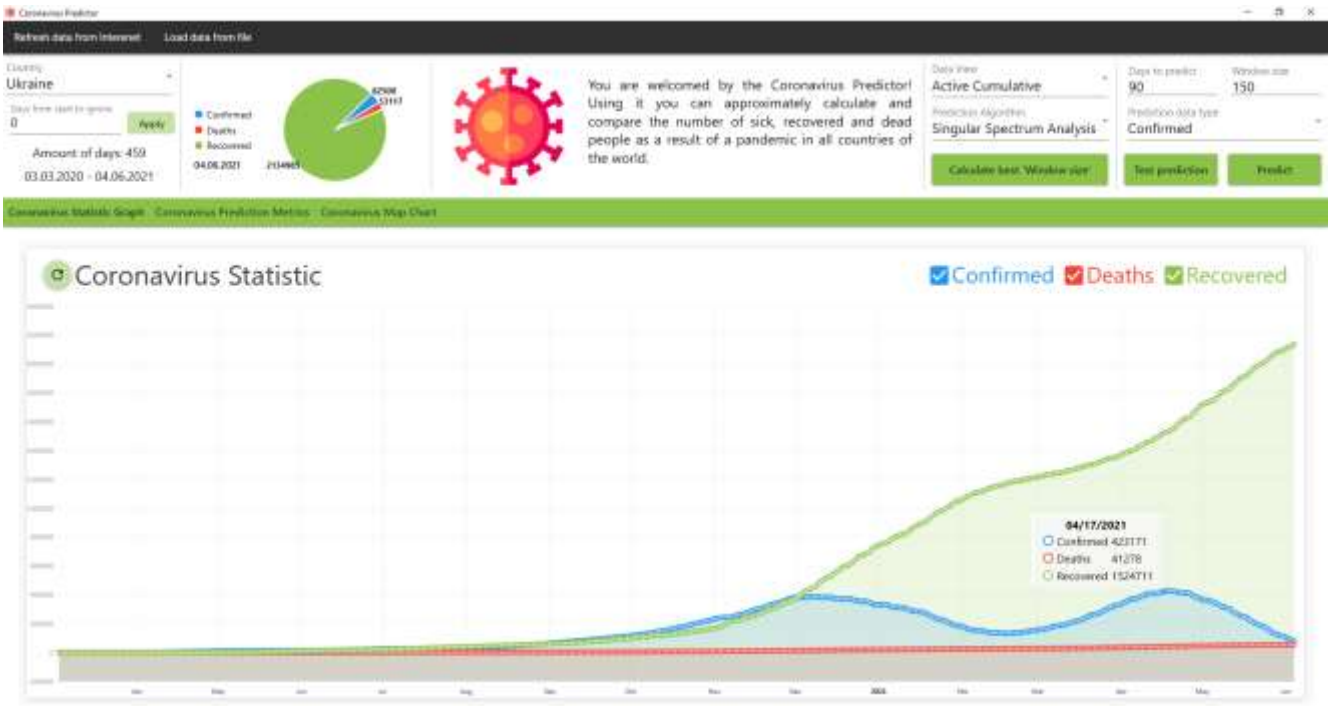

Рисунок 2.17 – Вкладка Coronavirus Statistic Graph

Мабуть, найважливіші функції – прогнозування та тестування отриманих результатів мають зручний та зрозумілий інтерфейс на панелі над графіком. Для обраної країни користувач має змогу ввести шість параметрів:

1) Тип представлення даних («Нові випадків кожного дня», «Накопичувальні данні» або «Накопичувальні данні з кількістю поточних хворих»).
2) Алгоритм для прогнозування («Сингулярний спектральний аналіз», «Поліноміальний метод найменших квадратів» або «Лінійний метод найменших квадратів»).
3) Тип даних для якого робити прогноз (хворі, померлі або одужавші).
4) Кількість днів від початку пандемії яку треба видалити з графіку і не враховувати при прогнозуванні (в інтерфейсі «Days from start to ignore»). Це інколи може бути корисним для підвищення точності прогнозу.
5) Кількість днів на яку він хоче отримати прогноз.
6) Параметр для алгоритму прогнозування, якщо алгоритм його вимагає (розмір вікна для алгоритму SSA або ступінь поліному для алгоритму поліноміального МНК).

Натиснувши кнопку Predict, користувачу відображаються точки фіолетового кольору – це нові спрогнозовані дані. Результат функції прогнозування зображено на рисунку 2.18 [1]:

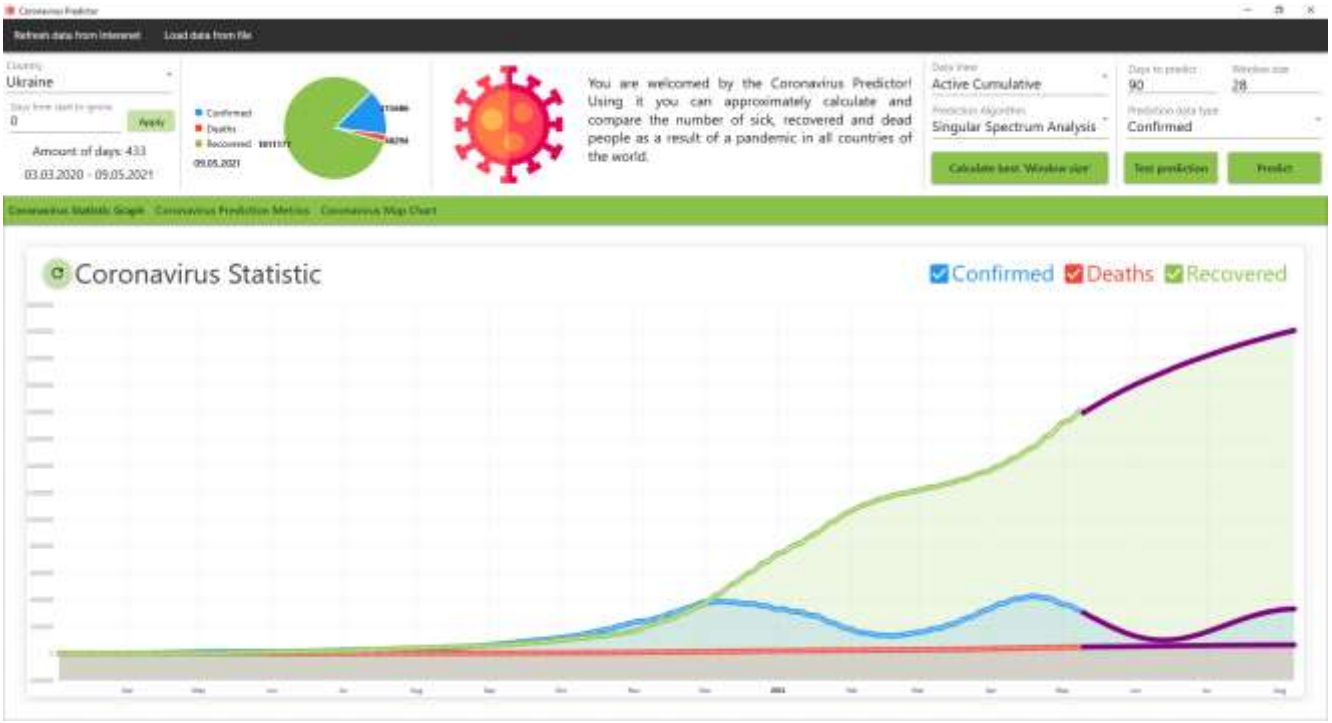

Рисунок 2.18 – Відображення результату прогнозування на графіку

Якщо користувач натискає Test Prediction – модель тренується на даних за винятком останніх N днів, де N – число вказане користувачем у полі Days to predict. У якості результату роботи на вказаній частині графіку для тестування прогнозування відображаються реальні дані (Рисунок 2.19) [1]:

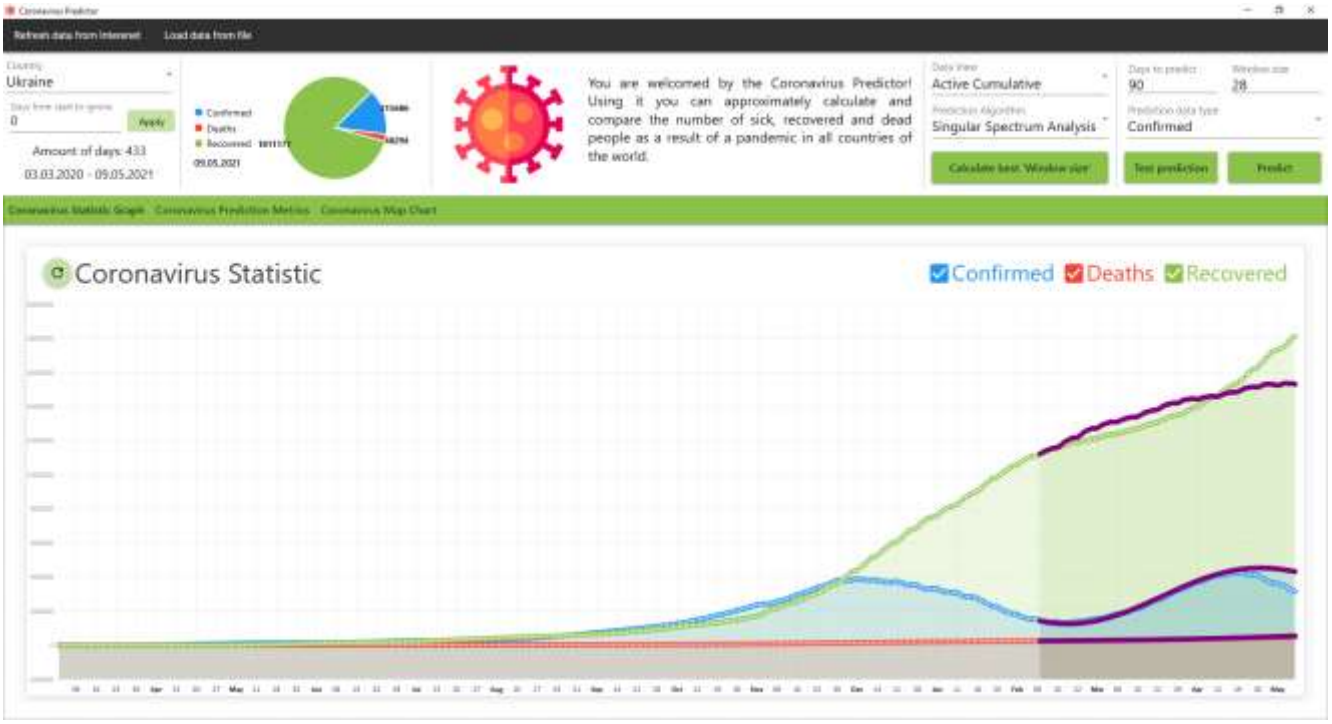

Рисунок 2.19 – Відображення результату тестування прогнозування на графіку

Усі прораховані метрики прогнозування, разом із обраними параметрами, відображаються на вкладці Coronavirus Prediction Metrics. У третьому розділі детально описано, як саме обраховуються дані метрики з математичної точки зору, а також опис реалізації. На даній вкладці відображаються метрики для окремо прорахованих категорій прогнозування: кількості померлих, одужавших та хворих (зображено на рисунку 2.20) [1].

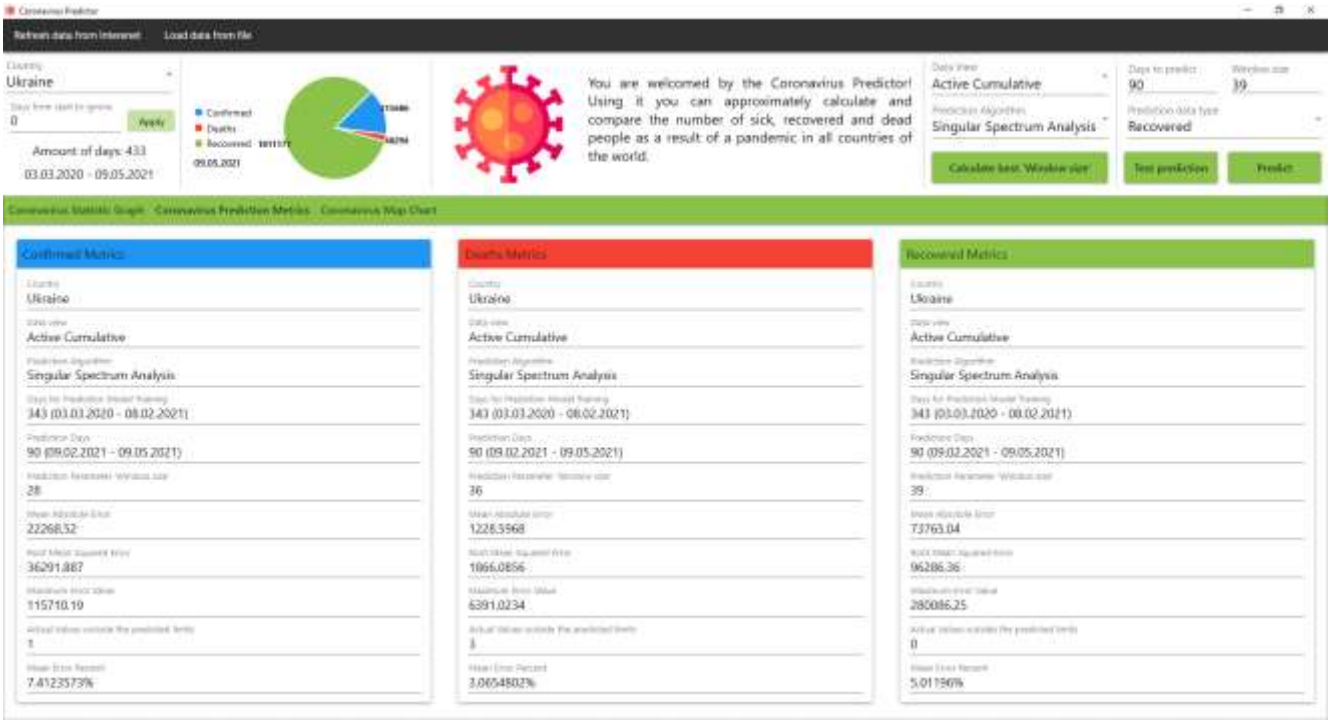

Рисунок 2.20 – Вкладка Coronavirus Prediction Metrics

РОЗДІЛ 3. АНАЛІЗ ОТРИМАНИХ РЕЗУЛЬТАТІВ

3.1 Основні характеристики прогнозування

При роботі над прогнозуванням часових рядів зазвичай проводиться розробка кількох можливих статистичних моделей досліджуваного процесу, а вже з них вибирається найбільш обґрунтована і відповідна ситуації модель. Яким чином можна оцінити розроблену модель? Існують спеціальні метрики для оцінки прогнозування. Найбільш популярними та часто використовуваними серед них є помилка прогнозування, середня процентна помилка прогнозування. Розглянемо ті з них, завдяки яким після розробки власної моделі ми зможемо оцінити отримані прогнози [1].

Помилка прогнозування – це величина, яка показує, як сильно прогнозоване значення відхилилося від фактичного. Вона використовується для розрахунку точності прогнозування, що у свою чергу допомагає нам оцінювати як точно і коректно ми сформували прогноз. Розрахувати помилку прогнозування можна за формулою 3.1 [1]:

$$e_y = y_j - \hat{y}_j, \qquad (3.1)$$

де y_j – фактичне значення змінної для j-го виміру, а $\hat{y}_j$ – прогнозне значення досліджуваної змінної.

Важливою характеристикою оцінки прогнозування є середня відсоткова помилка прогнозування. Позначимо дану величину *MAE* як скорочення від англ. the mean absolute error. Розрахувати її значення можна за наступною формулою (наведено у формулі 3.2) [1]:

$$\mathrm{M}AE = \frac{\sum_{j=1}^{N} |y_j - \hat{y}_j|}{N}, \qquad (3.2)$$

де y_j – фактичне значення змінної для j-го виміру, а $\hat{y}_j$ – прогнозне значення досліджуваної змінної, N – розмір певної вибірки вимірювань.

Точність прогнозування є поняттям прямо оберненим до помилки прогнозування. Якщо помилка прогнозування велика, то точність мала і навпаки, якщо помилка прогнозування мала, то точність велика. Говорячи про високу

точність, ми завжди говоримо про низьку помилки прогнозу і в цій області непорозуміння бути не повинно [31]. Важко знайти матеріали про прогнозування, у яких наведені оцінки саме точності прогнозу, хоча з точки зору людського сприйняття коректніше говорити саме про високу точність. Мабуть, тому у рекламних джерелах інформації завжди буде сказано про високу точність [1].

Ще однією загальновживаною метрикою є середнє квадратичне відхилення (з англ. the root mean square error, далі RMSE). Дана величина обраховується за формулою 3.3 [1]:

$$RMSE = \sqrt{\frac{\sum_{j=1}^{N}(y_j - \hat{y}_j)^2}{N}}, \qquad (3.3)$$

де y_j – фактичне значення змінної для j-го виміру, а $\hat{y}_j$ – прогнозоване значення досліджуваної змінної, N – розмір певної вибірки вимірювань.

Іншим поширеним показником середніх помилок прогнозування є процентна середня абсолютна помилка (з англ. the mean absolute percentage error). Оскільки даний показник ми будемо також обраховувати для оцінки реалізації власного прогнозування, тому нижче наведемо формулу її обчислення (формула 3.4) [1]:

$$\mathrm{M}APE = \frac{1}{N}\sum_{j=1}^{N}\frac{|y_j - \hat{y}_j|}{y_j} * 100\%, \qquad (3.4)$$

де y_j – фактичне значення змінної для j-го виміру, а $\hat{y}_j$ – прогнозне значення досліджуваної змінної, N – розмір певної вибірки вимірювань.

Також, у розрізі даної роботи було введене поняття кількість значень поза прогнозованими межами (з англ. the actual values outside the predicted limits). Воно означає кількість днів, для яких фактичне значення не знаходиться у спрогнозованих границях. Серед обраних алгоритмів прогнозування дана метрика актуальна тільки для сингулярного спектрального аналізу, адже його алгоритм окрім прогнозу повертає також мінімальну та максимальну межі «впевненості», за які реальні дані не повинні відхилятися з обраною ймовірністю, у даній роботі було використано значення за замовченням яке дорівнює 95%.

3.1.1 Розрахунок основних метрик отриманого прогнозу

Для оцінки прогнозу був створений окремий клас PredictionMetrics, що містить у собі змінні та метод CalculateMetrics для обрахування наступних метрик (детально описані у попередньому пункті даного розділу):

- середня відсоткова помилка прогнозування (MAE);
- середнє квадратичне відхилення (RMSE);
- кількість значень поза прогнозованими межами (AVOP);
- максимальна помилка;
- процентна середня абсолютна помилка (MAPE).

На рисунку 3.1 та 3.2 наведено методи CalculateMetrics та GetMeanErrorPercent, що належать класу PredictionMetrics та використовуються для обчислення метрик оцінки прогнозування [1]:

```
public static PredictionMetrics CalculateMetrics(float[] actualValues, CoronaTimeSeriesPrediction prediction,
    string country, int predictionParameter, DateTime firstActualValueDate, string predictionAlgorithm,
    CoronaData[] coronaDataForTraining, string dataView)
{
    var metrics = new PredictionMetrics
    {
        Country = country,
        DataView = dataView,
        PredictionAlgorithm = predictionAlgorithm,
        DaysForPredictionModelTraining = FormatDaysWithRange(coronaDataForTraining.Length,
            coronaDataForTraining.First().CoronaDate, coronaDataForTraining.Last().CoronaDate),
        PredictionDays = FormatDaysWithRange(prediction.PredictedCoronaUnits.Length, firstActualValueDate,
            firstActualValueDate.AddDays(actualValues.Length - 1)),
        PredictionParameter = predictionParameter
    };

    var errorForMetrics = actualValues.Zip(prediction.PredictedCoronaUnits,
        (actualValue, forecastValue) => Math.Abs(actualValue - forecastValue)).ToArray();

    metrics.MAE = errorForMetrics.Average();
    metrics.RMSE = (float)Math.Sqrt(errorForMetrics.Average(error => error * error));
    metrics.MaxError = errorForMetrics.Max();
    metrics.MeanErrorPercent = GetMeanErrorPercent(actualValues, prediction.PredictedCoronaUnits);

    if (prediction.PredictedCoronaLowerBound != null && prediction.PredictedCoronaUpperBound != null)
    {
        for (int i = 0; i < actualValues.Length; i++)
        {
            if (actualValues[i] < prediction.PredictedCoronaLowerBound[i] ||
                actualValues[i] > prediction.PredictedCoronaUpperBound[i])
            {
                metrics.ActualValuesOutsideBounds++;
            }
        }
    }
    else
    {
        metrics.ActualValuesOutsideBounds = -1;
    }

    return metrics;
}
```

Рисунок 3.1 – Метод CalculateMetrics

```
public static float GetMeanErrorPercent(float[] actualValues, float[] predictedCoronaUnits)
{
    return actualValues.Zip(predictedCoronaUnits,
        (actualValue, forecastValue) => Math.Abs((actualValue - forecastValue) * 100 / actualValue)).Average();
}
```

Рисунок 3.2 – Метод GetMeanErrorPercent

3.2 Практичні поради при прогнозуванні

Результати прогнозування алгоритмів, які вимагають додаткові параметри можуть дуже суттєво залежати від їх вибору. Розглянемо детальніше вибір параметрів обраних алгоритмів прогнозування. Серед обраних мною алгоритмів це параметр розміру вікна для алгоритму SSA та ступінь полінома для алгоритму поліноміального методу найменших квадратів.

Головною ознакою алгоритму SSA є наявність параметру розміру вікна. Умови відокремлюваності даних дають рекомендації щодо вибору розміру вікна L: він повинен бути достатньо великим ($L \sim N / 2$, де N – розмір певної вибірки вимірювань). А якщо ми хочемо отримати періодичну складову з відомим періодом, то розміри вікон, які поділяються на період націло, забезпечують кращу відокремлюваність. Алгоритм SSA з малим L частіше забезпечує більше згладжування спрогнозованих даних. Одне з рекомендованих значень розміру вікна – N / 3. Як правило, вибір розміру вікна важливий, але результат стабільний і при невеликих змінах значення L [16]. Крім того як показує практика алгоритм досить стійкий к «нестабільним» даним на початку тренувального часового ряду, навіть якщо вони поводять себе зовсім по іншому порівняно з даними наприкінці тренувального ряду.

Для алгоритму поліноміального МНК єдиним параметром є розмір полінома, за допомогою якого буде робитися апроксимація даних. Основні рекомендації – для прогнозу великої кількості даних, графік яких має періодичність і змінюється досить швидко є сенс обирати малі розміри поліному (наприклад 2 чи 3). Для прогнозу відносно не великої кількості даних $\sim N / 100$, де N – розмір певної

вибірки вимірювань, кращі результати дає поліном розміром до 20 графік якого є схожим на графік функції що він прогнозує. Для нестандартних функцій краще по експериментувати. Також метод досить чутливий до нестабільних даних на початку тренувального часового ряду, отже обрізавши якусь кількість днів з початку пандемії ви, ймовірно, можете покращити прогноз.

У реалізації готового застосунку був написаний алгоритм знаходження значення параметру, при якому досягається найкращі результати прогнозування на задану кількість днів в минулому, без урахування їх при тренуванні. Код цього розрахунку приведений у додатку А [1].

Отже, до виконання прогнозування, при використання алгоритму з параметром, у користувача є змога натиснути кнопку Calculate best parameter. Тоді, базуючись на поточні параметри прогнозу введені користувачем, відкриється вікно з наступними обчисленими за допомогою власного алгоритму даними:

- значення розміру параметра при якому досягається найближчий до фактичних результатів прогноз;
- Процент помилки прогнозу при такому параметрі.

На рисунку 3.3 зображено приклад такого вікна для кількості хворих для України на 10 днів:

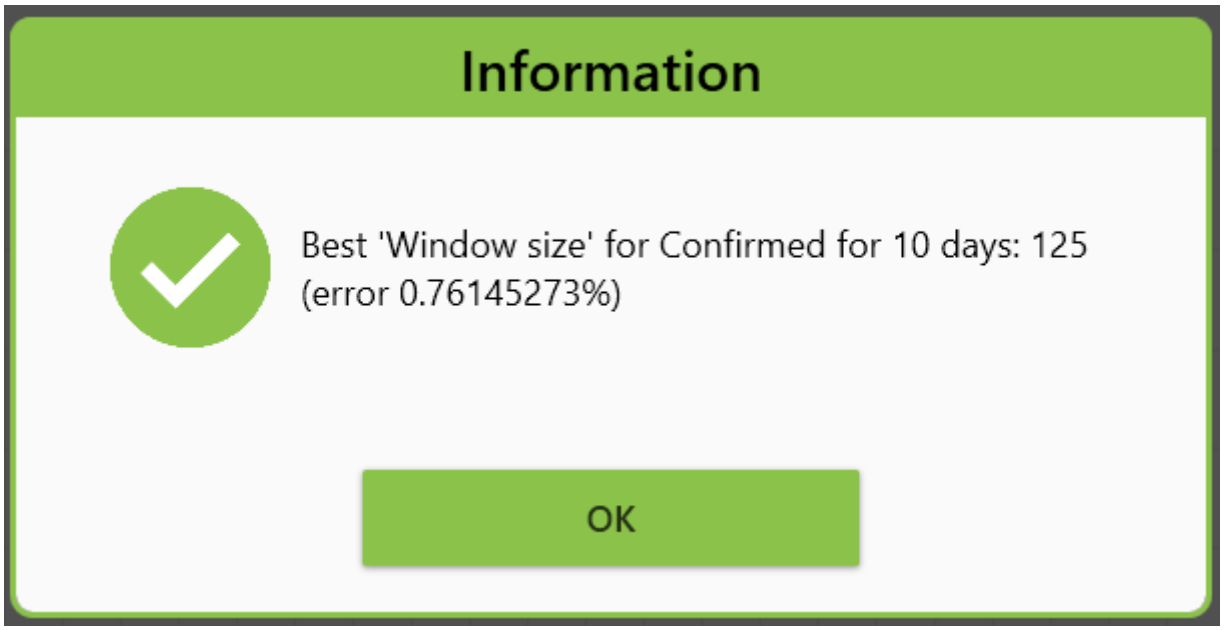

Рисунок 3.3 – Результат роботи алгоритму пошуку значення розміру вікна

Перед вибором параметру для подальшого прогнозу буде дуже корисно скористатися цією функцією, як правило знайдений параметр буде не дуже сильно відрізнятися від найкращого параметра для прогнозування днів в майбутньому та сильно зекономить вам час підбору оптимального параметру.

3.3 Порівняльний аналіз отриманих результатів

На даному етапі роботи було проведено експерименти для різних країн та часових проміжках використовуючи різні методи прогнозування. Для тестування були обрані наступні країни: Україна, Індія та Канада.

Україна була обрана оскільки публікації даної роботи у першу чергу будуть для українських наукових журналах, тому це буде ближче читачам/комісії. До того ж, доступ до джерел з різними статистичними даними є більш простіший.

Індія була обрана оскільки епідемічна ситуація у цій країні зараз хвилює чи не увесь світ. У період квітень-травень 2021 року Індія встановлює антирекорди не тільки за кількістю нових випадків захворювання, але й смертності. Дуже багато країн надсилають гуманітарну допомогу країні (США, Ізраїль, Україна також 30 травня 2021 року надіслала першу частину гуманітарної допомоги з кисневих концентраторів, оскільки там відчувається їх неймовірна нестача). До того ж, на фоні епідемічного положення економіка країни дуже погіршується, країни припиняють імпортну торгівлю з Індією. І, мабуть, найгірше, це спалахи багатьох інших страшних епідемій, і навіть був зафіксований новий штам коронавірусу (зараз його ідентифікують як індійський, його особливістю є подвійна мутація, що робить його набагато небезпечнішим).

Наступною країною для порівняння є Канада. Насправді, початкова робота та тестування застосунку проводились на США [1], але оскільки у певний момент США перестали надавати дані щодо кількості одужавших, прогнозування перестало бути релевантним, тому ми замінили США на Канаду.

Тестування вирішено було проводити за наступних представленнях даних:

1) «Накопичувальні данні з кількістю поточних хворих» – тобто дані за кожен день представляють собою поточну кількість активно хворих (тих хто захворів, але ще не одужав і не помер), поточну кількість одужавших та померлих за увесь час. Цей метод представлення добре показує стан пандемії. Графік даних виглядає досить лінійно.
2) «Нові випадки кожного дня» – тобто дані за кожен день представляють собою кількість виключно нових захворілих, одужавших та померлих (без накопичувальної складової). Цей метод представлення гарно показує динаміку розвитку пандемії. Графік даних виглядає досить періодично.

На таблицях 3.1 – 3.3 показані результати обчислення метрик оцінки прогнозування для України на 3 періоди: 7, 30 та 90 днів з представленням даних «Накопичувальні данні з кількістю поточних хворих»:

Таблиця 3.1 – Оцінка прогнозу для України на 7 днів (23.05.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	161	13	1	51	20	1	37	3	1
Середня помилка	1108	4087	65627	32	195	1463	1548	5713	8381
Середньоквадр. відхилення	1382	5068	66861	55	243	1484	1752	6845	9101
Максимальна помилка	2129	9827	81896	132	489	1856	2775	10802	12695
Кількість значень поза межами	0	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	0.76	2.5	45.3	0.06	0.37	2.8	0.27	1.25	0.4
Кількість днів для тренування моделі	446	446	45	446	446	45	446	446	45

Таблиця 3.2 – Оцінка прогнозу для України на 30 днів 30.04.2021 – 29.05.2021

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	43	3	1	32	4	1	59	2	1
Середня помилка	10363	64368	74402	102	2220	1956	4425	12850	11070
Середньоквадр. відхилення	14108	86852	87854	146	3235	2327	5360	15192	11999
Максимальна помилка	42402	176088	159363	394	7884	4374	10473	31222	19875
Кількість значень поза межами	0	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	5.6	35.6	39.1	0.2	4.3	3.8	1.9	0.6	0.5
Кількість днів для тренування моделі	423	72	192	423	72	12	222	192	12

Таблиця 3.3 – Оцінка прогнозу для України на 90 днів 01.03.2021 – 29.05.2021

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	20	7	1	13	2	1	23	3	1
Середня помилка	27302	192016	82756	1137	1774	5968	79608	161686	97291
Середньоквадр. відхилення	38735	247068	100806	2043	1898	7365	132880	203735	136018
Максимальна помилка	109534	427068	252446	6826	2642	11780	381234	523707	313876
Кількість значень поза межами	0	-	-	0	-	-	7	-	-
Процент помилки прогнозу (%)	12.71	60.2	38.3	2.3	4.7	13.2	4.2	9.7	5.5
Кількість днів для тренування моделі	262	192	192	262	192	192	363	192	45

На таблицях 3.4 – 3.6 показані результати обчислення метрик оцінки прогнозування для України на 3 періоди: 7, 30 та 90 днів з представленням даних «Нові випадки кожного дня»:

Таблиця 3.4 – Оцінка прогнозу для України на 7 днів (23.05.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	21	9	1	119	15	1	79	3	1
Середня помилка	454	563	988	13.9	58	76	1524	3675	9195
Середньоквадр. відхилення	580	727	1348	18.3	71	78	1902	4130	10650
Максимальна помилка	962	1549	2752	31.1	126	106	3648	6487	15393
Кількість значень поза межами	0	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	18.1	26.1	47.8	8.3	54.1	58.5	124	29.4	65.4
Кількість днів для тренування моделі	446	446	3	446	446	3	446	446	3

Таблиця 3.5 – Оцінка прогнозу для України на 30 днів (23.05.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	64	12	1	36	11	1	91	4	1
Середня помилка	1757	2339	2795	68	89	121	3124	5562	8375
Середньоквадр. відхилення	2266	2823	3424	85	99	145	4039	6855	9707
Максимальна помилка	5259	5563	4669	230	163	254	9411	11663	17663
Кількість значень поза межами	5	-	-	14	-	-	9	-	-
Процент помилки прогнозу (%)	49.9	50.6	70.0	38.2	44.2	83.2	23.6	36.4	87.4
Кількість днів для тренування моделі	423	423	22	423	423	22	423	423	22

Таблиця 3.6 – Оцінка прогнозу для України на 90 днів (01.03.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	69	2	1	32	32	1	37	3	1
Середня помилка	2591	4384	4607	99	99	109	4091	6918	4306
Середньоквадр. відхилення	3363	5222	5451	124	124	134	5606	8861	4913
Максимальна помилка	9378	11778	10497	260	260	281	13459	18124	9285
Кількість значень поза межами	26	-	-	48	-	-	31	-	-
Процент помилки прогнозу (%)	30.4	69.3	85.4	33.2	33.2	40.6	43.8	78.3	77.0
Кількість днів для тренування моделі	363	363	363	363	363	363	363	363	363

На таблицях 3.7 – 3.9 показані результати обчислення метрик оцінки прогнозування для Індії на 3 періоди: 7, 30 та 90 днів з представленням даних «Накопичувальні данні з кількістю поточних хворих»:

Таблиця 3.7 – Оцінка прогнозу для Індії на 7 днів (23.05.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	12	15	1	147	9	1	199	13	1
Середня помилка	19826	54108	271156	90	975	97767	17371	78696	105070
Середньоквадр. відхилення	25345	85017	326983	132	1239	97982	27625	89194	120409
Максимальна помилка	25345	203760	586662	305	2550	107247	67776	164966	205656
Кількість значень поза межами	0	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	0.8	2.4	10.7	0.02	0.3	31.0	0.06	0.3	0.4
Кількість днів для тренування моделі	479	278	205	479	278	205	479	278	28

Таблиця 3.8 – Оцінка прогнозу для Індії на 30 днів (30.04.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	72	4	1	192	8	1	99	25	1
Середня помилка	99617	963529	1375629	2809	10297	8887	148792	1310580	1108591
Середньоквадр. відхилення	133962	1106050	1851424	4273	11026	10677	319147	1720143	1310397
Максимальна помилка	271655	2340752	3801928	13711	14643	17772	1115560	5241498	2014232
Кількість значень поза межами	13	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	2.8	33.5	50.9	0.9	3.9	3.1	0.6	5.8	4.9
Кількість днів для тренування моделі	456	456	5	456	456	5	456	456	5

Таблиця 3.9 – Оцінка прогнозу для Індії на 90 днів (01.03.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	149	6	1	103	5	1	156	3	1
Середня помилка	81142	1056213	1311355	28761	13073	31410	3808689	4074813	2731994
Середньоквадр. відхилення	1172736	1450990	1717358	47929	19839	41424	5908738	6358678	4510226
Максимальна помилка	2306766	2700581	3117005	120938	46333	109478	14423772	15707388	11555893
Кількість значень поза межами	31	-	-	29	-	-	49	-	-
Процент помилки прогнозу (%)	35.0	49.8	86.6	10.8	5.2	13.9	19.7	21.1	13.7
Кількість днів для тренування моделі	396	396	396	396	396	396	396	396	396

На таблицях 3.10 – 3.12 показані результати обчислення метрик оцінки прогнозування для Індії на 3 періоди: 7, 30 та 90 днів з представленням даних «Нові випадки кожного дня»:

Таблиця 3.10 – Оцінка прогнозу для Індії на 7 днів (23.05.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	83	25	1	133	15	1	9	15	1
Середня помилка	3211	19662	8486	159	375	416	13002	20987	88624
Середньоквадр. відхилення	3387	23980	10793	247	417	462	16322	25204	91247
Максимальна помилка	5080	41753	23312	497	690	674	28271	44155	120277
Кількість значень поза межами	0	-	-	2	-	-	1	-	-
Процент помилки прогнозу (%)	1.6	10.4	4.3	4.0	9.5	11.3	4.5	7.3	31.2
Кількість днів для тренування моделі	479	479	13	479	479	13	479	479	13

Таблиця 3.11 – Оцінка прогнозу для Індії на 30 днів (30.04.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	142	3	1	2	4	1	190	4	1
Середня помилка	13636	110510	208374	349	862	1248	14879	94516	263975
Середньоквадр. відхилення	16344	131151	223272	421	945	1534	18931	101903	266600
Максимальна помилка	35682	218665	320088	906	1486	3196	54986	164649	350357
Кількість значень поза межами	5	-	-	9	-	-	0	-	-
Процент помилки прогнозу (%)	4.6	33.8	65.7	8.5	22.2	32.3	4.4	27.9	78.3
Кількість днів для тренування моделі	456	456	456	456	456	3	456	456	456

Таблиця 3.12 – Оцінка прогнозу для Індії на 90 днів (01.03.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	163	4	1	189	4	1	134	6	1
Середня помилка	93470	131753	146101	1429	1024	1544	115090	50932	128843
Середньоквадр. відхилення	131803	172871	190221	2018	1460	2057	164244	64509	174488
Максимальна помилка	277233	332809	361152	3810	2812	3867	341560	159970	364498
Кількість значень поза межами	38	-	-	0	-	-	50	-	-
Процент помилки прогнозу (%)	45.1	63.9	78.6	58.6	40.5	145.0	47.0	56.1	96.4
Кількість днів для тренування моделі	396	396	396	396	396	396	396	396	396

На таблицях 3.13 – 3.15 показані результати обчислення метрик оцінки прогнозування для Канади на 3 періоди: 7, 30 та 90 днів з представленням даних «Накопичувальні данні з кількістю поточних хворих»:

Таблиця 3.13 – Оцінка прогнозу для Канади на 7 днів (23.05.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	99	15	1	45	41	1	134	17	1
Середня помилка	1026	1060	4770	10	75	32	1547	4062	1660
Середньоквадр. відхилення	1112	1226	5553	12	83	35	2092	5101	1955
Максимальна помилка	1654	2149	8080	28	121	54	4892	8145	3013
Кількість значень поза межами	0	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	2.4	2.4	11.8	0.04	0.2	0.1	0.1	0.3	0.1
Кількість днів для тренування моделі	483	483	3	483	483	3	483	483	3

Таблиця 3.14 – Оцінка прогнозу для Канади на 30 днів (30.04.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	201	2	1	159	8	1	15	2	1
Середня помилка	6558	13085	10029	33	537	174	1934	23850	32791
Середньоквадр. відхилення	7405	15670	14740	38	760	202	2235	24683	39640
Максимальна помилка	12815	34313	33399	80	1816	364	4628	32079	77672
Кількість значень поза межами	0	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	10.5	24.5	21.1	0.1	2.1	0.6	0.1	1.9	2.5
Кількість днів для тренування моделі	460	460	2	460	460	2	460	460	2

Таблиця 3.15 – Оцінка прогнозу для Канади на 90 днів (01.03.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	47	2	1	35	2	1	159	2	1
Середня помилка	9129	19993	17138	227	838	2324	37583	19241	63659
Середньоквадр. відхилення	12845	24612	19143	269	1056	2345	54032	23357	92881
Максимальна помилка	37788	50452	33254	455	2098	2975	195508	42086	194698
Кількість значень поза межами	0	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	16.1	50.6	39.8	0.9	3.6	9.9	3.3	1.9	5.3
Кількість днів для тренування моделі	400	400	400	400	400	400	400	400	20

На таблицях 3.16 – 3.18 показані результати обчислення метрик оцінки прогнозування для Канади на 3 періоди: 7, 30 та 90 днів з представленням даних «Нові випадки кожного дня»:

Таблиця 3.16 – Оцінка прогнозу для Канади на 7 днів (23.05.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	67	17	1	49	5	1	2	12	1
Середня помилка	943	951	2175	10	13	25	1285	1363	4572
Середньоквадр. відхилення	1102	1138	2634	18	21	34	1674	1715	5148
Максимальна помилка	1789	2158	4721	48	51	79	3302	3301	8557
Кількість значень поза межами	0	-	-	1	-	-	0	-	-
Процент помилки прогнозу (%)	36.7	33.8	67.9	19.0	32.1	57.4	22.2	24.2	79.2
Кількість днів для тренування моделі	483	483	3	4483	483	3	483	483	3

Таблиця 3.17 – Оцінка прогнозу для Канади на 30 днів (30.04.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	208	4	1	47	3	1	171	3	1
Середня помилка	1306	1624	1975	14	17	30	1340	1643	1608
Середньоквадр. відхилення	1669	2017	2447	20	20	33	1694	2113	2047
Максимальна помилка	4006	4883	4881	58	40	51	4241	5714	5589
Кількість значень поза межами	0	-	-	0	-	-	0	-	-
Процент помилки прогнозу (%)	29.5	41.7	54.2	29.6	53.2	96.5	19.5	22.6	23.3
Кількість днів для тренування моделі	460	460	460	460	460	460	460	460	460

Таблиця 3.18 – Оцінка прогнозу для Канади на 90 днів (01.03.2021 – 29.05.2021)

	Хворі			Померлі			Одужавші		
	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК	SSA	ПМНК	ЛМНК
Параметр прогнозу	35	3	1	2	4	1	9	3	1
Середня помилка	1694	2397	2064	11	35	55	2353	2156	1896
Середньоквадр. відхилення	2227	2841	2356	15	39	56	3091	2540	2219
Максимальна помилка	6208	6846	4928	52	90	78	7875	2532	5266
Кількість значень поза межами	0	-	-	0	-	-	24	-	-
Процент помилки прогнозу (%)	27.1	42.0	46.9	33.8	92.4	179.4	35.7	59.6	47.3
Кількість днів для тренування моделі	400	400	400	400	400	400	400	400	400

Отже, можна зробити наступні висновки:

1) Сингулярний Спектральний Аналіз загалом прогнозує дані з COVID-19 краще за обидва методи найменших квадратів, незалежно від представлення даних.
2) Методи найменших квадратів суттєво краще прогнозують дані у представленні «Накопичувальні данні з кількістю поточних хворих», ніж у «Нові випадки кожного дня». Це пояснюється тим, що дані у першому представленні є більш лінійними і тому більш підходять для прогнозу за допомогою МНК.
3) Методи надають досить точні результати прогнозування – у середньому до 10% для 7-ми днів, до 40% для 30-ти, та до 100% для 90 днів. При цьому SSA загалом прогнозує у 2-3 рази краще за методи найменших квадратів.
4) Поліноміальний метод найменших квадратів краще прогнозує періодичні дані за його лінійний аналог, а лінійний МНК загалом краще прогнозує більш лінійні дані.

5) Для SSA та ПМНК досить високу роль відіграє добре підібраний параметр прогнозу. Для кожної категорії даних (кількість хворих, померлих, одужавших) та типу їх представлення варто підібрати окреме значення параметру.
6) Для покращення прогнозу ЛМНК досить часто вигідно тренувати його на невеликій кількості кінцевих даних.
7) Усі методи значно краще прогнозують на меншу кількість днів.

3.4 Прогноз для України на майбутнє

Як показав порівняльний аналіз тестування у розділі 3.2, найкращим алгоритмом для прогнозування даних з COVID-19 у більшості випадків виявився сингулярний спектральний аналіз. Тепер, за допомогою готового застосунку, спрогнозуємо дані на наступні 30 днів для України та зробимо відповідні висновки.

По кількості активно хворих прогноз показує (Рис. 3.4), що вона буде знижатися до кінця червня (точніше до 24.06.2021, з мінімальним значенням у 4165 активно хворіючих людей), а потім почне знову активно зростати. Отже, можна зробити висновок, що на початку липня слід підсилити карантин, якщо цього не зробити – ймовірна третя велика хвиля захворювань.

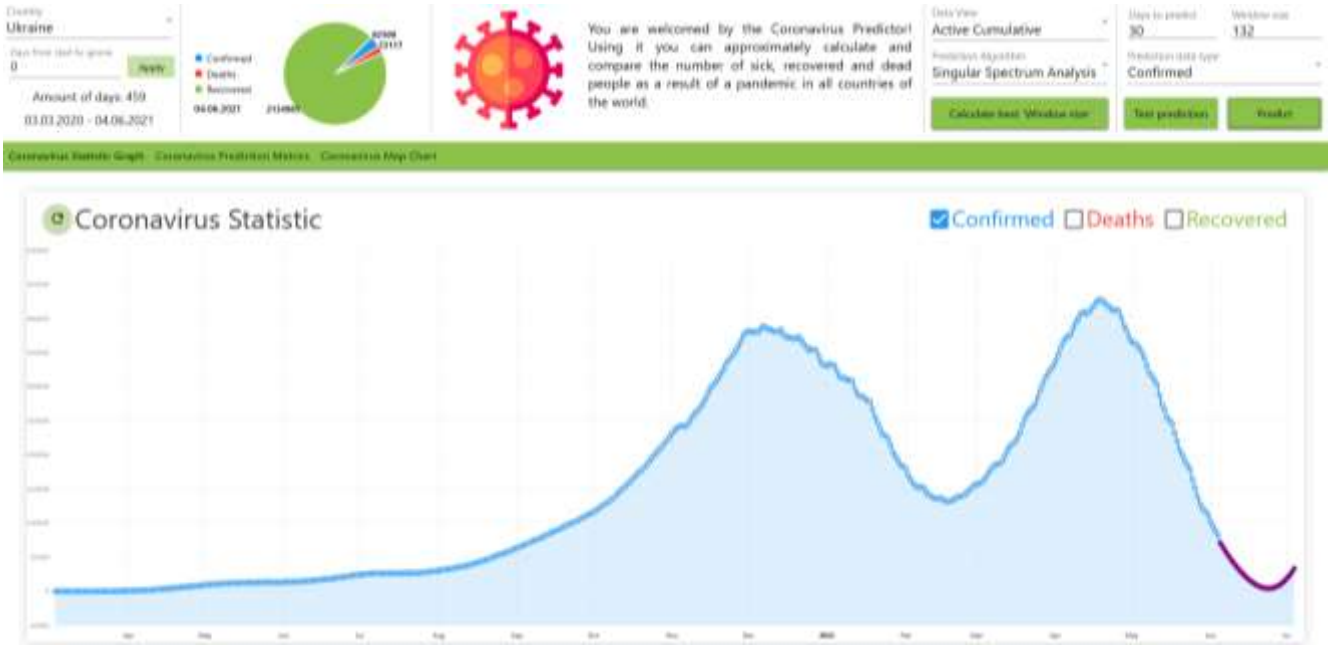

Рисунок 3.4 – Прогноз активно хворих для України на наступні 30 днів

Як видно на рисунку 3.5, помирати та одужувати люди будуть з приблизно з такою ж інтенсивністю, як і раніше (трохи меншою через меншу кількість поточних хворих у цей період), причому одужувати набагато більше, отже ніяких дуже категоричних заходів можна не вживати.

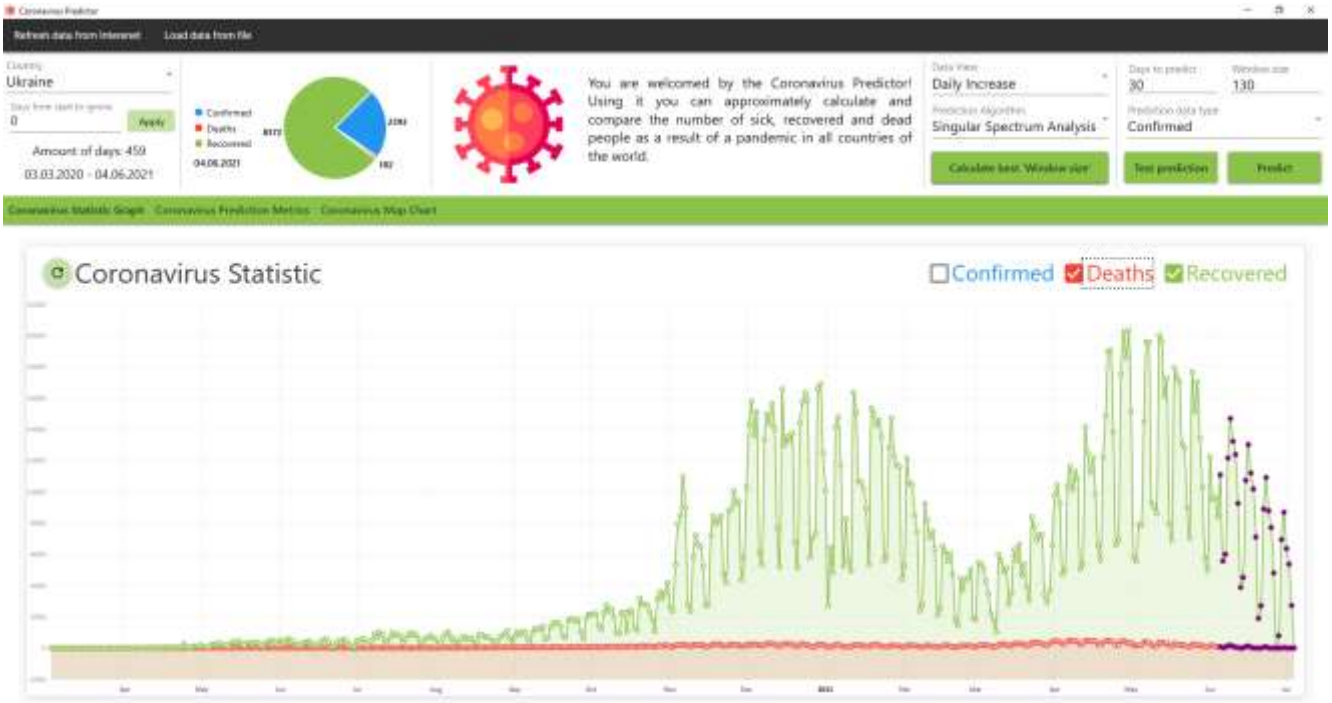

Рисунок 3.5 – Прогноз щоденного приросту одужавших та померлих

Також у застосунку можна побачити, що на даний момент одужало вже набагато більше людей ніж померло або активно хворіє (Рис. 3.6), отже повна загибель українцям не загрожує.

Рисунок 3.6 – Кругова діаграма даних з COVID-19 для України за 04.06.2021

ВИСНОВКИ

Пандемія коронавіруса COVID-19, яка викликала найсильніший епідеміологічний шок і застала зненацька усі без винятку країни світу, увійде в історію як одна з найважливіших подій першої чверті XXI століття. Криза виявила непідготовленість міжнародної спільноти до подібного роду випробувань, як на міжнародному рівні, так і на рівні держав та місцевих урядів.

Протягом останніх півтора року перед кожною країною світу стоїть задача не тільки забезпечити себе актуальними даними, а й навчитись оперувати цими даними, і навіть прогнозувати дані на майбутнє задля швидкого та більш ефективного реагування.

І хоча багато країн вже нормалізували ситуацію з епідеміологічними станом, навіть там час від часу все ще стаються спалахи захворювання (як, наприклад, у китайському мегаполісі Гуанчжоу, який закрили 30 травня на карантин на фоні того, що у всій країні довгий час були зафіксовані лише поодинокі випадки захворювання), вже не говорячи про Індію, де зараз ситуація дуже критична.

Отже питання оцінки епідеміологічної ситуації та на її основі прийняття рішень є все ще актуальними. Такі питання не вирішуються без прогнозування. Загалом, прогнозування тісно пов'язане з плануванням і використовується для ефективного прийняття рішень.

Кількість хворих, одужавших та померлих людей кожного дня з початку епідемії з математичної точки зору можна розглянути як часовий ряд. Тому, доцільним та актуальним є використання аналізу часового ряду для прогнозування епідемічної ситуації викликаної захворюванням, у наших реаліях саме COVID-19 у різних країнах, у тому числі і України. Так, у роботі розглядаються 3 алгоритми для прогнозування, на основі яких було розроблено прогнозування епідемічної ситуації:

1) сингулярний спектральний аналіз;
2) поліноміальний метод найменших квадратів;
3) лінійний метод найменших квадратів.

Загалом, у даній роботі було:

- ✓ опрацьовано літературу, присвячену часовим рядам, та методам їх аналізу;
- ✓ знайдено релевантні дані щодо кількості хворих на коронавірусну інфекцію, померлих і одужавших для подальшої роботи з ними;
- ✓ визначено вимоги до застосунку;
- ✓ розроблено модель для прогнозування епідемічної ситуації на основі сингулярного спектрального аналізу;
- ✓ розроблено модель для прогнозування епідемічної ситуації на основі поліноміального методу найменших квадратів;
- ✓ розроблено модель для прогнозування епідемічної ситуації на основі лінійного методу найменших квадратів;
- ✓ розроблено функціональну та візуальну частини застосунку згідно до визначених вимог на мові програмування C#, технології WPF, з використанням фреймворків ML.NET та Accord.NET;
- ✓ протестовано готовий застосунок на різних країнах, часових проміжках, представлень даних, моделях прогнозування та зроблені висновки;
- ✓ за результатами порівняльного аналізу було зроблено прогнозування для України на наступні 30 днів та надані поради для відповідного реагування.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. *Киян, М.* Прогнозування епідемічної ситуації викликаної захворюванням COVID-19 за допомогою методу аналізу часових рядів. Національний університет «Києво-Могилянська Академія», 2020. 33 с.
2. Семенова І. Лабораторна робота. Чому світ повернувся до дискусії про можливу «втечу» коронавірусу з Уханьського інституту [Електронний ресурс] – Режим доступу до ресурсу: https://nv.ua/ukr/world/countries/koronavirus-2021-vcheni-viznayut-shcho-virus-mig-vtekti-z-laboratoriji-ostanni-novini-50163098.html
3. *Hamilton, J.* Time Series Analysis. Princeton University Press. 1994. 820 p.
4. *Brockwell P., Davis R.* Introduction to Time Series and Forecasting (Springer Texts in Statistics). Springer, 2016. 425 p.
5. *Golyandina, N., Korobeynikov, A., Zhigljavsky, A.* Basic Singular Spectrum Analysis and Forecasting with R. Springer; 1st ed, 2018. 288 p.
6. ML.NET Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/dotnet/api/microsoft.ml.timeseriescatalog?view=ml-dotnet
7. McCaffrey J. Time-Series Regression Using a C# Neural Network [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/archive/msdn-magazine/2017/october/test-run-time-series-regression-using-a-csharp-neural-network
8. ML.NET Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/en-us/dotnet/api/microsoft.ml.regressioncatalog.regressiontrainers?view=ml-dotnet
9. Метод наименьших квадратов [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BD%D0%B0%D0%B8%D0%BC%D0%B5%D0%BD%D1%8C%D1%88%

D0%B8%D1%85_%D0%BA%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82%D0%BE%D0%B2

10. NIST/SEMATECH e-Handbook of Statistical Methods [Електронний ресурс] – Режим доступу до ресурсу: https://www.itl.nist.gov/div898/handbook/pmd/section1/pmd141.htm
11. Polynomial regression [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Polynomial_regression
12. Simple linear regression [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Simple_linear_regression
13. SSA [Електронний ресурс] – Режим доступу до ресурсу: https://ru.wikipedia.org/wiki/SSA_(%D0%BC%D0%B5%D1%82%D0%BE%D0%B4)
14. Голяндіна, Н. Метод "Гусеница"-SAA: анализ временных рядов: Учебное пособие. С.-Петербурзький державний університет, 2004. 76 с.
15. Golyandina, N., Nekrutkin, V., Zhigljavsky, A., 2001. Analysis of Time Series Structure: SSA and Related Techniques. Chapman&Hall/CRC.
16. *Golyandina, N.* On the choice of parameters in singular spectrum analysis and related subspace-based methods. Stat. Interface 3, 2010. 259-279 pp.
17. Golyandina, N., Zhigljavsky, A., 2013. Singular Spectrum Analysis for time series. Springer Briefs in Statistics. Springer.
18. *Capellman, J.* Hands-On Machine Learning with ML.NET: Getting started with Microsoft ML.NET to implement popular machine learning algorithms in C#. Packt Publishing, 2020. 296 p.
19. *Price, M.* C# 8.0 and .NET Core 3.0 – Modern Cross-Platform Development: Build applications with C#, .NET Core, Entity Framework Core, ASP.NET Core, and ML.NET using Visual Studio Code, 4th Edition. Packt Publishing, 2019. 818 p.
20. Accord.NET [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Accord.NET

21. Accord.NET Framework [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/accord-net/framework
22. Вступительное слово Генеральногодиректора ВОЗ на пресс-брифинге поCOVID-19 28 декабря2020 г. [Електронний ресурс] – Режим доступу до ресурсу: https://www.who.int/ru/director-general/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---28-december-2020
23. COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/CSSEGISandData/COVID-19
24. Coronavirus Pandemic (COVID-19) – the data [Електронний ресурс] – Режим доступу до ресурсу: https://ourworldindata.org/coronavirus-data
25. COVID-19 Projections [Електронний ресурс] – Режим доступу до ресурсу: https://covid19.healthdata.org/global?view=cumulative-deaths&tab=trend
26. COVID-19 Projection Dashboard [Електронний ресурс] – Режим доступу до ресурсу: https://iem-modeling.com/
27. Bian X. Coronavirus (COVID-19) Visualization & Prediction [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/therealcyberlord/coronavirus-covid-19-visualization-prediction
28. Devakumar K. COVID-19 Dataset [Електронний ресурс] – Режим доступу до ресурсу: https://www.kaggle.com/imdevskp/corona-virus-report#covid_19_clean_complete.csv
29. *Griffiths I.* Programming C# 8.0: Build Cloud, Web, and Desktop Applications. O'Reilly Media, 2019. 802 p.
30. *Nathan A.* WPF 4.5 Unleashed. Sams Publishing, 2013. 864 p.
31. *Nielsen, A.* Practical Time Series Analysis: Prediction with Statistics and Machine Learning. O'Reilly Media, 2019. 504 p.

ДОДАТОК А

```
        public static (int bestPredictionParameter, float lowestMeanErrorPercent)
CalculateBestPredictionParameter(
            CoronaData[] coronaDataArr, PredictionDataType predictionDataType, int
daysToTestPrediction, string predictionAlgorithm)
        {
            if (daysToTestPrediction < 1)
            {
                throw new Exception("Days to test prediction must be positive value");
            }
            var predictionParameterLimit = (coronaDataArr.Length - daysToTestPrediction) / 2;
            if (predictionParameterLimit < 3)
            {
                throw new Exception($"Can not calculate best
{AlgToParamConverter.GetAlgParamNameFromAlg(predictionAlgorithm, true)} for
{daysToTestPrediction} days to test prediction and {coronaDataArr.Length} amount of days");
            }

            var timeSeriesAnalyzer = new TimeSeriesAnalyzer(coronaDataArr,
predictionAlgorithm, daysToTestPrediction);
            int bestPredictionParameter = -1;
            float lowestMeanErrorPercent = float.MaxValue;

            var propertyToPredictName = $"Corona{predictionDataType}";
            var propertyToPredict = typeof(CoronaData).GetProperty(propertyToPredictName);
            var actualCoronaValues = coronaDataArr
                .TakeLast(daysToTestPrediction)
                .Select(c => (float)propertyToPredict.GetValue(c))
                .ToArray();

            for (int predictionParameter = 2; predictionParameter < predictionParameterLimit;
predictionParameter++)
            {
                var prediction = timeSeriesAnalyzer.PredictCoronaUnits(daysToTestPrediction,
propertyToPredictName, predictionParameter);
                var meanErrorPercent =
PredictionMetrics.GetMeanErrorPercent(actualCoronaValues, prediction.PredictedCoronaUnits);
                if (meanErrorPercent < lowestMeanErrorPercent)
                {
                    lowestMeanErrorPercent = meanErrorPercent;
                    bestPredictionParameter = predictionParameter;
                }
            }

            return (bestPredictionParameter, lowestMeanErrorPercent);
        }
```