

Міністерство освіти і науки, молоді та спорту України  
Дніпропетровський національний університет імені Олеся Гончара  
Факультет прикладної математики  
Кафедра обчислювальної математики та математичної кібернетики

## ЗВІТ

### з переддипломної практики

**Виконавець:** студент IV курсу, групи ПМ-13-1  
спеціальності 6.040301 -  
«прикладна математика»  
**Кривонос Олександр Дмитрович**

**Керівник:** д.ф.-м.н., проф., проф.  
**Кузьменко Василь Іванович**

Кількість балів \_\_\_\_\_  
Національна шкала \_\_\_\_\_  
Оцінка ECTS \_\_\_\_\_

Члени комісії: \_\_\_\_\_  
(підпис) (прізвище та ініціали)

\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_

(підпис) (прізвище та ініціали)

Дніпро  
2017

## ЗМІСТ

<b>1. ВСТУП .....</b>	<b>3</b>
<b>2. ПОСТАНОВКА ЗАДАЧІ .....</b>	<b>5</b>
<b>3. ПРОГНОЗУВАННЯ ЧАСОВИХ РЯДІВ .....</b>	<b>6</b>
<b>3.1. Огляд методів прогнозування часових рядів .....</b>	<b>6</b>
<b>3.1.1. Метод ковзного середнього ( moving average, MA). ....</b>	<b>6</b>
<b>3.1.2. Модель Бокса-Дженкінса (ARIMA) .....</b>	<b>7</b>
<b>3.1.3. Метод аналізу сингулярного спектру (SSA).....</b>	<b>8</b>
<b>3.1.4. Локальна апроксимація (LA).....</b>	<b>9</b>
<b>3.1.5. Нейромережеве прогнозування часових рядів .....</b>	<b>12</b>
<b>4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ПРОГНОЗУВАННЯ ЧАСОВИХ РЯДІВ.....</b>	<b>16</b>
<b>4.1. Функціональні можливості та структура програми .....</b>	<b>16</b>
<b>4.2. Інструкція користувача .....</b>	<b>19</b>
<b>5. РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНИХ ЕКСПЕРИМЕНТІВ.....</b>	<b>26</b>
<b>6. ВИСНОВКИ.....</b>	<b>32</b>
<b>7. СПИСОК ЛІТЕРАТУРИ .....</b>	<b>33</b>

## 1. ВСТУП

Для вивчення властивостей складних систем широко використовується підхід, заснований на аналізі сигналів, вироблених системою. Це дуже актуально в тих випадках, коли математично описати досліджуваний процес неможливо, але в нашому розпорядженні може бути деяка характерна спостережувана величина. Тому аналіз систем, особливо при експериментальних дослідженнях, часто реалізується за допомогою оброблення реєстрованих сигналів. Наприклад, в медицині — кардіограми, в сейсмології — коливання земної кори, в метеорології — дані метеоспостережень.

*Часовий ряд* - ряд значень будь-яких параметрів досліджуваного процесу за рівні проміжки часу.

*Скалярним часовим рядом*  $[x_i]_{i=1}^N$  називається масив з  $N$  чисел, що представляють собою значення деякої змінної  $x(t)$ , що спостерігається з деяким постійним кроком  $\tau$  по часу,  $t_i = t_0 + (i - 1)\tau$ ;  $x_i = x(t_i)$ ,  $i = 1..N$ . У аналізі часових рядів виділяються дві основні задачі: задача ідентифікації та задача прогнозу.

Задача ідентифікації при аналізі часового ряду передбачає відповідь на питання, які є параметри системи, що породила часовий ряд: розмірність вкладення, ентропія (перетворення) та інші. Розмірність вкладення — це мінімальне число динамічних змінних, що однозначно описують спостережуваний процес. Поняття ентропії пов'язане з передбачуваністю значень ряду і всієї системи.

*Задача прогнозу* має на меті за даними спостережень передбачити майбутні значення вимірюваних характеристик досліджуваного об'єкта, тобто побудувати прогноз на певний відрізок часу вперед. Є два основні класи методів прогнозу: локальні і глобальні. Такий поділ проводиться по області визначення параметрів апроксимуючої функції, що рекурентно встановлює наступне значення часового ряду за кількома попередніми.

Історично першими були розроблені глобальні методи, в яких на основі статистичного аналізу пропонувалося використовувати авторегресію, ковзне середнє і інші. Пізніше в рамках нелінійної динаміки були розроблені нові практичні методики:

- сингулярний спектральний аналіз (**SSA**), який є глобальним методом;
- локальна апроксимація (**LA**);
- поєднання **SSA-LA**.

Дослідження часових рядів базується на ідеї, що прогнозувати ряд можна, якщо замість змінних, що входять у вихідну систему, використовувати так звані вектори затримок спостережень  $z_i = [x_i, x_{i+1}, \dots, x_{i+m-1}]$ . Є два варіанти того, як можна подати затримку спостережень на вхід до апроксиматора:

- *Використовуючи неявне представлення.*

Час представляється ефектом, який він справляє на обробку сигналу, тобто неявним чином. Можна застосувати згортку до вектора затримок спостережень і отримати одне число, яке і вважати параметром. Таким чином ефект, який справляє час на сигнал можна контролювати змінюючи функцію згортки вектора.

- *Використовуючи явне представлення.*

Час має власне конкретне представлення. Наприклад, система ехолокації кажана посилає короткий частотно-модульований сигнал, при цьому встановлюючи єдиний рівень інтенсивності для кожного з частотних каналів на короткий період FM-розгортки. Для того щоб отримати точну інформацію про відстань до цілі, проводяться численні порівняння декількох різних частот, кодованих масивом слухових рецепторів. Коли відлуння отримується від об'єкта з невідомою затримкою, відповідає той нейрон (слухової системи), який має відповідну затримку в лінії. Таким чином оцінюється відстань до обсягу.

## 2. ПОСТАНОВКА ЗАДАЧІ

Розробити та реалізувати метод, за допомогою якого розв'язується наступна задача:

**Дані на вхід:**  $[x_t; t = \overline{1, n}]$  - часовий ряд (послідовність значень деякого показника впорядкована по даті фіксування; передбачається, що фіксування значень виконується з однаковим інтервалом), де  $x_t$ - елементи ряду,  $n$ - кількість елементів ряду;

**Вихід:**  $x_t = f(t)$  - функція прогнозування, що будується відштовхуючись від даних на вхід; за аргументи має номер елементу ряду (може бути більшим і меншим за  $n$ ), а за значення має  $t$ -ий елемент цього ряду.

Мають місце дані моніторингу первинної інвалідності в Україні за 24 роки (1992-2015). Для кожної адміністративної території, хвороби та типу населення дані представляють собою часовий ряд вигляду

$$[x_t; t = \overline{1, n}],$$

де  $x_t$ - значення первинної інвалідності на 10000 населення внаслідок хвороби  $x$ , зафіксоване у  $t$ -му році на певний адміністративній території для однієї з верств населення;  $n$ - кількість років, упродовж яких проводиться моніторинг (у даному випадку  $n = 24$ ).

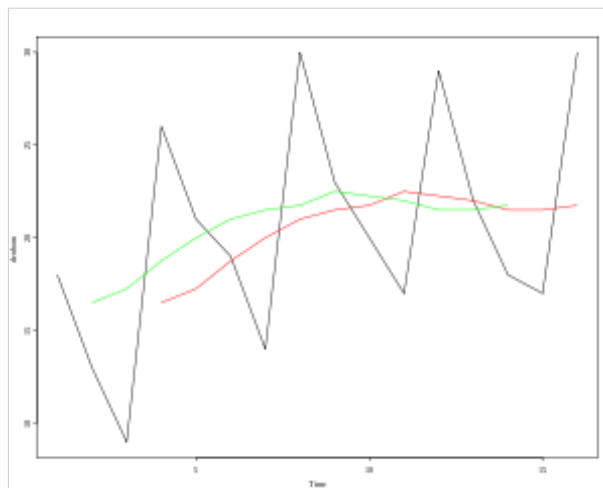
Ставиться задача прогнозування первинної інвалідності на наступний рік.

### 3. ПРОГНОЗУВАННЯ ЧАСОВИХ РЯДІВ

#### 3.1. Огляд методів прогнозування часових рядів

##### 3.1.1. Метод ковзного середнього ( *moving average, MA* ).

*Ковзне середнє* - загальна назва для сімейства функцій, значення яких в кожній точці визначення дорівнюють середньому значенню початкової функції за попередній період. Ковзне середнє зазвичай використовуються з даними часових рядів для згладжування короткострокових коливань і виділення основних тенденцій або циклів. Математично ковзне середнє є одним з видів згортки.



**Рис.1.** Просте ковзне середнє

Просте ковзне середнє, або арифметичне ковзне середнє **SMA** чисельно дорівнює середньому арифметичному значень вихідної функції за встановлений період і обчислюється за формулою:

$$SMA_t = \frac{1}{n} \sum_{i=0}^{n-1} p_{t-i} = \frac{p_t + p_{t-1} + \dots + p_{t-i} + \dots + p_{t-n+2} + p_{t-n+1}}{n}$$

де  $SMA_t$  - значення простого ковзного середнього в точці  $t$ ,  $n$ - кількість значень початкової функції для розрахунку ковзного середнього, чим ширше згладжує інтервал, тим більш плавним виходить графік функції,  $n$ значення вихідної функції в точці  $t - i$ .

На Рис.1 зображені вихідна функція і її 2 прості ковзаючі середні по чотирьом значенням ( $n = 4$ ): зелена лінія - центрування по середині інтервалу (справжній стан), червона лінія - зсув графіка вправо до останнього значенням вікна.

Експоненціально зважене ковзне середнє **ЕМА** - різновид зваженої ковзної середньої, ваги якої зменшуються експоненціально і ніколи не дорівнюють нулю. Визначається наступною формулою:

$$EMA_t = \alpha * p_t + (1 - \alpha) * EMA_{t-1}$$

де  $EMA_t$ - значення експоненціального ковзного середнього в точці  $t$ ,  $EMA_{t-1}$ - значення експоненціального ковзного середнього в точці  $t - 1$ ,  $p_t$  - значення вихідної функції в момент часу  $t$ ,  $\alpha$  - коефіцієнт що характеризує швидкість зменшення вагів, приймає значення від 0 і до 1, чим менше його значення тим більше вплив попередніх значень на поточну величину середнього.

### 3.1.2. Модель Бокса-Дженкінса (ARIMA)

ARIMA (autoregressive integrated moving average) - інтегрована модель авторегресії - модель і методологія аналізу часових рядів. Є розширенням моделей ARMA для нестационарних часових рядів, які можна зробити стаціонарними взяттям різниць деякого порядку від вихідного часового ряду.

Модель ARIMA(p,d,q) для нестационарного часового ряду  $X_t$  має вигляд:

$$\Delta^d X_t = c + \sum_{i=1}^p a_i \Delta^d X_{t-i} + \sum_{j=1}^q b_j \varepsilon_{t-j} + \varepsilon_t$$

де  $\varepsilon_t$  – стаціонарний часовий ряд,

$c, a_i, b_j$  – стаціонарний часовий ряд,

$\Delta^d$  – оператор різниці часового ряду порядку  $d$  (послідовне взяття  $d$  раз різниць першого порядку - спочатку від тимчасового ряду, потім від отриманих різниць першого порядку, потім від другого порядку і т.д.).

### 3.1.3. Метод аналізу сингулярного спектру (SSA)

*SSA (Singular spectrum analysis)* - метод аналізу часових рядів, заснований на перетворенні одновимірний часового ряду в багатовимірний ряд з подальшим застосуванням до отриманого багатомірний тимчасового ряду методу головних компонент.

Спосіб перетворення одновимірний ряду в багатовимірний є «згортку» часового ряду в матрицю, що містить фрагменти тимчасового ряду, отримані з деяким зрушенням. Загальний вигляд процедури нагадує «гусеницю», тому сам метод нерідко так і називають - «Гусениця»: довжина фрагмента називається довжиною «гусениці», а величина зсуву одного фрагмента щодо іншого кроком «гусениці».

SSA може бути використаний без попереднього завдання моделі ряду для аналізу довільних, в тому числі, нестаціонарних, рядів. Основна мета SSA - розкласти ряд в суму інтерпретованих компонент, таких як тренд, періодичні компоненти, шум. При цьому знання параметричної форми цих компонент не потрібно.

#### Базовий алгоритм SSA

*Крок 1.* Будується  $L \times K$  траєкторна матриця ряду  $X$  наступним чином:

$$X = [X_1 : \dots : X_K] = (x_{ij})_{i,j=1}^{L,K} = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_K \\ x_2 & x_3 & x_4 & \dots & x_{K+1} \\ x_3 & x_4 & x_5 & \dots & x_{K+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_L & x_{L+1} & x_{L+2} & \dots & x_N \end{bmatrix},$$



де  $X_i = (x_i, \dots, x_{i+L-1})^T$  - вектори вкладення довжини  $L$ . Матриця  $X$  є ганкелевою, тобто має однакові елементи на антидіагоналях  $i + j = \text{const}$ .

*Крок 2.* Виконується сингулярне розкладання (SVD) траєкторної матриці  $X$ . Нехай  $S = XX^T$ , позначимо  $\lambda_1, \dots, \lambda_L$  власні числа  $S$ , взяті у незростаючому порядку і ортонормовану систему власних векторів матриці  $S$ . Нехай  $d = \text{rank}X = \max\{i: \lambda_i > 0\}$  та  $V_i = \frac{X^T U_i}{\sqrt{\lambda_i}}$ . У цих позначеннях сингулярне розкладання траєкторної матриці  $X$  може бути записано як  $X = X_1 + \dots + X_d$ .

*Крок 3.* Множина всіх індексів  $\{1, \dots, d\}$  розбивається на  $m$  непересічних підмножин  $I_1, \dots, I_m$ . Нехай  $I = \{i_1, \dots, i_p\}$ . Тоді результуюча матриця  $X$ , що відповідає групі  $I$ , записується як  $X_I = X_{i_1} + \dots + X_{i_p}$ . Результуючі матриці обчислюються за групами і згруповане SVD розкладання матриці  $X$  може бути записано як  $X = X_{I_1} + \dots + X_{I_m}$ .

*Крок 4.* Кожна матриця  $X_{I_j}$  згрупованого розкладання ганкелізується (усереднюється по анти-діагоналям) і потім отримана ганкелева матриця трансформується в новий часовий ряд довжини  $N$  на основі взаємно-однозначної відповідності між ганкелевими матрицями і тимчасовими рядами. Діагональне усереднення, застосоване до кожної результуючої матриці  $X_{I_k}$ , виробляє відновлені ряди  $\tilde{X}^{(k)} = (\tilde{x}^{(k)}_1, \dots, \tilde{x}^{(k)}_N)$ . Таким чином, вихідний ряд розкладається в суму  $m$  відновлених рядів:

$$x_n = \sum_{k=1}^m \tilde{x}^{(k)}_n \quad (n = 1, 2, \dots, N)$$

### 3.1.4. Локальна апроксимація (LA)

Відповідно до теорії Такенса-Мане прийнятний опис фазового простору динамічної системи можна отримати, якщо взяти замість реальних змінних системи  $p$ -мірні вектори затримок із значень ряду в послідовні моменти часу. При виконанні умови  $p \geq 2d + 1$ , де  $d$  - розмірність вкладення, можливо реконструювати фазовий простір (простір станів) системи. За умови

стаціонарності часового ряду на базі цієї реконструкції будується прогноз його подальшої динаміки.

Способи визначення величини  $p$ :

- Алгоритм Грассбергера-Прокаччі. Недолік методу: неефективний при роботі з короткими (до 104 точок) тимчасовими рядами.
- Інші методи теж мають свої недоліки: складність реалізації, велика тривалість розрахунків, неоднозначність або сумнівність результатів.

У зв'язку з цим величина  $p$ , за винятком модельних прикладів, в яких вона достовірно відома, як правило, визначається емпірично. Головний критерій у цьому випадку - вибір такого  $p$ , починаючи з якого припиняється якісна зміна прогнозу.

### Побудова прогнозу на один крок вперед

*Крок 1.* Перетворимо скалярний тимчасовий ряд, що містить  $N$  значень, в матрицю затримок:

$$\{x_1, x_2, \dots, x_N\} \rightarrow \mathbf{X}_{p \times (N-p+1)} = \begin{pmatrix} x_p & x_{p+1} & \dots & x_N \\ \vdots & \vdots & \ddots & \vdots \\ x_2 & x_3 & \dots & x_{N-p+2} \\ x_1 & x_2 & \dots & x_{N-p+1} \end{pmatrix}$$

Розмірність матриці  $\mathbf{X}$  визначається кількістю затримок  $p$ .

Після побудови матриці затримок обирається вид локального уявлення, тобто вид функції, що зв'язує таке значення ряду з попередніми:

$$x_{t+1} = f(x_t, a)$$

де  $a$  - вектор параметрів уявлення.

Найбільш поширений варіант - це лінійна апроксимація першого (LA1) порядку:

$$x_{t+1} = a_0 + x_t^T a$$

*Крок 2.* Вибір найближчих сусідів - виділення локальної підобласті фазового простору. Будемо вважати сусідніми до стартового вектору  $\mathbf{x}_{N-p+1}$  вектори, що задовольняють умові:

$$\{\mathbf{x}_s\} : \sum_{s \in \omega_{\Xi}} \|\mathbf{x}_{N-p+1} - \mathbf{x}_s\| \rightarrow \min_{\omega_{N-p+1}}$$

де  $s \in \omega_{\Xi} \subset \omega_{N-p+1}$ ,  $\omega_{N-p+1} \equiv \{1, \dots, N - p + 1\}$ .

Норма зазвичай береться евклідова, хоча можливе використання й інших норм. Вибір сусідів автоматично визначає локальну підобласть, в якій параметри уявлення покладаються незмінними.

Даний етап алгоритму - головна відмітна ознака методу і найважливіший його етап. Вибір "хороших" сусідів визначає якість прогнозу.

*Крок 3.* Оцінка параметрів моделі і побудова прогнозу на один крок вперед.

Вибором типу лінійної апроксимації і розмірності реконструкції  $p$  встановлюється кількість невідомих параметрів, які потрібно оцінити. При цьому коло використовуваних даних обмежується набором сусідів стартового вектора.

Параметри моделі (вектор  $\mathbf{a}$ ) оцінюються методом найменших квадратів (МНК). Це найпоширеніший і найбільш ефективний метод. Оцінка за МНК для вектора  $\mathbf{a}$ , позначимо її як  $\hat{\mathbf{a}}$  знаходиться з умови:

$$\hat{\mathbf{a}} : \sum_{\omega_s} \left( x_{s+1} - f(\mathbf{x}_s, \hat{\mathbf{a}}) \right)^2 \rightarrow \min_{\hat{\mathbf{a}}}.$$

Однак в більшості випадків застосування МНК в чистому вигляді неможливо через виродженість матриці факторів, що є наслідком взаємної близькості сусідів. Тому зазвичай застосовується так званий моє сингулярне розкладання (SVD - Singular Value Decomposition). При цьому не завжди враховується, що оцінки, що даються цим методом, в загальному випадку зміщені, сильно залежать від машинної точності і вибору мінімального значущого сингулярного числа. Отже, при використанні SVD важливо завжди контролювати стійкість отриманих результатів.

Оцінивши значення параметрів апроксимації, неважко побудувати прогноз наступного значення ряду (стартовий вектор, позначимо індексом  $L$ ):

$$\hat{x}_{L+1} = f(x_L, \hat{a})$$

Таким чином, пройшовши три описаних кроку алгоритму LA, можна побудувати прогноз одного нового значення ряду.

### 3.1.5. Нейромережеве прогнозування часових рядів

При вирішенні завдань прогнозування роль нейронної мережі полягає в передбаченні майбутньої реакції системи по її попередній поведінці. Володіючи інформацією про значеннях змінної  $x$  в моменти, які передують прогнозуванню:

$$x(k-1), x(k-2), \dots, x(k-N),$$

мережа виробляє рішення, яким буде найбільш ймовірне значення послідовності  $x(k)$  в поточний момент  $k$ . При цьому нейронна мережа грає роль універсального апроксиматора функції від декількох змінних, реалізуючи нелінійну функцію:  $y = f(x)$ , де  $x$  - це вхідний вектор, а  $y$  - реалізація векторної функції декількох змінних.

Відповідно до класичної теорії апроксимації, функції, що апроксимується представляється в наступному вигляді:

$$y(x) = \sum_i \alpha_i \psi_i(x)$$

Тут набір функцій  $\psi_i(x)$ , як правило, вибирається апріорі, виходячи з інтересів автора конкретного методу апроксимації і деяких властивостей даних функцій, доказуваних і використовуваних в процесі апроксимації.

Для нейронних мереж базове вираз для функції, що апроксимується виглядає дещо інакше, наприклад, для тришарової нейронної мережі з послідовними зв'язками:

$$y(x) = \psi \left( \sum_i \alpha_i \psi \left( \sum_j \alpha_{ij} \psi \left( \sum_k \alpha_{kj} x_k \right) \right) \right)$$

Зазвичай формування пари навчальних прикладів здійснюється за принципом «ковзного вікна»: тобто береться певний відрізок тимчасового ряду і з нього виділяється кілька спостережень, які і будуть представляти собою вхідний вектор. Значним бажаного виходу в навчальному прикладі буде наступне по порядку спостереження. Потім «ковзне вікно» зсувається на одну позицію в напрямку зростання часу, і процес формування наступної пари навчальної вибірки повторюється. Частинка тимчасового ряду для навчання нейронної мережі повинен бути дорівнює обсягу навчальної вибірки (кількості прикладів). При виборі числа входів нейронної мережі, що працює в режимі передбачення часового ряду, слід вибирати розумний компроміс між глибиною передбачення і якістю навчання, оскільки збільшення числа входів нейронної мережі відповідно зменшує кількість навчальних прикладів.

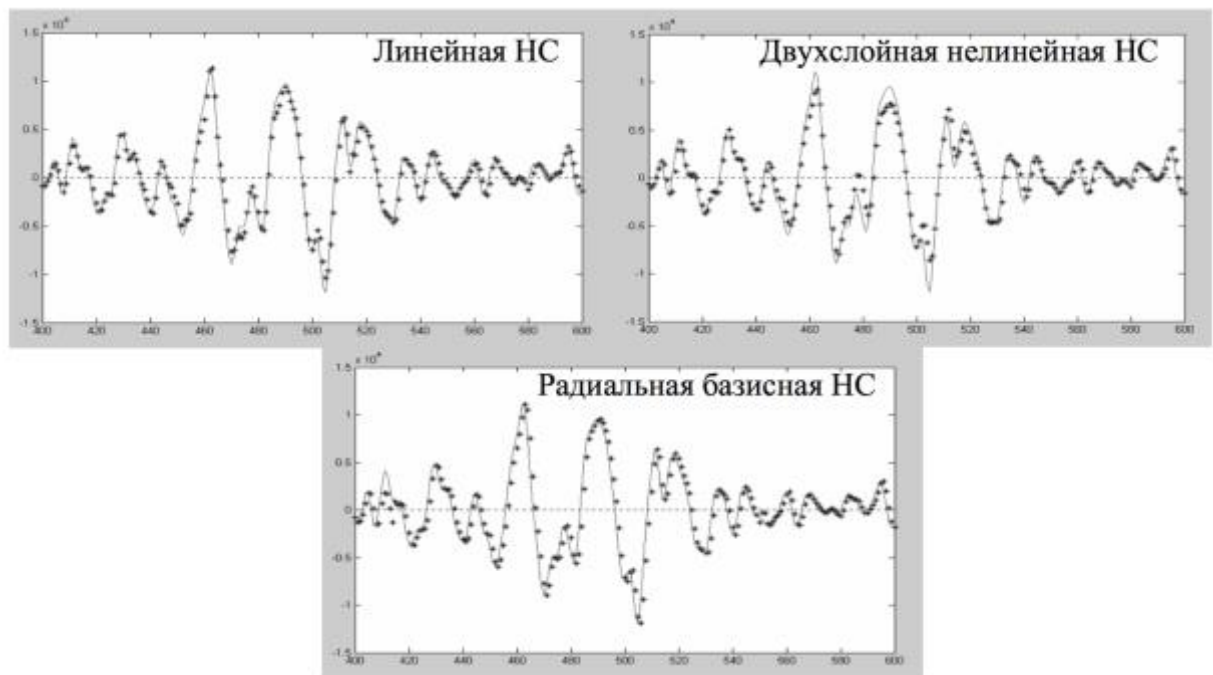
Згідно із загальноприйнятою принципом, якщо більш складна модель не дає результат краще, ніж більш проста, то з цих двох моделей слід віддати перевагу останню. Найпростішою архітектурою з нейронних мереж прямого поширення мають лінійні нейронні мережі. Будь-яка багат шарова нейронна мережа з лінійною функцією активації нейронних елементів еквівалентна одношарової лінійної нейронної мережі, тому лінійні нейронні мережі, як правило, не мають прихованих шарів. Під час своєї роботи лінійна нейронна мережа фактично примножує вектор входів на матрицю ваг, а потім до отриманого вектору додає вектор зміщення. Вихідне значення такої мережі визначається виразом:

$$y = W p + b,$$

де  $y$  - вихід мережі;  $W$  - матриця ваг;  $p$  - вектор входів;  $b$  - вектор зміщення.

Для організації «ковзного вікна» можна запропонувати використовувати на вході нейронної мережі лінію затримки. Послідовність значень вхідного сигналу  $p(k)$  надходить на лінію затримки, що складається з  $N - 1$  блоку запізнювання (по довжині «ковзного вікна»). Вихід лінії затримки -  $N$ -мірний вектор  $pd$ , складений з значень входу в моменти часу  $k, k - 1, \dots, k - N + 1$ .

Результат роботи лінійної нейронної мережі, що складається з одного лінійного нейрона і лінії затримки з десяти блоків запізнювання представлений на малюнку 1. Навчання даної нейронної мережі вироблялося за правилом Відроу-Хоффа, який передбачає мінімізацію середньоквадратичної помилки нейронної мережі.



**Рис.2.** Результат передбачення нейронної мережі: - вхідні дані;  
\* прогноз.

Використання нелінійної функції активації дозволяє налаштувати нейронну мережу на реалізацію нелінійних зв'язків між входом і виходом. Однак якщо вихідний шар багат шарової мережі буде використовувати нелінійні сігмоїдні функції активації, то виходи мережі будуть обмежені. Якщо ж у вихідному шарі використовуються лінійні нейрони, то виходи мережі можуть приймати довільні значення. Кількість нейронів у вихідному шарі визначається метою вироблених нейронною мережею обчислень. Таким чином, вихідний шар нейронної мережі для вирішення задачі передбачення динаміки економічних показників був прийнятий містити один лінійний нейрон. Оскільки функція активації нейронних елементів прихованого шару має область чутливості для входів в діапазоні трохи ширше інтервалу від -1 до

+1, вхідні дані були піддані препроцесуванню приведенням до діапазону  $[-1;1]$  за формулою:

$$PN = 2(P - \min P) / (\max P - \min P) - 1$$

де  $PN$  - матриця нормованих векторів входу;  $P$  - матриця векторів входу;

$\min P$  - мінімальний елемент входу;  $\max P$  - максимальний елемент входу.

Вихідна значення нейронної мережі:

$$y = \sum_{i=1}^s v_i a_i + b$$

де  $v_i$  -  $i$ -й ваговий коефіцієнт вихідного нейрона;  $b$  - зсув вихідного нейрона;  $s$  - кількість нейронів в прихованому шарі;  $a_i$  - вихід  $i$ -го нейрона прихованого шару.

Вихідні значення нейронних елементів прихованого шару:

$$a_i = F\left(\sum_{j=1}^n w_{ij} p_j + b\right)$$

де  $F$  - сігмоїдна функція активації прихованого шару ;  $w_{ij}$  - ваговий коефіцієнт від  $j$ -го входу до  $i$ -му нейрона;  $p_j$  -  $j$ -й елемент входу;  $b$  - зсув нейрона прихованого шару.

## 4. ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ПРОГНОЗУВАННЯ ЧАСОВИХ РЯДІВ

### *4.1. Функціональні можливості та структура програми*

Програма складається з двох основних блоків: бібліотеки, що реалізує програмне забезпечення прогнозування часових рядів, користувацького інтерфейсу, що надає змогу скористатися можливостями бібліотеки без написання додаткових програм.

Бібліотека написана мовою C++ і розділяється на дві логічні частини: незалежну від середовища, в якому буде виконуватися програма, і залежну. Таки спосіб розділення логіки надає змогу легко від'єднати незалежну частину бібліотеки і скористатися нею деінде.

Є такі незалежні від середовища виконання програмні блоки:

1. *Matrix* задає і реалізує інтерфейс і алгебраїчні операції для матриці дійсних чисел;
2. *Serialization* описує і визначає функції для серіалізації і десеріалізації (кодування і розкодування у буфер) важливих для бібліотеки структур даних;
3. *VectorAlgebra* описує і визначає функції алгебри векторів;
4. *Perceptron* відповідає за реалізацію багатошарового перцептрону. Надає можливість навчити його будь-якою кількістю паттернів завдяки функції `back_prop`. Також можна контролювали процес навчання завдяки функціям `forward_prop`, `put_errors` і `flush`, перша з яких відповідає за ForwardPropagation і запам'ятовує у буфері виходи кожного зі слоїв, друга за BackPropagation і наколює градієнт у буфері, а третя використовує цей градієнт для зміни вагів і очищує буфер. Використовувати ці функції при наявності масиву паттернів треба так:



(викликати `forward_prop`, передати помилку у `put_errors`) повторити для кожного патерну, викликати `flush`. Розділення способів навчання на більш простий і більш складний (1 і 3 функції) потрібне для того, щоб користувач об'єкта класу `Perceptron` мав змогу отримати помилки для першого слою перцептрону від функції `put_errors`, тобто, якщо дані для навчання перцептрону не є правильними (початковими, незмінними), користувач, базуючись на отриманих помилках, має змогу змінити дані для навчання перцептрону. Для цієї можливості і був створений клас `Perceptron`, а не використана вже готова програма чи бібліотека, бо такої можливості при навчанні нейронних мереж зазвичай не надають. При закінченні навчання перцептрону користувачу об'єкта класу `Perceptron` рекомендується викликати функцію `release_buffer`, що вивільняє усі ресурси, які були потрібні при навчанні перцептрону, але при використанні вже не є потрібними. Об'єкт класу `Perceptron` можна перевести у бінарний вигляд та зберігати у файлах чи бінарних змінних (сериалізувати); це надає змогу користувачеві відтворювати вже навчений перцептрон без витрачання часу на навчання. Сериалізацію реалізують функції `write_to_stream` і `from_stream`.

5. *GammaUnit* (гамма-юніт) прихована від користувача частина програми, але дуже важлива. Відповідає за згортку ряду, реалізує принцип гамма-пам'яті. Має змогу змінювати ваги гамма-пам'яті відповідно до величини помилки, що йому передалась. Цей логічний блок навчається у зв'язці з перцептроном: перцептрону передають данні на вхід, серед яких є значення згортки ряду від конкретного гамма-юніту; перцептрон повертає значення помилок вхідних даних; гамма-юніту змінює значення свого вагу або накопичує градієнт — залежить від кількості патернів.
6. *GammaNN* обгортає `Perceptron` для використання з часовими рядами, надає і реалізує інтерфейс для навчання перцептрону та гамма-юнітів, можливість контролювати кількість гамма-юнітів (`units`) і кількість

вільних входів (`trace_size`), що бачать останні значення часового ряду. Має функцію `learn` для навчання, куди передаються номери елементів часового ряду, що є патернами для навчання. Має оператор `[]`, для отримання об'єкту часового ряду за його номером. Оптимізує використання `[]` таким чином, що запам'ятовує усі елементи ряду (елементи початкового ряду також), що були колись отримані. Ця оптимізація є вдалою, бо для визначення наступних значень часового ряду треба звертатися до попередніх, котрі треба було б теж обчислювати, якщо б їх не запам'ятали, і так доки не дійдемо до елементів початкового ряду. Як і `Perceptron`, `GammaNN` можна серіалізувати таким самим чином: функції `write_to_stream` і `from_stream`.

7. Тестування підпрограм. Для тестування `Perceptron` використовувалися навчальні дані задачі XOR та інші. Для тестування `GammaNN` використовувалися навчальні дані: послідовність числа 1, послідовність нулів та одиниць, двомірна послідовність нулів та одиниць, зростаюча послідовність чисел за кроком 1.

Залежність бібліотеки від середовища зумовлена тим, що було задумано створити бібліотеку для мови програмування **R** на якій був написаний інтерфейс.

Є такі функції, що експортуються у середовище мови **R**:

1. `learn` приймає на вхід початковий часовий ряд і параметри на для навчання `GammaNN`; віддає `NNptr` - вказівник на об'єкт класу `GammaNN`. З середовища мови **R** неможливо звертатися до членів об'єкту `GammaNN`, тому для нього цей вказівник виконує роль дескриптора, який потрібно передавати в експортованій бібліотекою інтерфейс користування.
2. `get_series` приймає `NNptr` і номери об'єктів часового ряду; віддає масив елементів часового ряду, номери яких прийшли на вхід.

3. `get_series_length` приймає `NNptr`; віддає кількість елементів ряду, що вже визначені в об'єкті `GammaNN`.
4. `to_GammaNN` приймає строку (сериалізований `GammaNN` об'єкт); віддає `NNptr`.
5. `to_str` обернено до `to_GammaNN`
6. `create_from_file` приймає путь до файлу, де серіалізований `GammaNN` об'єкт; віддає `NNptr`.
7. `write_to_file` приймає `NNptr`, путь до файлу, куди серіалізувати `GammaNN` об'єкт.

Інтерфейс користувача створений на мові **R** з використанням програмного пакету **Shiny**. Таке поєднання визначає структуру частини програми відповідальної за інтерфейс. Є два логічних блоки, що відповідають за інтерфейс користувача: `UI` і `server`. Інтерфейс користувача в даному випадку — клієнт-серверна програма. `UI` відповідає за розміщення елементів інтерфейсу на екрані, а `server` за обробку даних.

#### ***4.2.Інструкція користувача***

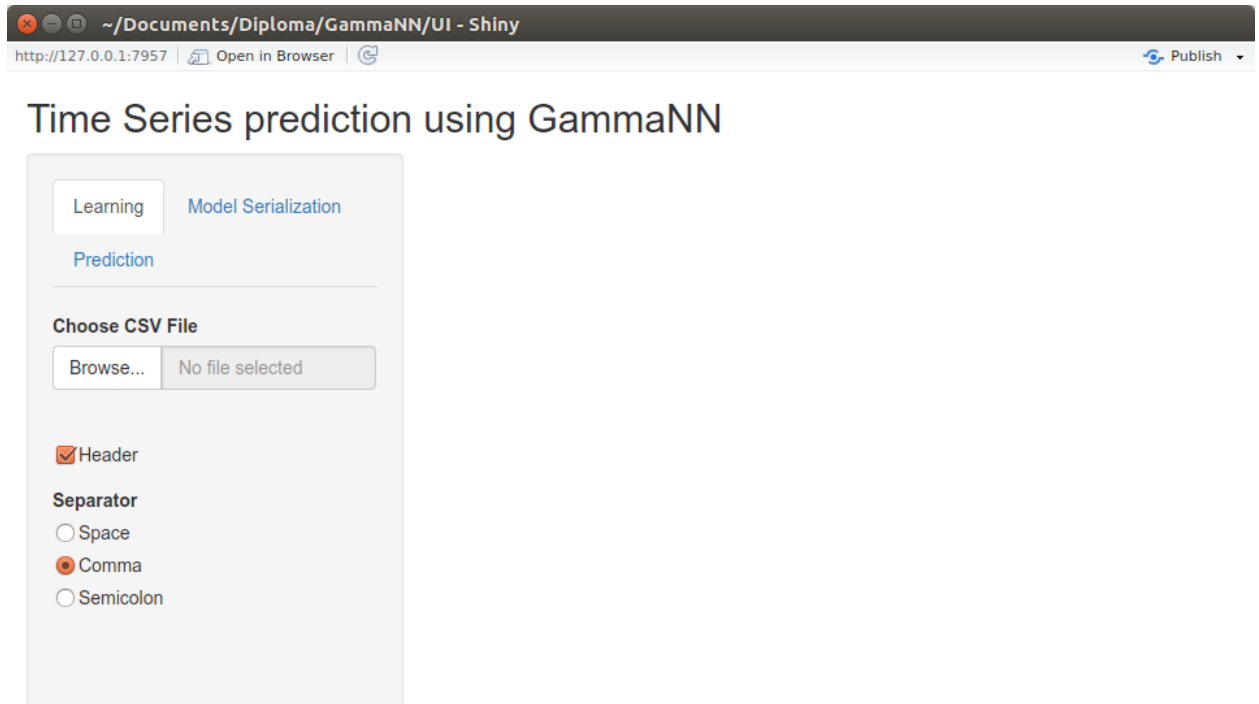
Розроблено програмний продукт, призначений для розв'язання задач прогнозування часових рядів. Алгоритм створений на мові `C++`, інтерфейс користувача створений на мові `R` з використанням програмного пакету `Shiny`.

#### ***Опис інтерфейсу програми:***

Для запуску програми необхідно встановити на комп'ютер інтерпритатор мови `R` та середовище розробки `RStudio`.

Щоб відкрити програму, необхідно відкрити файл ***UI.Rproj***.

Ви бачите інтерфейс:



*Рис.3. Головна сторінка програми*

Бачимо три вкладки: *Learning*, *Model Serialization* та *Prediction*.

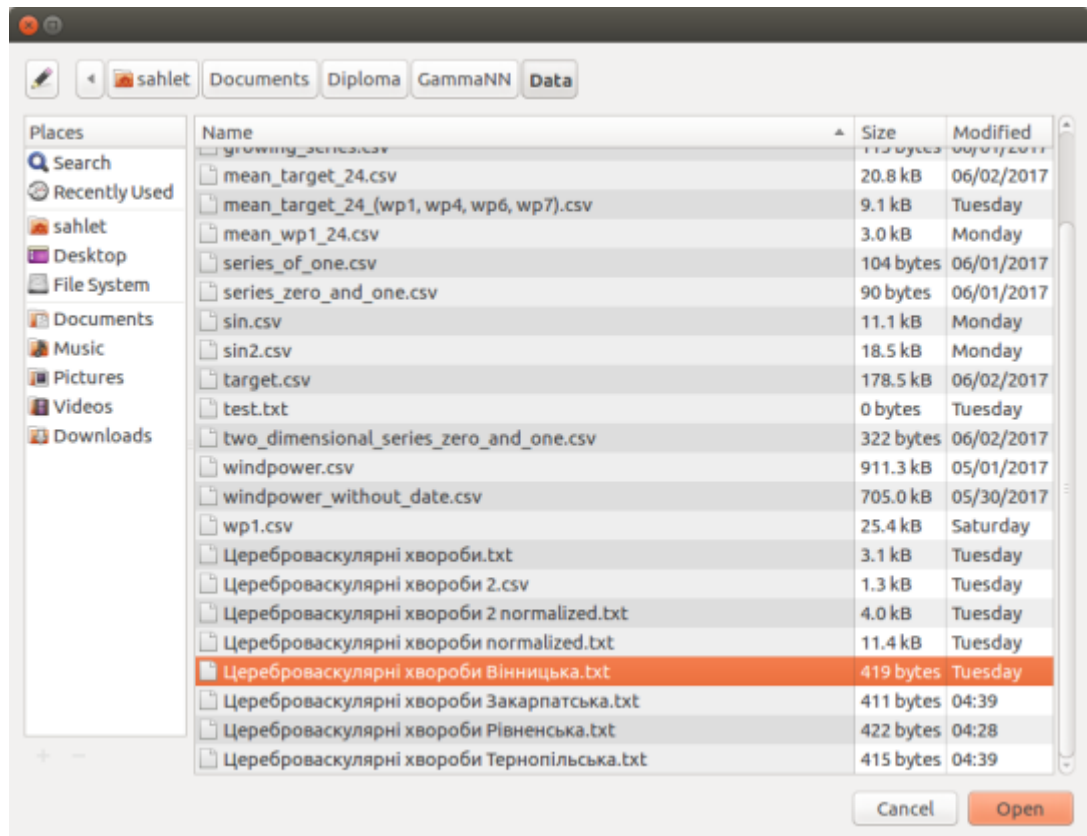
У вкладці ***Learning*** необхідно вибрати текстовий або .scv файл з даними часового ряду.

Галочка ***Header*** ставиться тоді, коли у файлі є назви стовпців.

***Separator*** відповідає за те, який символ сприймається як розділовий знак:

- *Space* – пробіл,
- *Comma* – кома,
- *Semicolon* – крапка з комою.

Натиснути кнопку ***Browse*** та обрати файл:



*Рис.4. Обираємо файл з часовим рядом*

Після відкриття файлу стає доступним інтерфейс для навчання моделі.

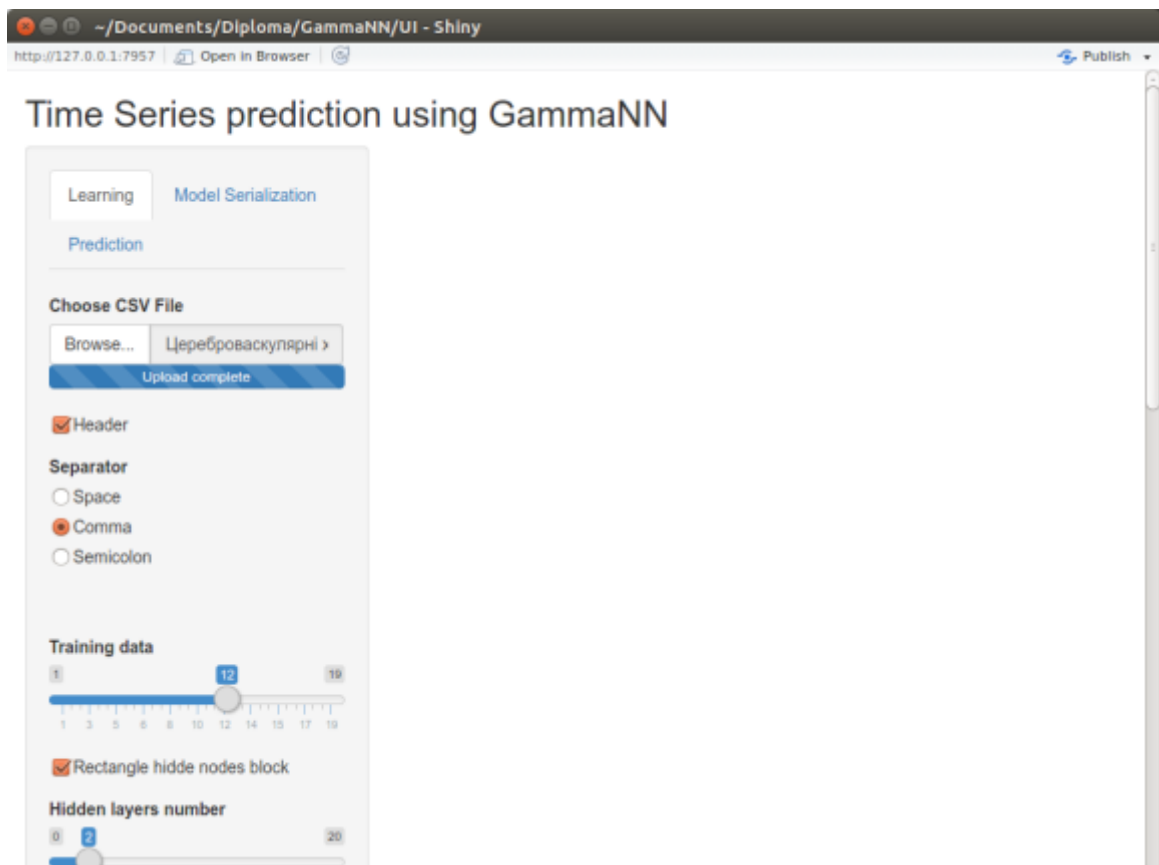


Рис.5. Інтерфейс навчання моделі

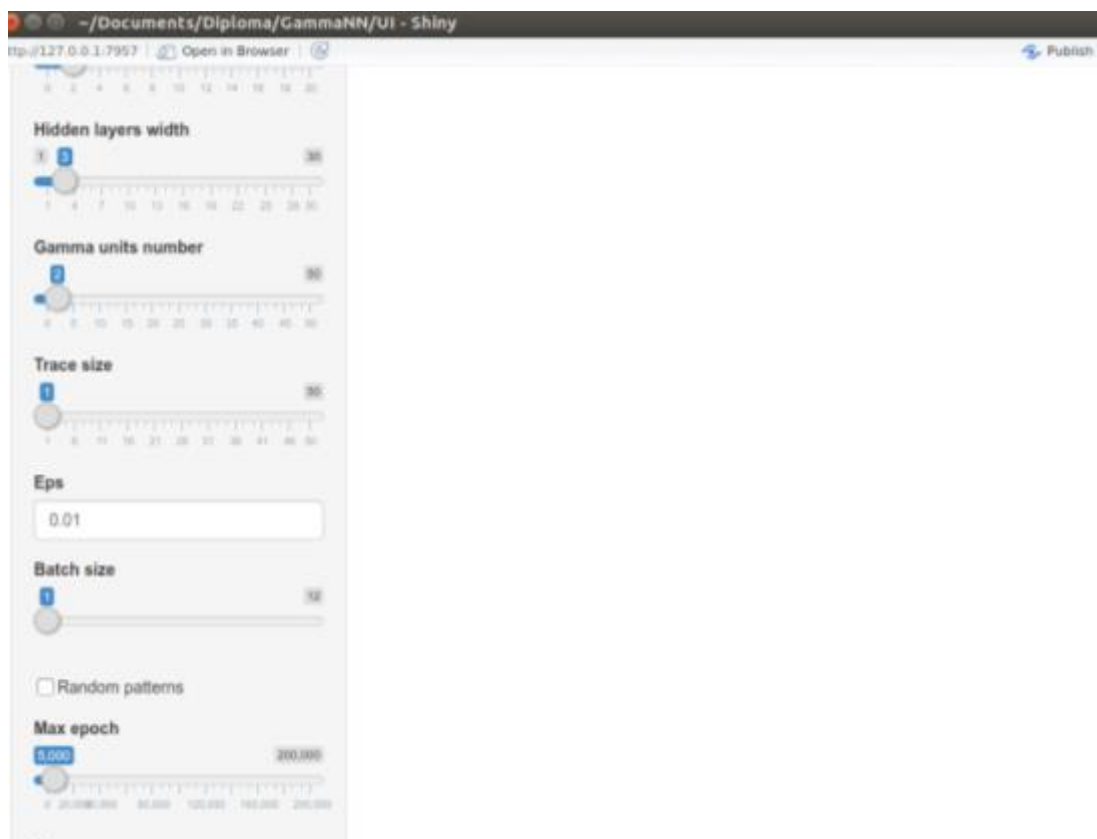


Рис.6. Інтерфейс навчання моделі

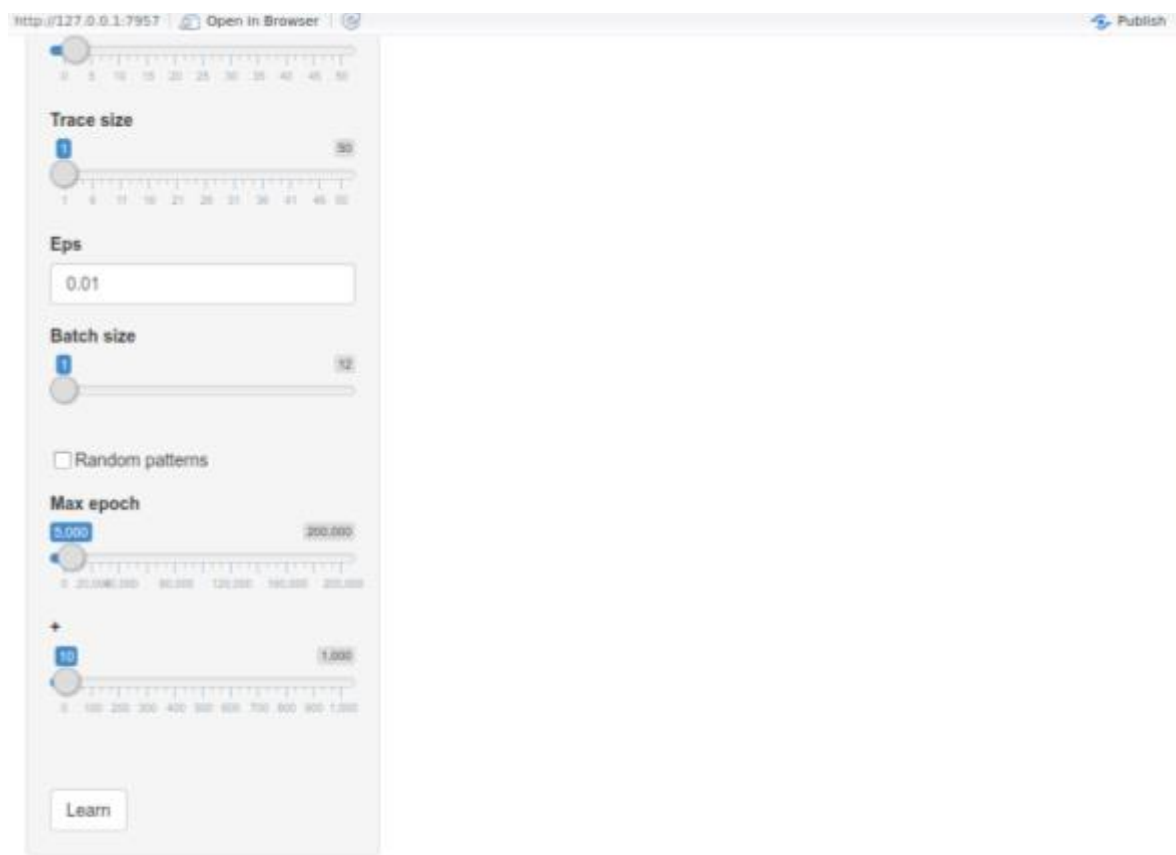
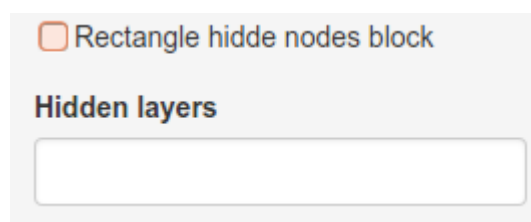


Рис.7. Інтерфейс навчання моделі

***Training data*** – встановлює кількість даних, які будуть використовуватись у моделі навчання (від 50% до 80% усіх даних);

***Rectangle hide nodes block*** – задання прямокутного масиву прихованих вузлів. Якщо прибрати галочку, то у ***Hidden layers*** записується через кому кількість прихованих вузлів у відповідному шарі (кількість чисел = кількість прихованих вузлів).



***Рис.7. Hidden layers***

***Hidden layers number*** – кількість прихованих шарів;

***Hidden layers width*** – ширина прихованого шару;

***Gamma units number*** – кількість гамма-блоків;

***Trace size*** – кількість останніх видимих елементів ряду;

***Eps*** – параметр установки алгоритму;

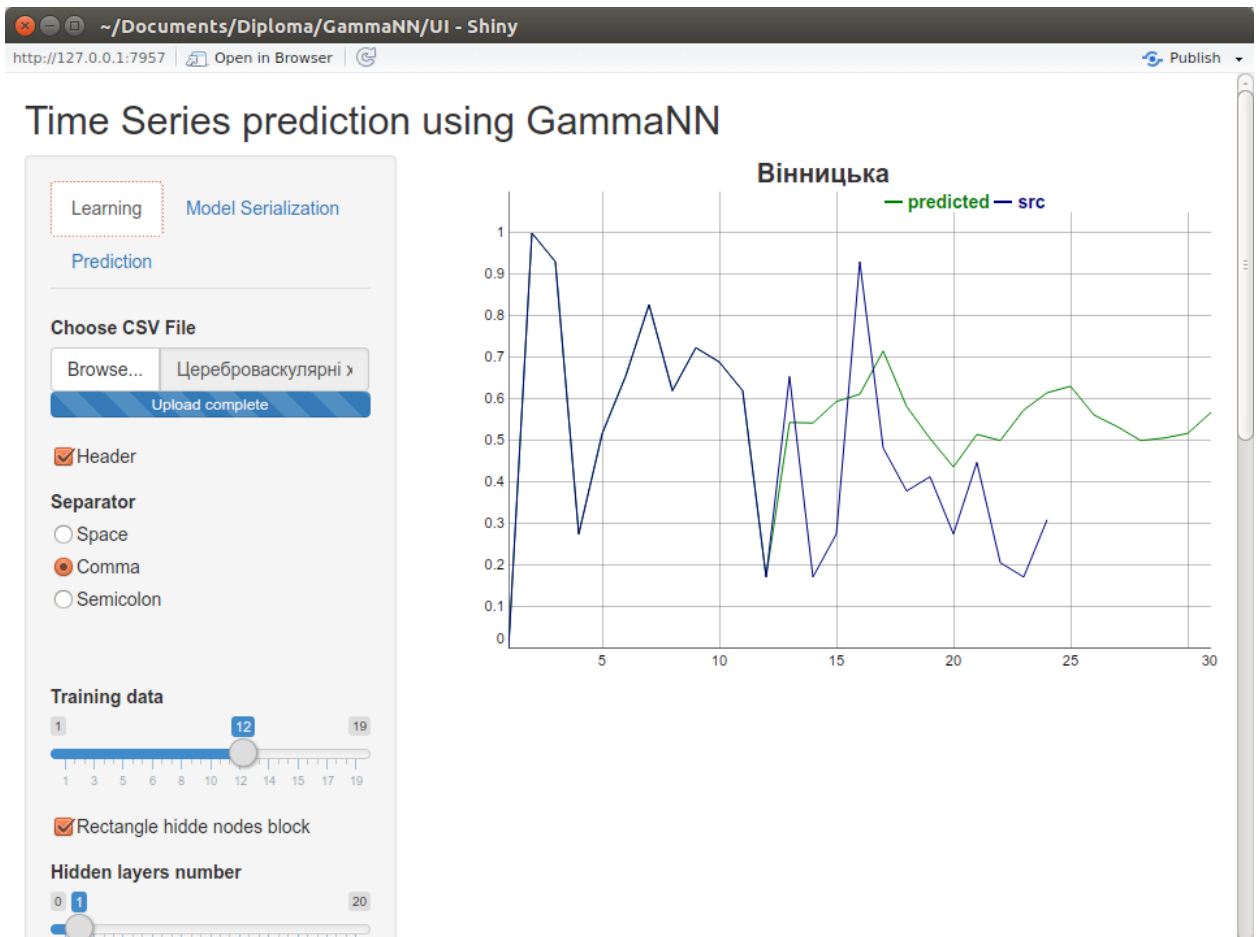
***Batch size*** – кількість елементів у навчальній ітерації;

***Random patterns*** – випадковим чином обираються елементи для навчання;

***Max epoch*** – максимальна кількість епох;

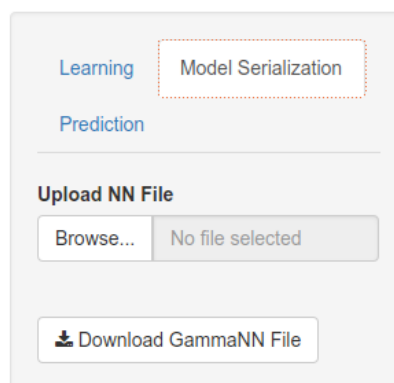
+ – кількість епох.

Для запуску роботи програми натиснути ***Learn***.



*Рис.7. Результат роботи програми*

У вкладці **Model Serialization** можна обрати або зберегти файл з моделлю.



*Рис.4. Вкладка Model Serialization*

У вкладці **Prediction** з'явився результат роботи програми:



Learning    Model Serialization

Prediction

Range:

1 23 219

Mean relative error 149.25 %  
RMSE 0.064

Choose CSV File to set src data

Browse...    Цереброваскулярні хі

Upload complete

☒ Header

Separator

☐ Space

☒ Comma

☐ Semicolon

*Рис.7. Вкладка Prediction*

***Range*** – проміжок на якому брати номери елементів з ряду;

***Mean relative error*** – середня відносна помилка;

***RMSE*** – *root mean squared error* – корінь середньоквадратичної помилки.

## 5. РЕЗУЛЬТАТИ ОБЧИСЛЮВАЛЬНИХ ЕКСПЕРИМЕНТІВ

Були проведені обчислювальні експерименти з використанням таких даних:

Програма випробувана на даних первинної інвалідності в Україні; даних сили вітру; даних щоденної кількості проданого товару.

1. *Первинна інвалідність в Україні.* З них взято дані про цереброваскулярні хвороби Вінницької, Закарпатської, Рівненської та Тернопільської областей.

2. *DAYSALES* – масив даних щоденної кількості проданого товару. Взято 196-ту та 285-ту торгову точку.

3. *Дані сили вітру* – масиви даних середнього значення сили вітру за день. Дані зібрані з декількох датчиків в одному районі. Взято дані окремо з датчику WP1; разом з датчиків WP1, WP4, WP6, WP7.

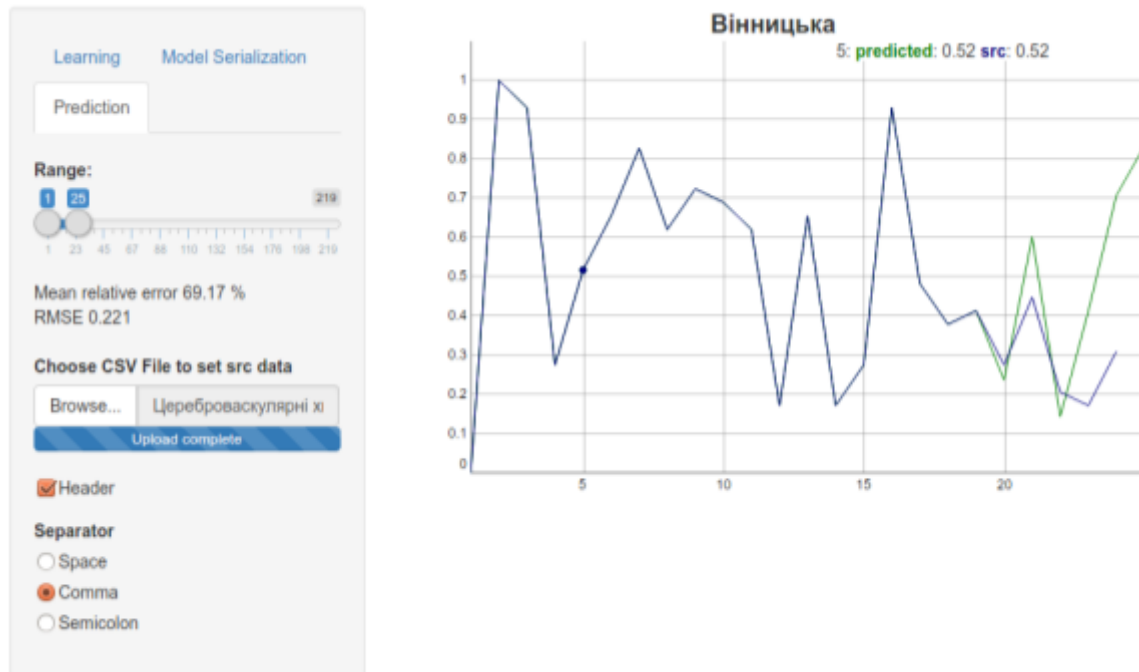
Будемо вважати, що прогноз на один крок вперед задовільний, якщо :

- $RMSE$  ( *root mean squared error* – корінь середньоквадратичної помилки)  $\leq 0,1$   
або
- $MRE$  (*mean relative error* – середня відносна помилка)  $\leq 10\%$ .

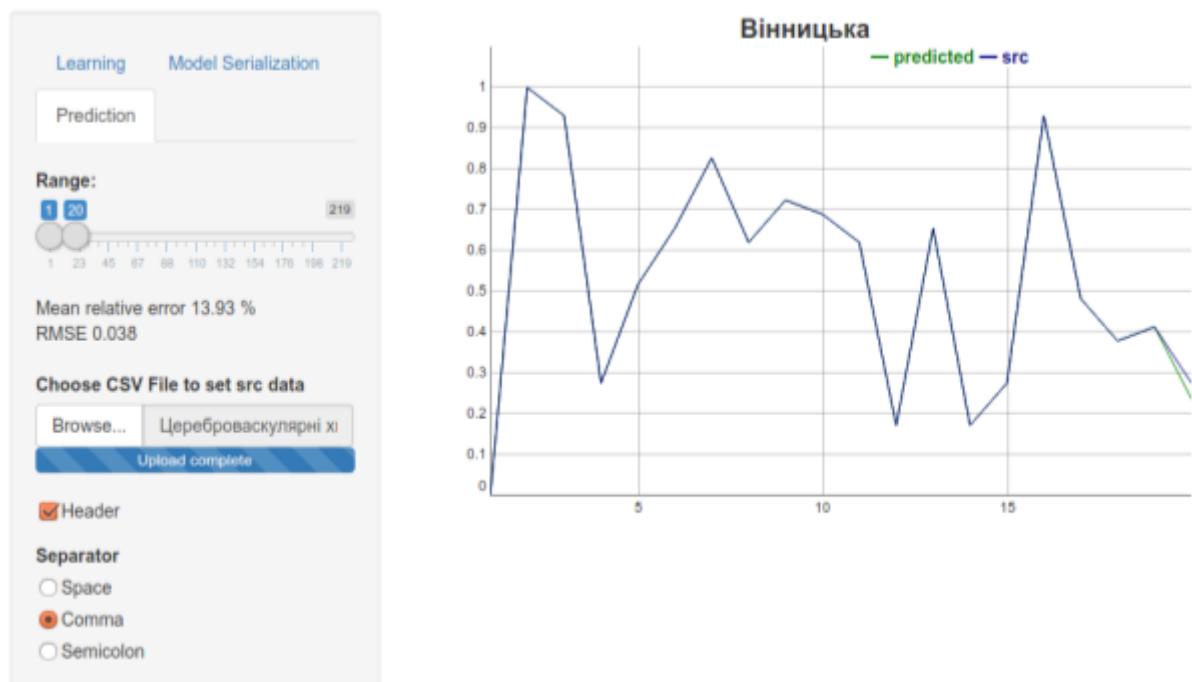
Будемо вважати, що загальний прогноз характеру ряду задовільний, якщо динаміка прогнозованого ряду схожа з динамікою початкового ряду.

### Прогнозування цереброваскулярних хвороб:

Мають місце дані моніторингу первинної інвалідності в Україні за 24 роки (1992-2015).



**Рис.7.** Прогнозування “Цереброваскулярні хвороби Вінницької області”



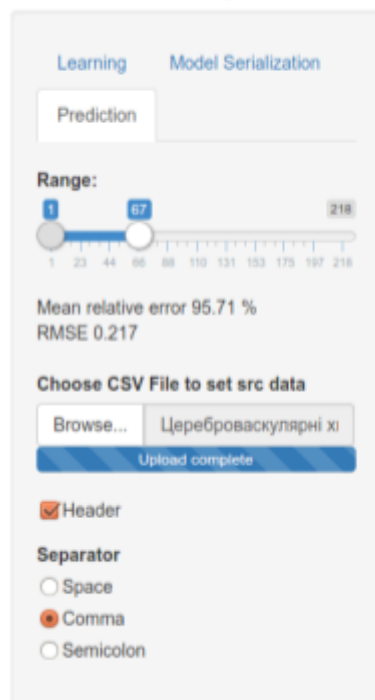
**Рис.7.** Прогнозування для першого елемента “Цереброваскулярні хвороби Вінницької області”

Ця модель має 4 гамма-юніти.

Як бачимо, для першого прогнозування  $RMSE = 0,038$ ;  $RME=13,93\%$  і динаміка прогнозованого ряду схожа з динамікою початкового ряду, отже загальний прогноз характеру ряду і прогноз на один крок вперед задовільні.

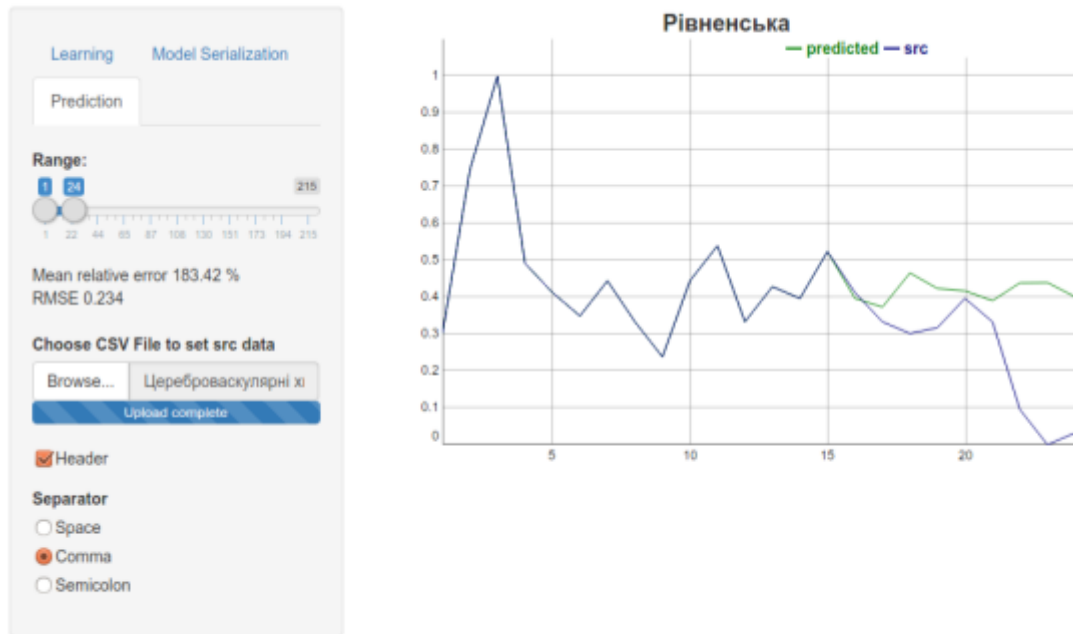
Надалі ми не будемо розглядати графік для прогнозування на один крок вперед, а просто будемо вказувати  $RMSE$  та  $RME$  для нього.

### Time Series prediction using GammaNN



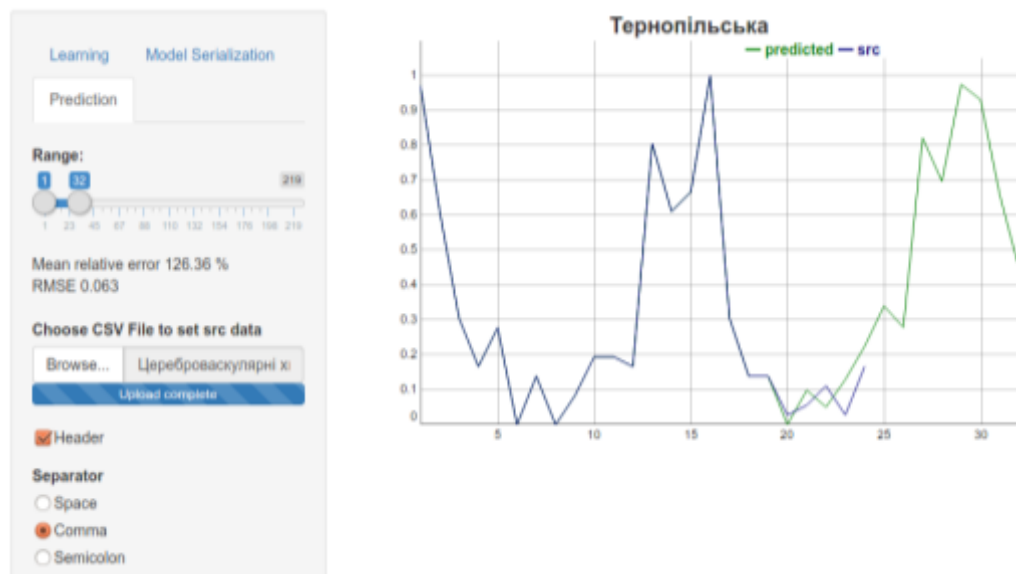
**Рис.7.** Прогнозування “Цереброваскулярні хвороби Закарпатської області”

Для першого прогнозування:  $RMSE = 0,343$ ;  $RME=98,56\%$  - прогноз не є задовільними, загальний прогноз характеру ряду є задовільний. Тут можна побачити явний приклад перенавчання нейронної мережі. Це є наслідком того, що даних для навчання дуже мало, параметри виходу з циклу навчання є великими.



**Рис.7.** Прогнозування “Цереброваскулярні хвороби Рівненської області”

Для першого прогнозування:  $RMSE = 0,016$ ;  $RME=3,81\%$  - прогноз є задовільними, але загальний прогноз характеру ряду не є задовільний, бо маємо близький до паралічу мережі випадок.



**Рис.7.** Прогнозування “Цереброваскулярні хвороби Тернопільської області”

Для першого прогнозування:  $RMSE = 0,028$ ;  $RME=99,34\%$  - прогноз є задовільними, загальний прогноз характеру ряду є задовільний.

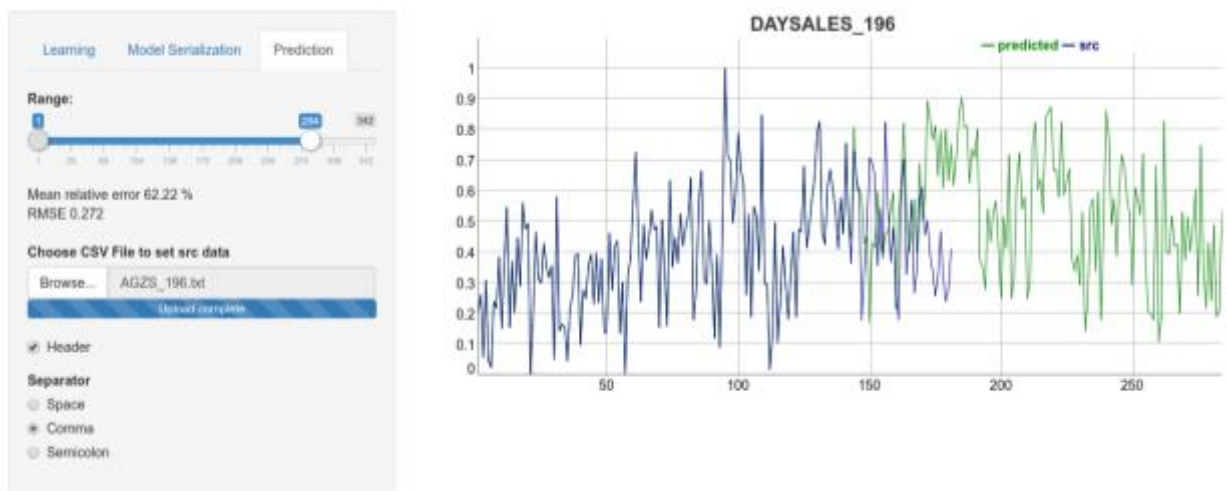
*Прогнозування щоденної кількості проданого товару.*



**Рис.7.** Прогнозування “DAYSALES\_196” без гамма-юнітів

Ця модель не має гамма-юнітів,  $\text{trace\_size} = 22$ ,  $\text{hidden\_layer} = \{12\}$ .

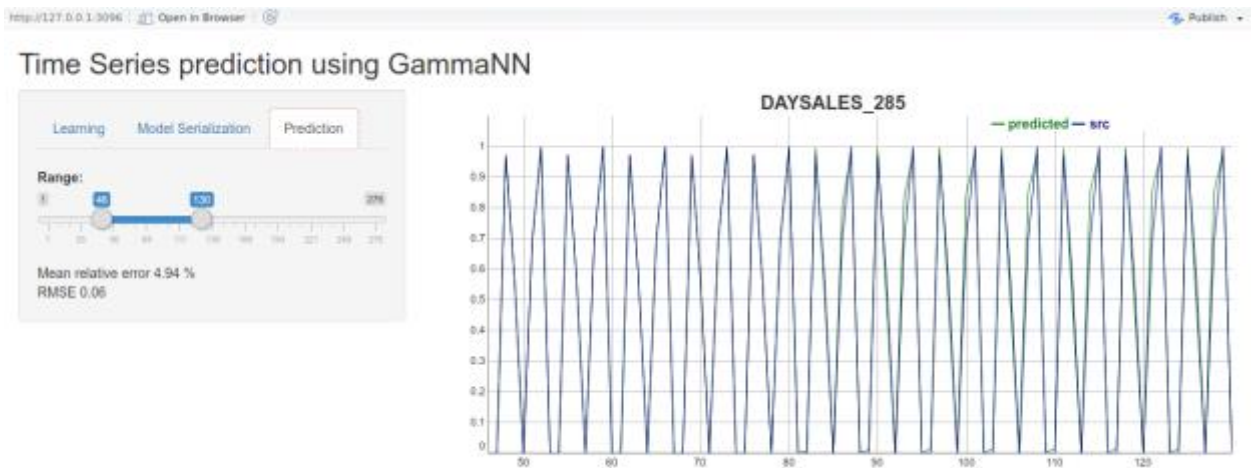
Для першого прогнозування:  $\text{RMSE} = 0,065$ ;  $\text{RME} = 17,93\%$  - прогноз є задовільними, але загальний прогноз характеру ряду не є задовільний.



**Рис.7.** Прогнозування “DAYSALES\_196” з 4 гамма-юнітами

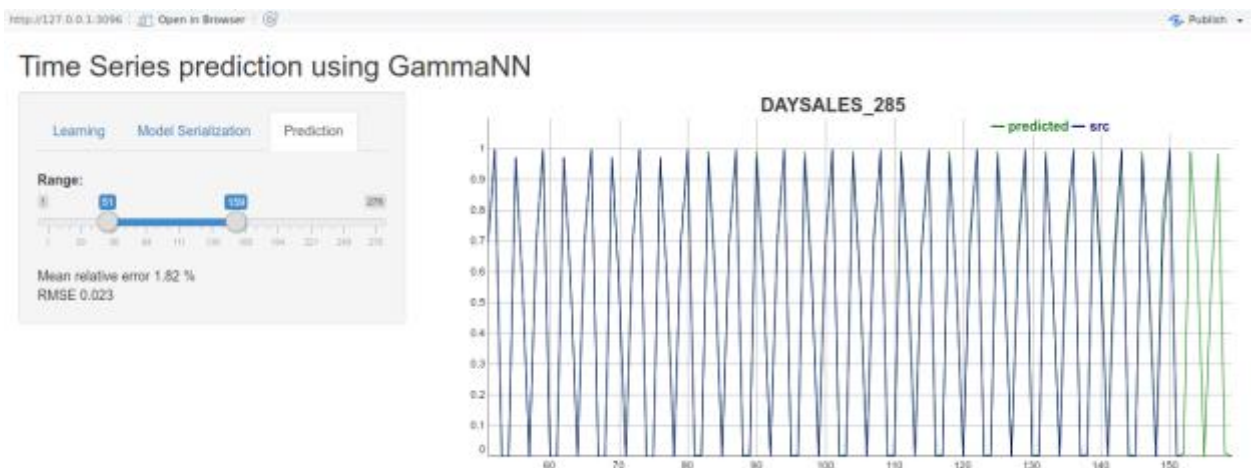
Ця модель має 4 гамма-юніти,  $\text{trace\_size} = 22$ ,  $\text{hidden\_layer} = \{12\}$ .

Для першого прогнозування:  $\text{RMSE} = 0,081$ ;  $\text{RME} = 22,16\%$  - прогноз є задовільними, загальний прогноз характеру ряду є задовільний.



*Рис.7. Прогнозування “DAYSales\_285” без гамма-юнітів*

Для першого прогнозування:  $RMSE = 0$ ;  $RME = 0,01\%$  - прогноз є задовільними, загальний прогноз характеру ряду є задовільний.



*Рис.7. Прогнозування “DAYSales\_196” з 1 гамма-юнітом*

Для першого прогнозування:  $RMSE = 0,003$ ;  $RME=0,49\%$  - прогноз є задовільними, загальний прогноз характеру ряду є задовільний.

## 6. ВИСНОВКИ

З результатів прогнозування рядів “Цереброваскулярні хвороби” можна зробити такі висновки:

- Через невелику кількість даних отримані прогнози неможна розцінювати як вагоме ствердження;
- Здебільшого загальний характер ряду можна відновити і на невеликій кількості даних;

З результатів прогнозування рядів DAYSALLES можна зробити такі висновки:

- Просту періодичну модель як “DAYSALLES\_285” можна відтворити незалежно від того, чи є гамма-юніти чи немає;
- Різницю між помилками першого прогнозування “DAYSALLES\_196” можна вважати вагомою – помилка моделі без гамма-юнітів менша ніж помилка моделі з гамма-юнітами, але загальний прогноз характеру ряду моделі з гамма-юнітами є кращий;

Загальні висновки можна зробити такі:

- Гамма-юніти треба застосовували для відновлення моделей з великою кількістю початкових даних, бо для невеликих масивів даних гамма-юніти ускладнюють модель і призводять до перенавчання моделі.
- Нейронні мережі в поєднанні з гамма-юнітами і без них є хорошим апаратом для відтворення і прогнозування часових рядів.



## 7. СПИСОК ЛІТЕРАТУРИ

1. А.Ю.Лоскутов, А.С.Міхайлов. Основи теорії складних систем. – М. Іжевськ: НДЦ – Регулярна і хаотична динаміка, 2007 – 620 с.
2. С.Хайкін. "Нейронні мережі повний курс", 2-е видання. Видавничий дім "Вільямс", 2006. – 1104 с.
3. Безручко Б.П., Смирнов Д.О. Математичне моделювання та хаотичні тимчасові ряди, Саратов: ГосУНЦ «Коледж», 2005 – 320 с.
4. Г.Г.Малінецький, А.Б.Потапов. Сучасні проблеми нелінійної динаміки. – М. УРСС, 2000 – 360с.
5. А.Ю.Лоскутов, О.Л.Котляров, І.А.Істомін, Д.І.Журавлев. Проблеми нелінійної динаміки. Локальні методи прогнозування тимчасових рядів. – Москва, 2002– 36 с.
6. А.Ю.Лоскутов. Аналіз часових рядів. Курс лекцій., Фізичний факультет МГУ, 113 с.
7. Ніколаєва І.В. Застосування штучних нейронних мереж для прогнозування динаміки економічних показників. Стаття. – Краснодарський філіал РГТЕУ, 2012 – 32 с.