Міністерство освіти і науки України

Харківський національний університет імені В. Н. Каразіна

Факультет комп'ютерних наук

Кафедра теоретичної та прикладної системотехніки

«Затверджую»
Зав. кафедри теоретичної та
прикладної системотехніки
_____ д.т.н., проф. С. І. Шматков
«___» _____ 2021 р.

# Пояснювальна записка

до кваліфікаційної роботи
магістра

на тему: «**Нейромережеві моделі класифікації стану комп'ютеризованої системи»**

**Захищено на засіданні**
**Атестаційної комісії № 39**
протокол № __ від __.12.2021 р.
Оцінка _____ / _____
**Голова Атестаційної комісії**
д.т.н., професор
**_____МІНУХІН**
**Сергій Володимирович**

**Виконав:**

студент 6 курсу, групи КУ– 61
Галузь знань: 15 – Автоматизація та приладобудування
за спеціальністю 151 – Автоматизація та комп'ютерно-інтегровані технології.
**Ткаченко Андрій Максимович_____**

**Керівник:**

к.т.н., доцент кафедри
теоретичної та прикладної
системотехніки
**СТРІЛЕЦЬ Вікторія Євгенівна_____**

**Рецензент:**

к.т.н., доцент електроніки та управляючих систем
**СТЄРВОЄДОВ Микола Григорович_____**

Харків – 2021

# АНОТАЦІЯ

Кваліфікаційна робота складається зі вступу, трьох розділів, висновків до кожного розділу, загальних висновків, переліку використаної літератури та додатків. Обсяг роботи становить 62 сторінки, з яких 51 сторінка основної частини з 14 рисунками та 1 таблицею, 3 сторінки списку використаних джерел із 25 найменувань, 4 додатки на 10 сторінках.

**Мета роботи**: підвищення точності класифікації станів комп'ютеризованої системи.

**Об'єкт дослідження**: процес класифікації станів комп'ютеризованих систем.

**Предмет дослідження:** моделі і методи класифікації станів комп'ютеризованої системи.

В даній кваліфікаційній роботі приведено основні теоретичні відомості зі штучних нейронних мереж, виконано огляд і аналіз існуючих моделей та методів класифікації станів технічних комп'ютеризованих систем, побудовано нейромережеві моделі класифікації станів, виконана їх програмна реалізація. Виконаний порівняльний аналіз створених нейромережевих моделей класифікації; розроблено рекомендації щодо використання моделей класифікації станів технічних комп'ютеризованих систем.

**Ключові слова**: КЛАСИФІКАЦІЯ, МЕРЕЖІ КОХОНЕНА, БАГАТОШАРОВИЙ ПЕРЦЕПТРОН, ЙМОВІРНІСНІ НЕЙРОННІ МЕРЕЖІ, КОМП'ЮТЕРИЗОВАНА СИСТЕМА.

# ABSTRACT

Qualification work consists of an introduction, three sections, conclusions to each section, general conclusions, a list of used literature and applications. The volume of work is 62 pages, of which 51 pages of the main part with 14 drawings and 1 tables, 3 pages of the list of used sources with 25 names, 4 additions on 10 pages.

Purpose of this work: development of neural network models of classification of computerized system states.

Object of study: process of classification of states using neural network methods.

Subject of study: computer model of recognition and classification of computerized system states.

This qualification work provides basic theoretical information, review and analysis of existing models and methods of classification of states of technical computerized systems, constructed neural network models of state classification, performed their software implementation, made a comparative analysis of the considered neural network classification models; developed recommendations for the use of models for classifying the states of technical computerized systems.

KEYWORDS: CLASSIFICATION, NEURAL NETWORKS, COHONEN NETWORKS, MULTI-LAYER PERCEPTRON, PROBABILISTIC NEURAL NETWORKS, NEURAL NETWORK ARCHITECTURE, COMPUTERIZED SYSTEM.

# ЗМІСТ

# ВСТУП

Процес класифікації об’єктів – явище, що зустрічається майже у всіх сферах людської життєдіяльності. Людина навіть не замислюється, що класифікація є невід’ємною частиною пізнання світу. Наприклад, класифікація явищ природи, класифікація виконаних робіт, меблів, продуктів харчування тощо. Якщо розглядати більш масштабно, то можна привести такі загальні приклади використання класифікації: у торгівлі — класифікація клієнтів та товарів дозволяє оптимізувати маркетингові стратегії, стимулювати продажі, скорочувати витрати; у сфері телекомунікацій – класифікація абонентів дозволяє визначати рівень лояльності, розробляти програми лояльності; у медицині та охороні здоров'я – діагностика захворювань, класифікація населення за групами ризику; у банківській сфері — кредитний скоринг.

Моніторинг і класифікація станів технічних систем, до яких відносяться і комп’ютеризовані системи, є невід’ємною складовою експлуатації цих систем. Класифікація - це процес пошуку моделі, яка допомагає розділити дані на різні категоріальні класи. У цьому процесі дані класифікуються під різними мітками відповідно до деяких параметрів, наведених у вводі, і тоді мітки прогнозуються для даних. Категоричне означає, що вихідною змінною є категорія, тобто червоний або чорний, спам або не спам, діабетичний або недіабетичний тощо. Визначення стану системи у будь-який проміжок часу дає можливість вчасно відреагувати на зміни у її роботі та не допустити виникнення відмов.

В даній кваліфікаційній роботі розглядається поняття класифікації, як одного з найбільш популярних напрямків машинного навчання. Описуються моделі та методи розв’язання задачі класифікації. В практичній частині виконується реалізація задачі класифікації стану комп’ютеризованої системи за допомогою трьох архітектур нейронних мереж: мережі Кохонена, багатошарового перцептрону, ймовірнісної нейронної мережі.

*Метою* роботи є підвищення точності класифікації станів комп’ютеризованої системи.

*Об'єкт* дослідження: процес класифікації станів комп’ютеризованих систем.

*Предмет* дослідження: моделі і методи класифікації станів комп’ютеризованої системи.

Для досягнення мети дослідження необхідно вирішити такі *завдання*:

– проаналізувати особливості задачі класифікації як задачі машинного навчання;

– розглянути існуючі методи класифікації станів складних систем;

– розглянути особливості застосування штучних нейронних мереж для розв’язання задачі класифікації;

– обґрунтувати вибір архітектур штучних нейронних мереж для розв’язання задачі класифікації комп’ютеризованої системи;

– розробити програмну реалізацію архітектур штучних нейронних мереж для розв’язання задачі класифікації;

– протестувати створені програмні моделі штучних нейронних мереж на дійсному наборі даних та провести порівняльний аналіз результатів;

– надати рекомендації щодо застосування створених моделей штучних нейронних мереж.

# РОЗДІЛ 1.
# КЛАСИФІКАЦІЯ ЯК ЗАДАЧА МАШИННОГО НАВЧАННЯ

## 1.1 Історія виникнення поняття «класифікація»

Класифікація – поняття, що позначає різновид поділу об’єму, кількості речовини за певними ознаками, на деяку кількість класів, які, в свою чергу, можуть ділитися на підкласи.

Поглянемо на історію виникнення поняття «класифікація».

Ще далекі цивілізації, за багато часу до нас, вміли класифікувати об’єкти. Наприклад:

- класифікація їстівних та неїстівних елементів;
- класифікація надійного та ненадійного житла;
- класифікація явищ природи;
- класифікація ворога за рівнем небезпеки тощо.

Чим більше розвивався світ, тим більше відомостей, пов’язаних з класифікацією з’являлося. Як яскравий приклад можна розглянути періодичну систему Д.І. Менделєєва, що являє собою класифікацію хімічних елементів.

Людина навіть не задумується, скільки разів у повсякденному житті навколо ней розв’язується задача класифікації. Наведемо ще приклади класифікації у повсякденному житті, науці, спорті, медицині та ін.:

- класифікація тварин за середовищем проживання;
- класифікація рослин за типом живлення;
- класифікація документів;
- класифікація хвороб;
- класифікація психологічних захворювань;
- класифікація знаряддя для певних видів спорту;
- класифікація меблів тощо.

Тому, важливість правильного поділу на класи посідає велику роль у житті сьогоднішнього світу, тому потребую не тільки антропогенного втручання, а й автоматизованого.

## 1.2 Аналіз класифікації як задачі машинного навчання

### 1.2.1 Постановка задачі класифікації

Класифікація є однією з найбільш популярних задач машинного навчання.

*Машинне навчання* – поєднання методів аналізу в області штучного інтелекту. По суті, це набір алгоритмів, які застосовують, щоб створити машину, що навчається на своєму досвіді [1].

Для навчання машині необхідно опрацьовувати величезні об’єми вхідних даних та знаходити в них певні спільні риси.

До задач машинного навчання відносяться [2]:

- класифікація;
- кластеризація;
- регресія;
- ранжирування;
- пошук аномалій та ін.

В даній роботі розглядається саме вирішення задачі класифікації.

*Задача класифікації* – задача, в якій є певна кількість об’єктів, поділених на класи, групи, підгрупи. Для задачі класифікації обов’язково задано чітко визначену множину, для якої відомо, до яких класів відносяться її елементи. Така множина називається «*вибіркою*». Щодо об’єктів, які залишились, – невідомо до якого з класів вони відносяться. Тобто, постає задача створення такого алгоритму, що дозволив би класифікувати (розділити на групи) будь-який елемент з початкової множини [3].

*Клас* – це набір, який визначається певною властивістю, і всі об'єкти, що володіють цією властивістю, є елементами цього класу.

Класифікувати дані, означає встановити для кожного об’єкту вибірки приналежність до певної групи чи класу, спираючись на якусь властивість чи якість.

Задача класифікації – прогнозування категорії об’єкта та поділ об’єктів відповідно до заданих ознак. Тобто, відбувається сортування даних по якимось категоріям. Наприклад, книги можна класифікувати за авторами, жанрами, видавництву, мові написання, меблі – за кольором, матеріалом, приналежність до певної кімнати, одяг – за порою року, кольором, тканині, стилю тощо.

Сформулюємо загальну *постановку задачі класифікації*.

Нехай X – множина опису об’єктів, Y – кінцева множина імен чи міток класів. Тоді маємо цільову залежність:

$$y^*: X \to Y, \tag{1}$$

значення якої відомі тільки на об’єктах кінцевої вибірки $x^m = (x_1, y_1), \cdots (x_m, y_m)$. Необхідно побудувати алгоритм $a: X \to Y$, що має змогу класифікувати довільний об’єкт $x \in X$ [4].

### 1.2.2 Моделі класифікації

Розглянемо існуючі моделі класифікації.

1. **Двокласова класифікація (бінарна класифікація).** Найбільш простий у технічному відношенні випадок, який є основою вирішення складніших завдань.

*Бінарна класифікація* – класифікація з категоріальною вихідною змінною, що може приймати лише два значення. Наприклад: 0 та 1, True та False, Yes та No. В задачах бінарної класифікації розглядається можливість приналежності об’єкта до одного з двох класів.

Безліч завдань класифікації в Data Mining (DT) може бути зведено до бінарних. При використанні цього типу класифікації вдається спростити

модель і зняти деякі обмеження, пов'язані з великою кількістю можливих станів вихідної змінної.

Крім цього, бінарні класифікаційні моделі є більш зрозумілими та інтерпретованими.

Бінарний класифікатор формує деяке правило і оцінює, до якого з двох можливих класів слід віднести об'єкт, що вивчається.

Під час тестування моделі можливі помилкові ситуації, такі як віднесення зразку не до того класу. Результати тестування моделі, побудованої за допомогою бінарної класифікації, можна подати у вигляді матриці неточностей (таблиця 1.1).

Таблиця 1.1

**Матриця неточностей**

| | Результати теста | |
|---|---|---|
| Правдиві значення | 1 | 0 |
| 1 | Істинно-позитивні(TP) | Хибно-негативні (FN) |
| 0 | Хибно-позитивні (FP) | Істинно-негативні (TN) |

У цих позначеннях об'єктивна цінність бінарного класифікатора, що розглядається, визначається такими показниками:

A. Чутливість (sensitivity) SE=TP/(TP+FN).

B. Специфіка (specificity) SP=FP/(FP+TN).

C. Точність (accuracy) AC=(TP+TN)/(TP+FP+FN+TN) [5].

**2. Багатокласова класифікація.**

Більшість існуючих методів багатокласової класифікації [6] або базуються на бінарних класифікаторах, або до них зводяться. Загальна ідея такого підходу полягає у використанні набору бінарних класифікаторів, навчених розділяти різні групи об'єктів друг від друга. При класифікації використовують різні схеми голосування. Загалом, усі сучасні багатокласові методи машинного навчання можна розділити на 3 групи:

A. Елементарні – прості підходи, які використовують набір бінарних класифікаторів «один проти всіх» та «кожен проти кожного» [7];

B. Класичні - методи, що зводяться до групи елементарних підходів;

C. Методи, що базуються на кодах з корекцією помилки

**3. Непересічні класи.**

**4. Пересічні класи.** Об'єкт може відноситись одразу до кількох класів.

**5. Нечіткі класи.**

Нечітка класифікація – це процес угруповання елементів в нечітку множину, функція приналежності якого визначається істиною значення нечіткої функції пропозиції.

### 1.2.3 Методи класифікації

До поширених методів розв'язання задачі класифікації відносяться:

- штучні нейронні мережі;
- логістична регресія;
- пробіт-регресія;
- дерева рішень;
- метод найближчого сусіда;
- машини опорних векторів;
- дискримінантний аналіз.

Розглянемо детальніше деякі з вищеприведених методів.

**1. Нейронні мережі.**

Нейронна мережа – структура, що складається зі штучних нейронів, кожен з яких пов’язаний один з одним та навколишнім середовищем за допомогою певних зв’язків. Кожен з цих зв’язків має певний коефіцієнт, на який множиться значення, що надходить через нього.

Нейронні мережі – це моделі, засновані на машинному навчанні. Вони отримують необхідні властивості в процесі навчання, який полягає в

ітеративному налаштуванні ваг мережі за певними правилами. Ці правила мають назву алгоритму навчання.

При побудови нейронних мереж може застосовуватися як навчання з учителем (для багатошарових персептронів), і без вчителя (для мереж Кохонена). Саме ці дві архітектури було більш детально розглянуто для вирішення поставленої задачі [8].

**2. Логістична регресія**

Логістична регресія — це статистична модель, що використовується для передбачення ймовірності виникнення деякої події шляхом підгонки даних до логістичної кривої.

Логістична регресія застосовується для передбачення ймовірності виникнення деякої події за значеннями безлічі ознак. Для цього вводиться так звана залежна змінна y, приймаюча лише одне з двох значень. Як правило, це числа 0 (подія не відбулося) і 1 (подія відбулася), і безліч незалежних змінних (також званих ознаками, предикторами або регресорами) – речових $X_1$, $X_2$, ..., $X_n$, на основі значень яких потрібно обчислити ймовірність прийняття того чи іншого значення залежної змінної [9].

Основна ідея логістичної регресії – простір вихідних значень може бути розділений лінійною межею на дві області, що відповідають класам. У разі двох вимірів лінійною межею виступає пряма лінія без вигинів. У разі трьох – площина, і так далі. Межа задається в залежності від наявних вихідних даних і навчального алгоритму.

Математичне рівняння, що оцінює лінію простої логістичної регресії:

$$Y = a + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n, \quad (2)$$

де $x_n$ незалежна змінна або предиктор;

Y — залежна змінна або змінна відгуку;

a — вільний член лінії оцінки;

$b_n$ — кутовий коефіцієнт або градієнт оціненої лінії;

a і b називають коефіцієнтами регресії оціненої лінії.

Недоліком логістичної регресії є те, що вона застосовується лише до обмеженої кількості вхідних факторів. Іншими словами, на етапі підготовки даних необхідно проводити додаткову обробку даних для виявлення найвпливовіших характеристик і включення в модель лише їх. Можуть також виникнути проблеми з аномальними даними, а також інколи з'являється необхідність відкидання викидів, регуляризації ваг, відкидання ознак, стандартизації даних. Трудомісткість методу вища за звичайний метод найменших квадратів, оцінки ймовірностей можуть виявитися неадекватними, якщо функція правдоподібності не експоненціального вигляду тощо.

Опишемо особливості даного методу. До них відносяться:

1. Визначення для конкретної групуючої ознаки Y, набору ознак-предикторів $X_i$, що пояснюють наборами своїх значень ймовірності віднесення певного спостереження до групи порівняння.

2. Впорядкування відібраних ознак-предикторів Xi за рівнем впливу на залежну ознаку Y.

3. Оцінка надійності приналежності пацієнтів до конкретного класу залежної ознаки Y, за допомогою певної комбінації відібраних ознак-предикторів $X_i$.

4. Можливість оцінки не одного, а багатьох рівнянь логістичної регресії.

5. Вибір різних наборів потенційних предикторів, виходячи з яких алгоритми, що використовуються оцінюють різні рівняння логістичної регресії.

**3. Пробіт-регресія**

Пробіт-модель – це статистична модель бінарного вибору, яка використовується для передбачення ймовірності виникнення події, що цікавить, на основі функції стандартного нормального розподілу.

Модель пробіт-регресії, як і модель логістичної регресії, відносять до моделей бінарного вибору. Тому функції та завдання її побудови аналогічні логіт-моделі [10].

У моделі пробіт-регресії розрахункове значення залежної змінної виражається як значення функції розподілу стандартного нормального закону. Пробіт – це значення, для якого обчислюється функція розподілу стандартного нормального закону розподілу. Значення пробіта залежить від лінійних комбінацій значень факторних змінних. Як і для логіт-моделі залежна змінна у пробіт-моделі є дихотомічною. Фактори в пробіт-моделі повинні бути кількісними змінними або категоріальними, перетвореними на дихотомічні змінні.

Сфери застосування пробіт-моделі такі самі, як і сфери застосування логістичної регресії. Результати моделювання та класифікації за моделлю логістичної регресії та пробіт-моделі в цілому дуже схожі. Однак є свої особливості застосування пробіт-моделей, коли результати можуть бути різними.

**4. Дерева рішень**

Дерева рішень належать до самих популярних і потужних інструментів Data Mining, що дозволяють ефективно вирішувати задачі класифікації. В основі роботи дерев рішень лежить процес рекурсивної розбивки вхідної множини спостережень або об'єктів на підмножини, асоційовані із класами [11].

Алгоритми конструювання дерев рішень складаються з етапів "побудова" або "створення" дерева (tree building) і "скорочення" дерева (tree pruning). У ході створення дерева вирішуються питання вибору критерію розщеплення й зупинки навчання (якщо це передбачене алгоритмом). У ході етапу скорочення дерева вирішується питання відсікання деяких його гілок.

Критерій розщеплення. Процес створення дерева відбувається зверху вниз, тобто є спадним. У ході процесу алгоритм повинен знайти такий критерій розщеплення, іноді також називаний критерієм розбивки, щоб розбити множину на підмножини, які б асоціювалися з даним вузлом перевірки. Кожний вузол перевірки повинен бути позначений певним атрибутом. Існує правило вибору атрибута: він повинен розбивати вхідну

множину даних таким чином, щоб об'єкти підмножин, одержуваних у результаті цієї розбивки, були представниками одного класу або ж були максимально наближені до такої розбивки. Остання фраза означає, що кількість об'єктів з інших класів, так званих "домішок", у кожному класі повинне прагнути до мінімуму.

Зупинка побудови дерева. Розглянемо правило зупинки. Воно повинне визначити, чи є розглянутий вузол внутрішнім вузлом, при цьому він буде розбиватися далі, або ж він є кінцевим вузлом, тобто вузлом розв'язком. Зупинка – такий момент у процесі побудови дерева, коли слід припинити подальші розгалуження. Один з варіантів правил зупинки - "рання зупинка" (prepruning), вона визначає доцільність розбивки вузла. Перевага використання такого варіанта – зменшення часу на навчання моделі. Однак тут виникає ризик зниження точності класифікації. Тому рекомендується " замість зупинки використовувати відсікання" (Breiman, 1984).

Другий варіант зупинки навчання – обмеження глибини дерева. У цьому випадку побудова закінчується, якщо досягнута задана глибина. Ще один варіант зупинки - завдання мінімальної кількості прикладів, які будуть утримуватися в кінцевих вузлах дерева. При цьому варіанті розгалуження тривають до того моменту, поки всі кінцеві вузли дерева не будуть чистими або будуть містити не більш ніж задане число об'єктів. Скорочення дерева або відсікання гілок. Розв'язком проблеми занадто гіллястого дерева є його скорочення шляхом відсікання (pruning) деяких гілок. Якість класифікаційної моделі, побудованої за допомогою дерева рішень, характеризується двома основними ознаками: точністю розпізнавання й помилкою.

Відсікання гілок або заміну деяких гілок піддеревом слід проводити там, де ця процедура не приводить до зростання помилки. Процес проходить знизу нагору, тобто є висхідним. Це більш популярна процедура, чим використання правил зупинки. Дерева, одержувані після відсікання деяких гілок, називають усіченими. Якщо таке усічене дерево усе ще не є інтуїтивним і складно для розуміння, використовують добування правил, які поєднують у набори для

опису класів. Кожний шлях від кореня дерева до його вершини або листа дає одне правило. Умовами правила є перевірки на внутрішніх вузлах дерева.

**5. Машина опорних векторів**

Метод опорних векторів (SVM, support vector machine) – набір схожих алгоритмів навчання з учителем, що використовуються для завдань класифікації та регресійного аналізу. Належить сімейству лінійних класифікаторів і може розглядатися як окремий випадок регуляризації по Тихонову. Особливою властивістю методу опорних векторів є безперервне зменшення емпіричної помилки класифікації та збільшення зазору, тому метод також відомий метод класифікатора з максимальним зазором.

Основна ідея методу – переведення вихідних векторів у простір більш високої розмірності та пошук роздільної гіперплощини з найбільшим зазором у цьому просторі. Дві паралельні гіперплощини будуються по обидва боки гіперплощини, що розділяє класи. Розділяючою гіперплощиною буде гіперплощина, що створює найбільшу відстань до двох паралельних гіперплощин. Алгоритм заснований на припущенні, що чим більша різниця чи відстань між цими паралельними гіперплощинами, тим меншою буде середня помилка класифікатора.

Опишемо постановку задачі класифікації даних. Кожен об'єкт даних представляється як вектор (точка) в p-мірному просторі (упорядкований набір p чисел). Кожна з цих точок належить лише одному із двох класів. Питання полягає в тому, чи можна розділити точки гіперплощиною розмірності (p−1). Це типовий випадок лінійної роздільності. Шуканих гіперплощин може бути багато, тому вважають, що максимізація зазору між класами сприяє більш впевненій класифікації. Тобто, можна знайти таку гіперплощину, щоб відстань від неї до найближчої точки була максимальною. Це еквівалентно тому, що сума відстаней до гіперплощини від двох найближчих до неї точок, що лежать по різні боки від неї, є максимальною. Якщо така гіперплощина існує, вона називається оптимальною розділяючою гіперплощиною, а відповідний їй

лінійний класифікатор називається оптимально розділяючим класифікатором [12].

**6. Дискримінантний аналіз**

Дискримінантний аналіз – метод багатовимірного статистичного аналізу. Він включає методи класифікації багатовимірних спостережень за принципом максимальної подібності за наявності навчальних ознак. На відміну від кластерного аналізу, нові кластери не утворюються, а є правилом, за яким об'єкти відносяться до певної групи. Завдання дискримінантного аналізу багато в чому схожі на завдання логістичної регресії – класифікація спостережень на групи на основі прогностичної моделі. Незважаючи на деякі подібності дискримінантний аналіз і логістична регресія мають істотні відмінності. Ідеї дискримінантного аналізу тісно пов'язані з дисперсійним, регресійним аналізом.

Сенс дискримінантного аналізу – виходячи з навчальних вибірок перетворити багатовимірний масив на одновимірний показник для прогнозування належності спостережень до груп, т. е. побудувати новий узагальнений показник, значення якого максимально різняться для об'єктів, віднесених до різних груп. Навчальна вибірка – це безліч об'єктів, заданих значеннями ознак та належність яких до того чи іншого класу достовірно відома [13].

**Висновки до першого розділу**

В даному розділі було наведено основні теоретичні відомості відносно поняття «класифікація». Розглянуто історичне становлення явища класифікації, моделі класифікації, а також методи, за допомогою яких можна розв’язати задачу класифікації.

Клас – це набір, який визначається певною властивістю, і всі об'єкти, що володіють цією властивістю, є елементами цього класу.

Класифікувати дані, означає встановити для кожного об’єкту вибірки приналежність до певної групи чи класу, спираючись на якусь властивість чи якість.

Задача класифікації – прогнозування категорію об’єкта та поділ об’єктів відповідно до заданих ознак. Тобто, відбувається сортування даних по якимось категоріям.

Були розглянуті та проаналізовані найбільш використовувані методи класифікації, такі як:

– нейронні мережі;
– логістична регресія;
– пробіт-регресія;
– дерева рішень;
– машини опорних векторів;
– дискримінантний аналіз.

Логістична регресія — це статистична модель, що використовується для передбачення ймовірності виникнення деякої події шляхом підгонки даних до логістичної кривої.

Дерева рішень належать до самих популярних і потужних інструментів Data Mining, що дозволяють ефективно вирішувати задачі класифікації.

Метод опорних векторів – набір схожих алгоритмів навчання з учителем, що використовуються для завдань класифікації та регресійного аналізу. Належить сімейству лінійних класифікаторів і може розглядатися як окремий випадок регуляризації по Тихонову.

Для розв’язання поставленої задачі було обрано реалізацію задачі класифікації за допомогою нейронних мереж. В наступному розділі розглянуто три архітектури нейронних мереж, за допомогою яких буде розв’язано поставлену задачу. До цих архітектур відносяться: мережі Кохонена, ймовірнісні штучні мережі, багатошаровий перцептрон.

Дискримінантний аналіз – метод багатовимірного статистичного аналізу. Він включає методи класифікації багатовимірних спостережень за

принципом максимальної подібності за наявності навчальних ознак. На відміну від кластерного аналізу, нові кластери не утворюються, а є правилом, за яким об'єкти відносяться до певної групи.

Нейронна мережа – структура, що складається зі штучних нейронів, кожен з яких пов’язаний один з одним та навколишнім середовищем за допомогою певних зв’язків. Кожен з цих зв’язків має певний коефіцієнт, на який множиться значення, що надходить через нього.

# РОЗДІЛ 2.
# НЕЙРОМЕРЕЖЕВІ МЕТОДИ РОЗВ'ЯЗАННЯ ЗАДАЧІ КЛАСИФІКАЦІЇ

## 2.1 Мережі Кохонена

Нейронні мережі Кохонена – нейромережева архітектура, що навчається без вчителя.

За допомогою нейронних мереж Кохонена можна розв'язувати задачі класифікації, кластеризації або прогнозування. Крім того, мережі Кохонена можуть використовуватися з метою зменшення розмірності даних із мінімальною втратою інформації.

В архітектурі, що розглядається, сигнал поширюється від входів до виходів у прямому напрямку. Структура нейронної мережі містить єдиний шар нейронів (шар Кохонена) без коефіцієнтів зміщення (рис. 2.1) [14].

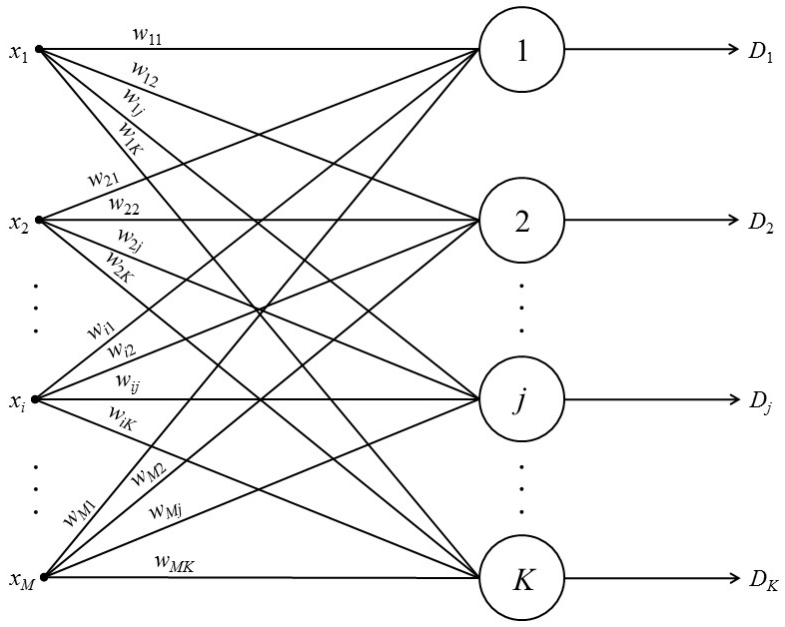


Рисунок 2.1 – Структура мережі Кохонена

Загальна кількість вагових коефіцієнтів розраховується як добуток (3):

$$N_w = MK, \tag{3}$$

де К – кількість нейронів  шарі Кохонена.

Нормалізація вхідних змінних виконується на інтервалі [-1;1].

*Алгоритм навчання мережі Кохонена.*

1. Задати структуру мережі (вказати кількість нейронів в мережі Кохонена).

2. Випадковим чином ініціалізувати ваги, що задовольняють наступним правилам:

а) при нормалізації початкової вибірки у межах [-1;1] (4):

$$\left|W_{ij}\right| \leq \frac{1}{\sqrt{M}}, \tag{4}$$

б) при нормалізації початкової вибірки у межах [0;1] (5):

$$0{,}5 - \frac{1}{\sqrt{M}} \leq W_{ij} \leq 0{,}5 + \frac{1}{\sqrt{M}}, \tag{5}$$

де М – кількість вхідних змінних мережі – характеристичних ознак об’єкта дослідження.

3. Подача на входи мережі випадкового прикладу для навчання епохи навчання, що триває в цей момент, а також розрахунок евклідових відстаней від вхідного вектору до центрів усіх кластерів (6):

$$R_j = \sqrt{\sum_{i=1}^{m} \left(\tilde{x}_i - w_{ij}\right)^2} \tag{6}$$

4. За найменшим зі значень $R_j$ обирається нейрон-переможець j, в найбільшій мірі близький по значенням з вхідним вектором. Для обраного нейрона, і не тільки для нього, виконується корекція вагових коефіцієнтів (7):

$$w_{ij}^{(q+1)} = w_{ij}^{(q)} + v\left(\tilde{x}_i - w_{ij}^{(q)}\right), \quad (7)$$

де V – коефіцієнт швидкості навчання.

5. Цикл повторюється з кроку 3 до виконання одного або декількох умов завершення:

– вичерпано кількість епох навчання;

– не сталося значної зміни вагових коефіцієнтів в межах заданої точності протягом останньої епохи навчання;

– вичерпано заданий час навчання.

Коефіцієнт швидкості навчання може задаватися постійним з інтервалу (0,1] або змінним значенням, що постійно зменшується від епохи до епохи.

## 2.2 Багтошаровий перцептрон

*Багатошаровий персептрон* – це клас штучних нейронних мереж прямого поширення, що складаються як мінімум із трьох шарів: вхідного, прихованого та вихідного. Крім вхідних, всі нейрони використовують нелінійну функцію активації [15].

Вперше багатошаровий персептрон було запропоновано Ф. Розенлаттом. Однак у тому вигляді, в якому він використовується в даний час, багатошаровий персептрон розроблено Д. Румельхартом.

*Багатошаровий перцептрон Румельхарта* – окремий випадок перцептрону Розенблатта, в якому один алгоритм зворотного поширення помилки навчає всі шари.

Особливістю є наявність більш ніж одного шару, що навчається (як правило два або три, для застосування більшого числа на даний момент немає обґрунтування, втрачається швидкість без придбання якості). Необхідність у великій кількості шарів, що навчаються, відпадає, так як теоретично єдиного прихованого шару достатньо, щоб перекодувати вхідне представлення таким

чином, щоб отримати лінійну карту для вихідного представлення. Але є припущення, що використовуючи більше шарів можна зменшити кількість елементів у них, тобто. сумарна кількість елементів у шарах буде меншою, ніж якщо використовувати один прихований шар [17].

Персептрон Румельхарта відрізняється від персептрона Розенблатта за такими властивостями:

– використання нелінійної активаційної функції;

– число прихованих шарів більше одного (зазвичай трохи більше трьох);

– вхідні сигнали не бінарні, а кодуються десятковими числами, нормованими до інтервалу [0,1];

– вихідна помилка мережі визначається не як число помилково розпізнаних прикладів, а як деяке значення нев’язки;

– навчання проводиться не до мінімізації помилки, а до стабілізації ваги мережі, що дозволяє уникнути перенавчання.

Кількість вхідних та вихідних елементів у багатошаровому персептроні визначається умовами задачі. Сумніви можуть виникнути, які вхідні значення використовувати, а які ні. Питання про те, скільки використовувати проміжних шарів та елементів у них, поки що зовсім. Як початкове наближення можна взяти один проміжний шар, а число елементів у ньому покласти рівним напівсумі числа вхідних і вихідних елементів. Загальна структура перцептрону представлена нижче (рис. 2.2).

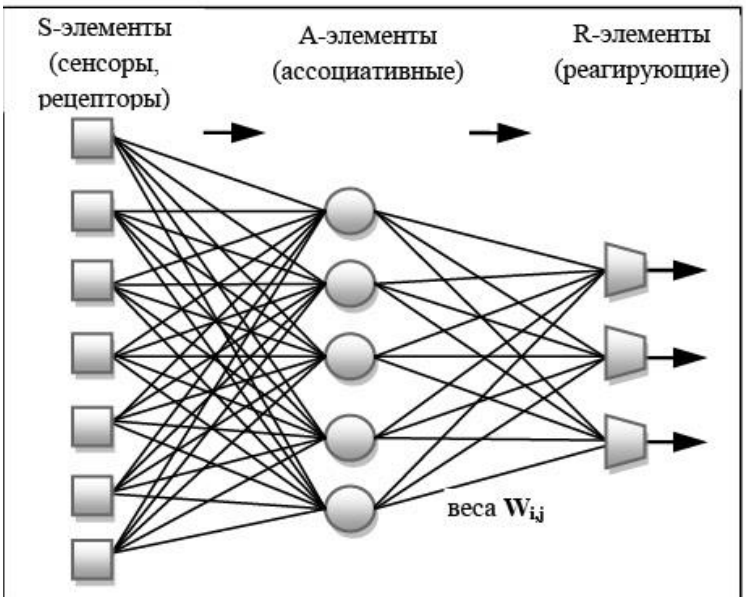


Рисунок 2.2. – Структура перцептрону

## 2.3 Ймовірнісні нейронні мережі

Імовірнісна нейронна мережа (PNN) – це нейронна мережа прямого поширення, яка широко використовується в задачах класифікації та розпізнавання образів.

В алгоритмі PNN батьківська функція розподілу ймовірностей кожного класу апроксимується вікном Парзена та непараметричною функцією. Потім, використовуючи батьківська функція розподілу ймовірностей кожного класу, оцінюється ймовірність класу нових вхідних даних, а потім застосовується правило Байєса виділення класу з найвищою апостеріорною ймовірністю новим вхідних даних. За допомогою цього методу ймовірність помилкової класифікації зводиться до мінімуму [18].

Ймовірнісна нейронна мережа складається з чотирьох шарів: вхідний шар, прихований шар з нейронами, що відповідають навчальним прикладам, підсумовуючий шар, що одержує висновки від нейронів прихованого шару

навчальних прикладів відповідного класу, та вихідний шар, на якому проводиться порівняння ймовірностей та виводиться результат (рис. 2.3).

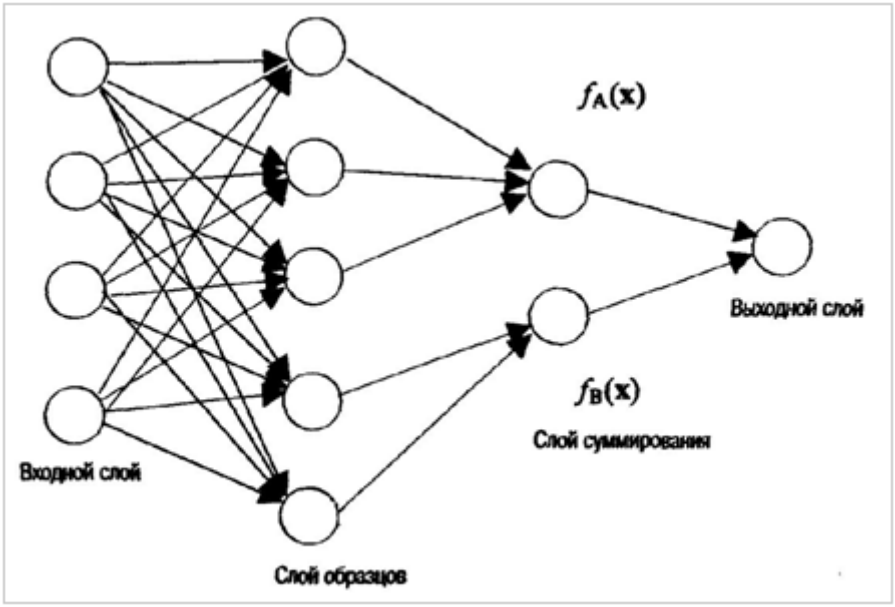


Рисунок 2.3 – Структура ймовірнісної нейронної мережі

Імовірнісні нейронні мережі широко застосовуються в задачах класифікації та розпізнавання, таких як різні завдання у медицині, діагностику в медицині, розпізнавання голосу, визначення особи по голосу, райдужній оболонці ока, зображенню обличчя, розпізнавання, обробка та класифікація зображень тощо.

У ряді робіт також зазначається, що імовірнісні нейронні мережі вигідно відрізняються при вирішенні класифікаційних завдань, здатністю працювати з нелінійними, складно організованими системами, стійкості до шумів, високої швидкості навчання та легкості модифікації для включення нових даних, що є важливою перевагою при роботі з базами, що постійно доповнюються.

Використання PNN замість багатошарового персептрона має кілька переваг та недоліків.

До переваг відносяться:

– PNN набагато швидше, ніж багатошарові мережі персептронів.

– PNN може бути більш точнішим, ніж багатошарові мережі персептронів.

– Мережі PNN відносно нечутливі до викидів.

– Мережі PNN генерують точні прогнозовані цільові імовірнісні оцінки.

– PNN наближаються до оптимальної класифікації.

До недоліків відносяться:

– PNN повільніша, ніж багатошарові мережі персептронів при класифікації нових випадків.

– PNN потребує більшого обсягу пам'яті для зберігання моделі.

## 2.4 Алгоритми навчання нейронних мереж

Штучні нейронні мережі відносять до методів машинного навчання, які «навчаються з вчителем». Навчання з учителем базується на використанні послідовності навчальних вибірок, що задають бажані значення вихідних векторів нейронної мережі для відповідних значень вхідних векторів. При навчанні без учителя замість навчальних вибірок використовується попередній досвід функціонування нейронної мережі. Але як у першому, так і у другому випадку єдиним залишається механізм навчання, який полягає в модифікації вагових коефіцієнтів нейронів.

Основним критерієм оцінки ефективності навчання є цільова функція, що дає функціональний опис того, наскільки модельний вихід співпадає з ідеальним для даної вхідної вибірки. Цільову функцію для методів навчання з учителем ще називають похибкою вихідного вектору нейронної мережі [16].

### 2.4.1 Метод градієнтного спуску

Ідея методу градієнтного спуску полягає в послідовній зміні параметрів ШНМ в напрямку, що забезпечує зменшення цільової функції $E$. Оскільки функція $E$ диференційовна по кожному з параметрів, то існує можливість

обчислення градієнтного вектора. Рухаючись у напрямку від'ємного градієнта по кожному з параметрів, знаходимо локальні мінімуми цільової функції. На рис.2.4 зображений принцип зміни параметра за методом градієнтного спуску.

Така зміна виражається формулою:

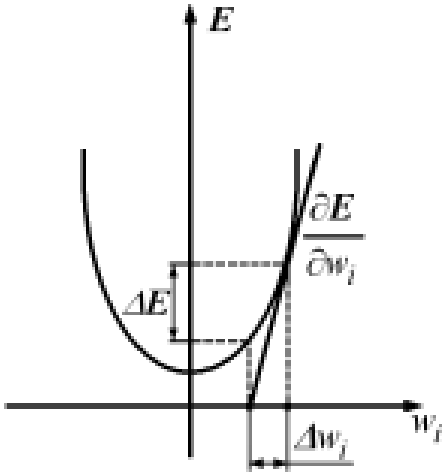


Рисунок 2.4 – Зміна параметра за методом градієнтного спуску

На рис. 2.5 зображено блок-схему алгоритму, що побудований на базі методу градієнтного спуску. Цей алгоритм називають алгоритмом пакетного типу, тому що для визначення величини кроку зміни параметра необхідно провести обробку всієї навчальної вибірки.

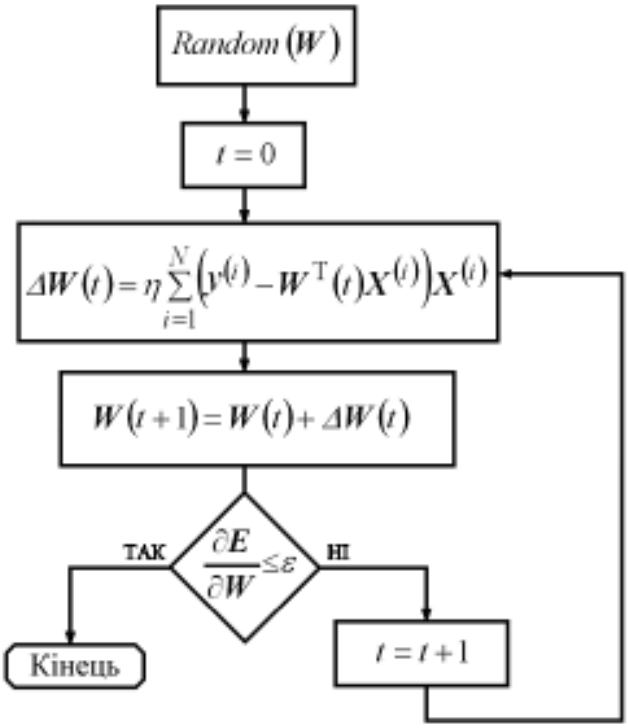


Рисунок 2.5 – Алгоритм градієнтного спуску

Практика показує високу стійкість даного алгоритму, що забезпечує незначні коливання модуля цільової функції за умови зміни характеру вхідних даних. Але невелика швидкість збіжності обумовлює необхідність значних затрат часу на навчання ШНМ. Основним параметром, що впливає на швидкість збіжності алгоритму градієнтного спуску, є коефіцієнт навчання η. Відповідно до методу, модифікація цього параметра може бути проведена тільки після обробки всієї вибірки даних [16].

### 2.4.2 Метод найменших квадратів

Метод найменших квадратів (Last Mean Square) розроблений з метою покращення швидкості збіжності алгоритму шляхом підвищення динаміки регулювання коефіцієнта η . Він був уперше застосований для нейронної мережі Adaline з лінійною активаційною функцією нейронів, а потім узагальнений для довільних ШНМ прямого поширення. Динамічний характер методу полягає в тому, що процес корекції кроку параметра може відбуватися не після обробки всієї вибірки, як у методі градієнтного спуску, а після

обчислення кожного елемента вибірки. На рис. 2.6 показана блок-схема алгоритму найменших квадратів, який має необмежену навчальну вибірку.

Такий алгоритм називають алгоритмом реального часу. Критерієм закінчення алгоритму навчання є досягнення мінімуму цільової функції. Швидкість збіжності алгоритму критично залежить від величини кроку параметра, який регулюється коефіцієнтом навчання після кожної ітерації.

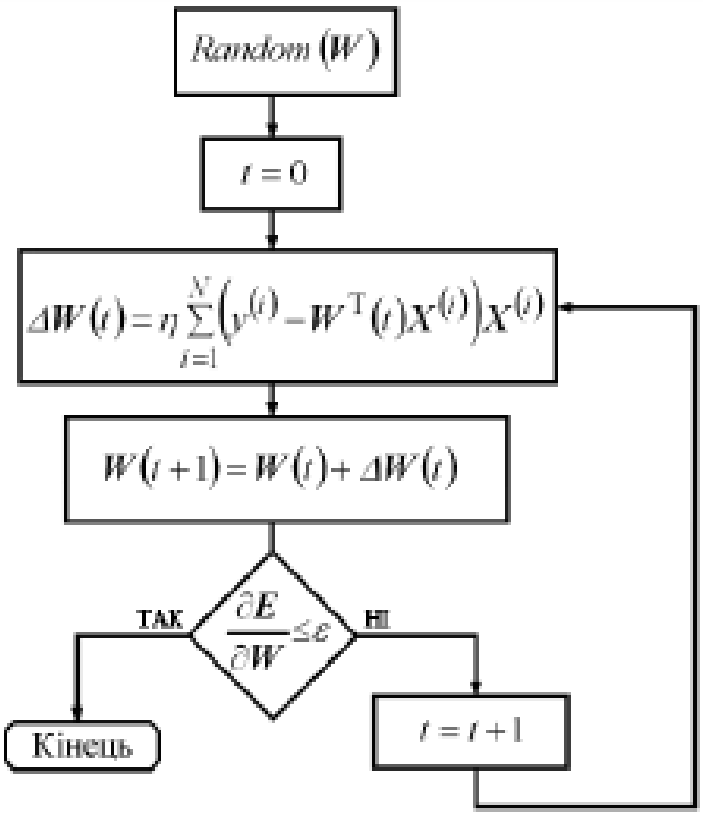


Рисунок 2.6 – Алгоритм методу найменших квадратів

### 2.4.3 Метод Гауса-Ньютона

Ідея розглянутих методів навчання ШНМ полягає у використанні похідної першого порядку для мінімізації цільової функції. Однак більш точні результати можуть бути досягнуті у випадку представлення цільової функції у вигляді ряду Тейлора з урахуванням члена цього ряду, що містить похідну другого порядку. Такий метод має назву методу Гауса–Ньютона і забезпечує мінімізацію цільової функції

$$E(w+\Delta w) = E(w) + \left(\frac{\partial E}{\partial w}\right)^T \Delta w + \frac{1}{2}\Delta w^T \frac{\partial^2 E}{\partial w^2}\Delta w + O(\|\Delta w\|^3) \approx$$

$$\approx E(w) + \left(\frac{\partial E}{\partial w}\right)^T \Delta w + \frac{1}{2}\Delta w^T H \Delta w \ , \quad (8)$$

де $w = [w_1, w_2 w_3, \dots, w_n]^T$ – вектор параметрів ШНМ, що містить вагові коефіцієнти прихованого та вихідного шарів;

$\frac{\partial E}{\partial w} = \left[\frac{\partial E}{\partial w_1}\frac{\partial E}{\partial w_2}\frac{\partial E}{\partial w_3}\dots\frac{\partial E}{\partial N_n}\right]^T$ - вектор частинних похідних; Δw – вектор приростів параметрів; H – матриця Гессе похідних другого порядку; $H = \frac{\partial^2 E}{\partial w^2}$

Мінімальне значення цільової функції одержують шляхом прирівнювання похідної ряду Тейлора до нуля (9):

$$\frac{\partial E(w+\Delta w)}{\partial w} \approx \frac{\partial E(w)}{\partial w} + H\Delta w = 0, \quad (9)$$

З цієї формули випливає, що оптимальний вектор приростів параметрів (10):

$$\Delta w = -H^{-1}\frac{\partial E}{\partial w}, \quad (10)$$

Практичне застосування даного методу визначення оптимальних кроків для приросту параметрів стикається з проблемами, пов’язаними з інвертуванням матриці H . Кількість обчислень, необхідних для виконання даної операції для матриці з розмірністю (n × n), зростає пропорційно до 3 n . Існують також обмеження на вигляд матриці H . Для того, щоб гарантовано існував мінімум цільової функції, необхідно, щоб матриця H була позитивно означеною, тобто всі її власні значення були додатними. У випадку нульових власних значень маємо сідлову точку, в якій H не може бути обернена. Наявність від’ємних власних значень свідчить про те, що існує хоча б один максимальний ваговий коефіцієнт, тобто одержане значення цільової функції не відповідає умовам мінімуму [16].

### 2.4.4 Метод Левенберга-Маркара

Обмеження на вигляд матриці H, що характерні для методу Гауса–Ньютона, усувають шляхом регуляризації. Для цього вводять заміну з метою фільтрації власних значень, які менші, ніж λ (11):

$$H \rightarrow H + \lambda I \tag{11}$$

Підставивши в елемент ij h матриці вище значення цільової функції, яка задана середньоквадратичною похибкою, одержимо (12):

$$h_{ij} = \frac{1}{2}\frac{\partial^2 E}{\partial w_i \partial w_j} = \frac{1}{N}\sum_{n=1} \frac{\partial \tilde{y}^{(n)}}{\partial w_i}\frac{\partial \tilde{y}^{(n)}}{\partial w_j}, \tag{12}$$

При достатньо малих значеннях $(y^n - \tilde{y}^n)$ другий член формули можна не враховувати в подальших обчисленнях. Тому було зроблено спрощення (13):

$$h_{ij} \approx \frac{1}{N}\sum_{n=q}^{v} \frac{\partial_y^n}{\partial w_i}\frac{\partial_y^2}{\partial w_j} \ , \tag{13}$$

яке дає можливість представити матрицю H у вигляді (14):

$$H \approx \frac{1}{N}\sum_{n=1}^{e} J^n (J^n)^T \tag{14}$$

Операції з якобіаном (J) потребують значно менших обчислювальних ресурсів, що пов’язано з використанням частинних похідних першого порядку. Використовуючи згаданий підхід до регуляризації матриці, яка підлягає інвертуванню, та ввівши спрощення, одержують формулу Левенберга–Маркара для формування вектора приросту параметрів (15):

$$\Delta w = \left[\frac{1}{v}\sum_{n=1}^{w} J^n (J^n)^T + \lambda I\right]\frac{\partial E}{\partial w} \tag{15}$$

Метод Левенберга–Маркара є узагальненням методів градієнтного спуску та Гауса–Ньютона.

### 2.4.5 Метод навчання багатошарових персептронів

Багатошарові ШНМ можуть розглядатися як послідовне з'єднання одношарових ШНМ прямого поширення. Структуру вагових коефіцієнтів у цих мережах організовують таким чином, щоб класи більш складної форми оброблялися на шарах нейронів високих рівнів шляхом об'єднання та перетину простих класів, які формуються на нижчих рівнях ШНМ. Відомі строгі доведення того факту, що двошарова ШНМ здатна розпізнати довільний клас опуклої форми за умови, що існує можливість використати достатню кількість нейронів прихованого шару, а вагові коефіцієнти налаштовані належним чином. ШНМ прямого поширення з кількома прихованими шарами потенційно здатна до розпізнавання класів довільної форми.

Отже, постановка задачі на ШНМ прямого поширення включає визначення мінімально можливої кількості нейронів прихованого шару та вибір ефективного методу настройки вагових коефіцієнтів. На сьогодні обидві ці проблеми не є тривіальними. Для пояснення основних принципів побудови методів навчання з учителем будемо розглядають двошарову ШНМ (рис. 2.7).

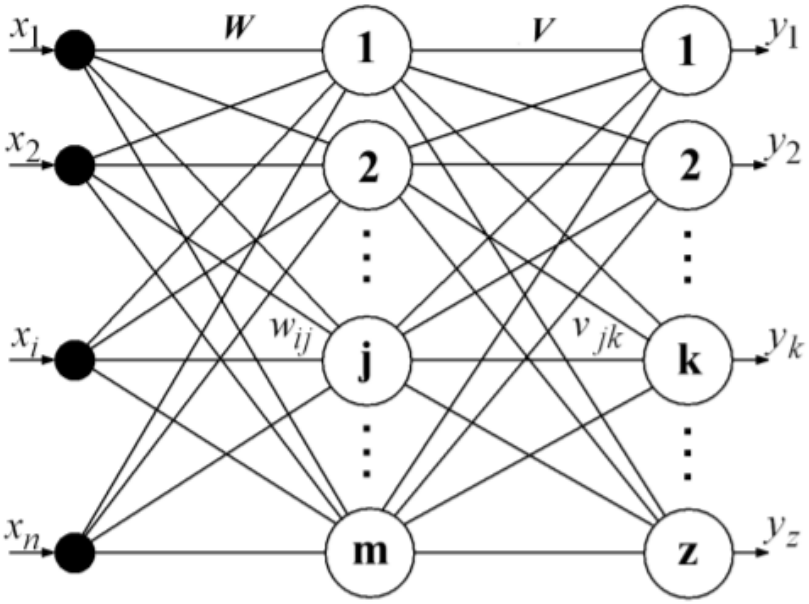


Рисунок 2.7 - Узагальнена багатошарова ШНМ прямого поширення

Нульовий шар цієї мережі виконує допоміжну функцію розгалуження сигналів і не містить нейронів. З цієї причини його робота не приводить до модифікації вхідного вектора. Останній шар ШНМ називають вихідним шаром. Всі шари, що розташовані між нульовим та вихідним, є прихованими шарами з нелінійною активаційною[16].

Роботу даної ШНМ можна описати такими рівняннями (16-19):

$$\tilde{y} = \varphi[s(x)], \tag{16}$$

$$S(x) = v_0 + \sum_{j=1}^{m} v_j h_j, \tag{17}$$

$$h_j = f(t_j) = f\left(w_{Oj} + \sum_{i=1}^{n} w_{i_j} x_i\right), \tag{18}$$

$$t_j = w_{0j} + \sum_{i=1}^{n} w_{ij}\, x_i, \tag{19}$$

де i, j — індекси елемента вхідного вектора та нейрона прихованого шару відповідно; $x_i$ — елемент вхідного вектора; $t_j$ — аргумент активаційної функції нейрона прихованого шару; $h_j$ — елемент вихідного вектора

прихованого шару; s — аргумент активаційної функції нейрона вихідного шару; y — вихідний вектор ШНМ; f(t) — активаційна функція нейронів прихованого шару; ϕ(s)— активаційна функція нейронів вихідного шару.

**Висновки до другого розділу**

В другому розділі було описано три архітектури нейронних мереж, які необхідно побудувати, для реалізації поставленої мети. До цих архітектур відносяться:

– мережі Кохонена;

– ймовірнісні нейронні мережі;

– багатошаровий перцептрон.

Імовірнісна нейронна мережа (PNN) – це нейронна мережа прямого поширення, яка широко використовується в задачах класифікації та розпізнавання образів. Вона складається з чотирьох шарів: вхідний шар, прихований шар з нейронами, що відповідають навчальним прикладам, підсумовуючий шар, що одержує висновки від нейронів прихованого шару навчальних прикладів відповідного класу, та вихідний шар, на якому проводиться порівняння ймовірностей та виводиться результат.

Багатошаровий персептрон – це клас штучних нейронних мереж прямого поширення, що складаються як мінімум із трьох шарів: вхідного, прихованого та вихідного. Крім вхідних, всі нейрони використовують нелінійну функцію активації

Нейронні мережі Кохонена – нейромережева архітектура, що навчається без вчителя. Структура нейронної мережі містить єдиний шар нейронів (шар Кохонена) без коефіцієнтів зміщення.

Під час аналізу теоретичної частини було встановлено, що кожна з наведених архітектур має як свої переваги, так і недоліки одна відносно одної.

Також, в цьому розділі розглянуто різні алгоритми навчання нейронних мереж:

- метод градієнтного спуску;

- метод Левенберга-Маркара;
- метод навчання багатошарових персептронів;
- метод найменших квадратів;
- метод Гауса-Ньютона.

# РОЗДІЛ 3.
# ПРОГРАМНА РЕАЛІЗАЦІЯ МОДЕЛЕЙ КЛАСИФІКАЦІЇ

## 3.1 Вибір середовища реалізації

Для реалізації заданих архітектур нейронних мереж було розглянуто дві мови (та два програмних середовища) – Python(PyCharm), R(RStudio).

**Python** – багатоцільова мова програмування, яка дозволяє писати простий та зрозумілий код. Простота цієї мови дозволяє створювати коротші програми, ніж на інших мовах. Але це не означає те, що програми, написані на мові програмування Python, можуть бути тільки невеликими скриптами, вони можуть буди також і складними системами [22].

Python добре підходить для машинного навчання, адже при роботі з Python розробнику не потрібно приділяти багато уваги написанню коду.

**PyCharm** – інтегроване середовище розробки для мови програмування Python. Воно надає засоби для аналізу коду, графічний відладчик, інструмент для запуску юніт-тестів і підтримує веб-розробку на Django (рис. 3.1).


Рисунок 3.1– Інтерфейс PyCharm

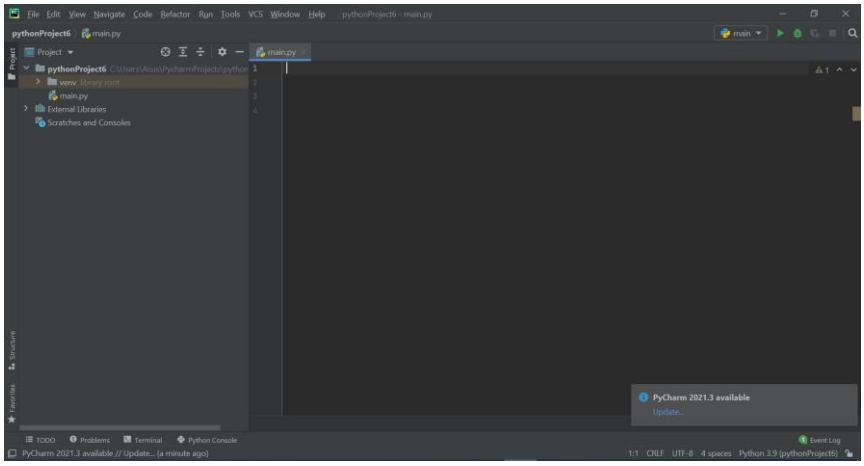

Рисунок 3.1 (продовження) – Інтерфейс PyCharm

PyCharm надає розумну перевірку коду, швидке виявлення помилок і оперативне виправлення, укупі з автоматичним рефакторингом коду, і багатими можливостями в навігації.

PyCharm надає спеціальну підтримку наукових бібліотек. Він підтримує Anaconda, Numpy, Matplotlib і інші наукові бібліотеки, надаючи користувачу глибоке розуміння коду, інтерактивні графіки, перегляд масивів і багато іншого [23,24].

**R** – вільно поширювана система, яка є найбільш повною, надійною і є статистичним середовищем, що динамічно розвивається (рис. 3.2). R об'єднує мову програмування високого рівня і потужні бібліотеки програмних модулів для обчислювальної та графічної обробки даних [23,24].

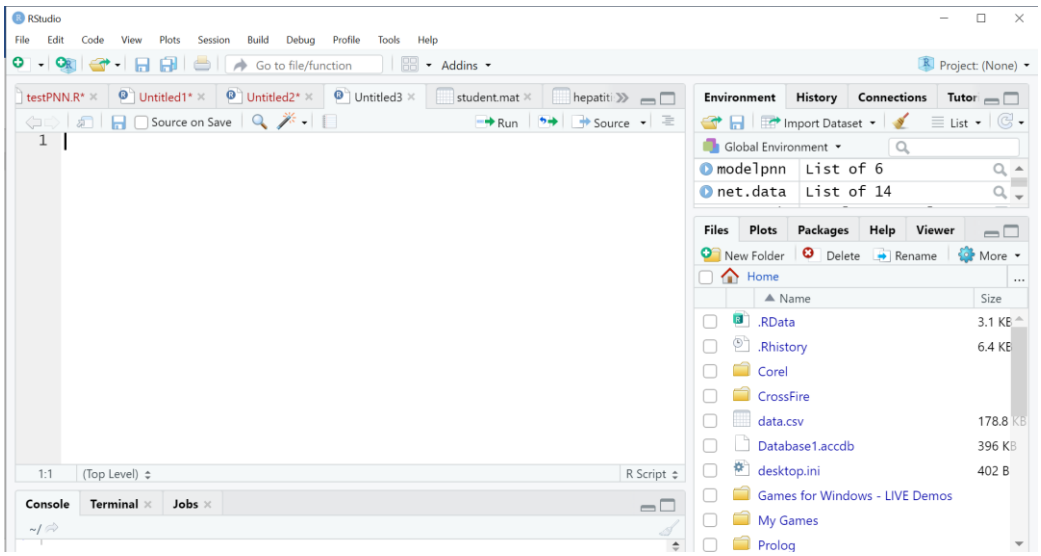

Рисунок 3.2 – Інтерфейс програмного середовища RStudio

R надає широкий спектр статистичних та графічних прийомів, наприклад:

– лінійне моделювання;
– нелінійне моделювання;
– аналіз часових рядів;
– кластеризація;
– класичні статистичні тести;
– класифікація.

R включає ефективний механізм обробки та зберігання даних, набір операторів для обчислень масивів, зокрема матриць, великий, узгоджений, інтегрований набір допоміжних інструментів для аналізу даних, графічні засоби аналізу даних та відображення на екрані.

Однією з переваг мови R є простота. За допомогою цього можна створити добре розроблені сюжети. Також, перевагою є те, що R доступний як безкоштовне програмне забезпечення. Він компілює та працює на найрізноманітніших платформах UNIX та подібних системах.

**RStudio** – безкоштовне середовище для розробки програмного забезпечення з відкритим вихідним кодом для мови програмування R, яке призначене для статистичної обробки даних і роботи з графікою [24].

Для реалізації поставленої задачі було вирішено використовувати мову R та програмне середовище RStudio.

### 3.2 Опис навчальної вибірки

Для розв’язання задачі класифікації було обрано вибірку даних, що містить в собі інформацію щодо використання комп’ютеризованої системи оцінювання якості знань з математики учнів двох португальських шкіл за навчальний рік [25].

Вибірка складається зі 149 записів. Вона містить 11 змінних, з яких 10 незалежних, а одна – залежна. Залежна змінна, в нашому випадку — змінна, що показує на стан комп’ютеризованої системи.

Система може бути у трьох станах:

– оцінка виставлена правильно;

– оцінка виставлена помилково;

– технічний збій системи під час виконання учнем тесту.

До незалежних речових змінних відносяться:

– школа;

– стать;

– вік;

– клас;

– адреса;

– ідентифікаційний код;

– оцінка за перший тест;

– оцінка за другий тест;

– оцінка за третій тест;

– оцінка за четвертий тест.

### 3.3 Створення програмних моделей нейронних мереж

На першому кроці було завантажено файл з початковими даними до програмного середовища.

Вибірка складається з наступних змінних:

1. school – школа, обрана студентом (бінарна: 'GP' – школа Габріела Перейра або 'MS' – школа Мусіно де Сільвейра).

2. sex – стать студента (бінарна: 'F' – жіночий або 'M' – чоловічий).

3. age – вік студента (числова: з 15 до 22).

4. address – домашня адреса учня (бінарна: 'U' – місто або 'R' – село).

5. form – клас, який відвідує студент (числова: з 8 по 12).

6. T1 – Оцінка за перший тест (числова: з 0 до 20).

7. T2 – Оцінка за другий тест (числова: з 0 до 20).

8. T3 – Оцінка за третій тест (числова: з 0 до 20).

9. T4 – Оцінка за четвертий тест (числова: з 0 до 20).

10. Number – ідентифікаційний код студента.

11. Class – залежна змінна (числова: система правильно виставила оцінку – 0, система помилилася – 1, система дала технічний збій – 2).

Дана вибірка не містить пропущених значень, отже обробляти її непотрібно.

Першим кроком було розглянуто реалізацію архітектури багатошарового перцептрону. Для цього потрібно використати вбудовану до R бібліотеку – neuralnet.

Спочатку необхідно поділити вибірку на тестову та тренувальну у відношенні 3:7.

З усіх наявних змінних у вибірці зрозуміло, що найбільш впливати на результат будуть змінні, що відповідають за оцінки за навчальний рік (T1,T2, T3,T4). Нижче приведено модель, що була побудована саме на цьому принципі (рис. 3.3). На екран виведено результуючу матрицю.

```
> net.data<-neuralnet(Good+Bad+Problem~T1+T2+T3+T4, trainset, hidden = 3)
> net.data$result.matrix
                                [,1]
error                   6.318525e+00
reached.threshold       9.517902e-03
steps                   6.470000e+03
Intercept.to.1layhid1  -1.268541e+01
T1.to.1layhid1          1.192076e+00
T2.to.1layhid1         -9.063791e-01
T3.to.1layhid1          1.456143e+00
T4.to.1layhid1         -1.848553e-01
Intercept.to.1layhid2  -4.259155e+00
T1.to.1layhid2          3.001725e+00
T2.to.1layhid2          3.388843e+00
T3.to.1layhid2         -1.881445e+00
T4.to.1layhid2          2.143205e+00
Intercept.to.1layhid3   1.334844e+01
T1.to.1layhid3         -1.218316e+00
T2.to.1layhid3          9.232530e-01
T3.to.1layhid3         -1.507674e+00
T4.to.1layhid3          1.867565e-01
Intercept.to.Good      -1.978572e-01
```

Рисунок 3.3 – Побудова моделі класифікації за допомогою багатошарового перцептрону

Приведемо схему нейронної мережі, що була побудована засобами RStudio, для цього було використано функцію «plot» (рис. 3.4).

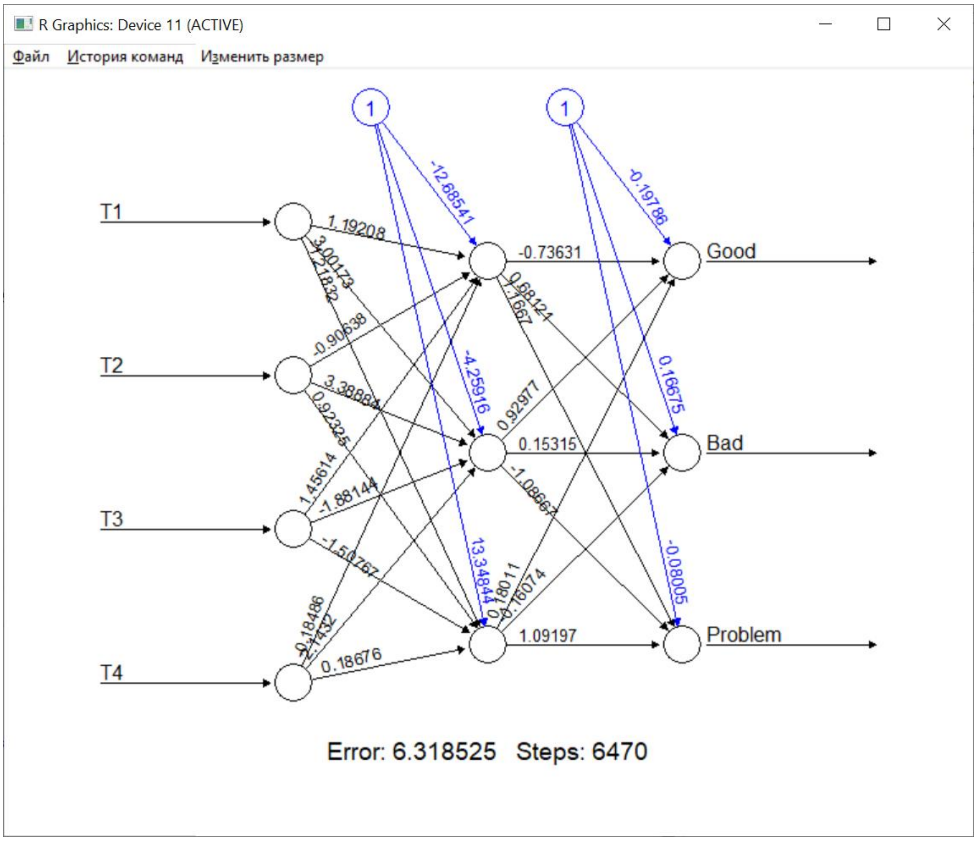


Рисунок 3.4 – Схема побудованої нейронної мережі

Нижче приведемо отриману матрицю класифікованих зразків. Виведемо точність, з якою було класифіковано вибірку (рис. 3.5).

```
> net.prob <- compute(net.data, testset[-5])$net.result
> pred = c("0",   "1", "2", "3")[apply(net.prob, 1, which.max)]
> (table(Фактично = testset$Class, Зпрогнозовано = pred))
        Зпрогнозовано
Фактично  0  1  2
       0 12  0  0
       1  3 23  0
       2  0  0  5
> rez = mean(pred == testset$Class)
> paste("Розрахована точність =", round(100*rez, 2), "%", sep = "")
[1] "Розрахована точність =93.02%"
```

Рисунок 3.5 – Матриця класифікованих зразків та розрахунок точності класифікації

Як можна бачити, побудована модель є якісною, показує відмінні результати.

Перейдемо до реалізації мережі Кохонена (рис. 3.6).

Повний лістинг програмної реалізації приведено в Додатку Г.

```
student.mat1<-som(student.mat)
predict.kohonen()
student.mat1
(table(Фактично = student.mat1$Class, Зпрогнозовано = pred))
rez = mean(pred == testset$Class)
paste("Розрахована точність =", round(100*rez, 2), "%", sep = "")
```

```
[1] "Розрахована точність =94.88%"
```

Рисунок 3.6 – Реалізація моделі класифікації за допомогою мереж Кохонена

Як можна бачити, друга побудована модель є також якісною, показує відмінні результати.

Тепер реалізуємо ймовірнісну нейронну мережу. Для цього скористаємося вбудованим пакетом в мову R – **pnn** (рис. 3.7).

```
> modelpnn<-learn(student.mat, category.column = 10)
> modelpnn$model
[1] "Probabilistic neural network"
> modelpnn$categories
[1] "0" "1" "2"
> modelpnn$k
[1] 10
> modelpnn$n
[1] 149
> modelpnn<-smooth(modelpnn,sigma=0.9)
> modelpnn$sigma
[1] 0.9
```

```
[1] "Розрахована точність =95.81%"
```

Рисунок 3.7 – Побудова моделі класифікації за допомогою ймовірнісних штучних мереж

**Висновок до 3 розділу**

В третьому розділі було розглянуто два середовища реалізації нейромережевих архітектур – PyCharm та RStudio.

RStudio - безкоштовне середовище для розробки програмного забезпечення з відкритим вихідним кодом для мови програмування R, яке призначене для статистичної обробки даних і роботи з графікою.

PyCharm - інтегроване середовище розробки для мови програмування Python. Воно надає засоби для аналізу коду, графічний відладчик, інструмент для запуску юніт-тестів і підтримує веб-розробку на Django.

PyCharm надає розумну перевірку коду, швидке виявлення помилок і оперативне виправлення, укупі з автоматичним рефакторингом коду, і багатими можливостями в навігації.

PyCharm надає спеціальну підтримку наукових бібліотек. Він підтримує Anaconda, Numpy, Matplotlib і інші наукові бібліотеки, надаючи користувачу глибоке розуміння коду, інтерактивні графіки, перегляд масивів і багато іншого.

Для реалізація поставленої задачі було обрано середовище RStudio.

Щоб реалізувати потрібні архітектури необхідно скористатися вбудованими до середовища пакетами:

– neuralnet;

– pnn;

– kohonen.

Пакет neuralnet використовується для роботи з багатошаровим перцептроном.

Пакет pnn дозволяє будувати ймовірнісні нейронні мережі.

Пакет kohonen – дозволяє працювати з мережами, картами Кохонена.

В розділі описано вибірку даних, яку потрібно було класифікувати. Вона складається зі 149 записів, 11 змінних, 10 з яких – залежні, 1 – незалежна. До незалежних змінних відносяться:

- Школа.
- Стать.
- Вік.
- Клас.
- Адреса.
- Ідентифікаційний код.
- Оцінка за перший тест.
- Оцінка за другий тест.
- Оцінка за третій тест.
- Оцінка за четвертий тест.

Вибірка є результатом проведення моніторингу знань з математики за допомогою комп’ютеризованої системи. Система може знаходить у трьох станах:

- оцінка виставлена правильно;
- оцінка виставлена помилково;
- технічний збій системи під час виконання учнем тесту.

Після побудови всіх моделей можна зробити висновки, що всі архітектури показали відмінні результати класифікації.

# ВИСНОВКИ

В даній кваліфікаційній роботі було розглянуто поняття класифікації, як одного з найбільш популярних напрямків машинного навчання. Описано моделі та методи розв’язання задачі класифікації. В практичній частині виконується реалізація задачі класифікації за допомогою трьох архітектур нейронних мереж:

– мережі Кохонена,

– багатошарового перцептрону,

– ймовірнісної нейронної мережі.

Метою даної роботи була розробка нейромережевих моделей класифікації станів комп’ютеризованої системи.

На дане дослідження ставилися задачі:

1) проаналізувати особливості задачі класифікації як задачі машинного навчання;

2) розглянути існуючі методи класифікації станів складних систем;

3) розглянути особливості застосування штучних нейронних мереж для розв’язання задачі класифікації;

4) обґрунтувати вибір архітектур штучних нейронних мереж для розв’язання задачі класифікації комп’ютеризованої системи;

5) розробити програмну реалізацію архітектур штучних нейронних мереж для розв’язання задачі класифікації;

6) протестувати створені програмні моделі штучних нейронних мереж на дійсному наборі даних та провести порівняльний аналіз результатів;

7) надати рекомендації щодо застосування створених моделей штучних нейронних мереж.

Під час виконання даної роботи всі задачі були реалізовані. На виході отримано: проаналізовані моделі та методи класифікації, розглянуті

архітектури нейронних мереж, за допомогою яких можна розв’язати задачу класифікації, наведено переваги та недоліки методів один відносно одного. Програмно реалізовані три архітектури. Після побудови можна зробити наступний висновок, що архітектури показали відмінні результати класифікації. Кожну з наведених архітектур можна використовувати на практиці.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Нейронные сети, машинное обучение и искусственный интеллект: в чем разница и для чего их используют [Електронний ресурс]. URL: https://evergreens.com.ua/ru/articles/machine-learning-overview.html (дата звернення 11.04.2021).

2. Задачи машинного обучения в ML.NET [Електронний ресурс] URL: https://docs.microsoft.com/ru-ru/dotnet/machine-learning/resources/tasks (дата звернення 12.05.2021).

3. Задача классификации [Електронний ресурс] URL: https://wiki.moda/%D0%97%D0%B0%D0%B4%D0%B0%D1%87%D0%B0_%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D0%B8 (дата звернення 22.05.2021).

4. Классификация [Електронний ресурс] URL: http://www.machinelearning.ru/wiki/index.php?title=%D0%9A%D0%BB%D0%B0%D1%81%D1%81%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%86%D0%B8%D1%8F (дата звернення 22.05.2021).

5. Модели для предсказания класса объектов [Електронний ресурс] URL: https://ranalytics.github.io/data-mining/023-Models-for-Class-Prediction.html (дата звернення 19.07.2021).

6. Многоклассовая классификация в задаче семантической сегментации [Електронний ресурс] URL: https://www.graphicon.ru/html/2009/conference/se14/130/130_Paper.pdf

7. Ryan Rifkin, Aldebaro Klautau In Defense of One-Vs-All Classification // The Journal of Machine Learning Research. 2004. Volume 4. P. 101-141

8. Нейронная сеть (Neural network) [Електронний ресурс] URL: https://wiki.loginom.ru/articles/neural-network.html (дата звернення 02.06.2021)

9. Логистическая регрессия в медицине и биологии [Електронний ресурс] Режим доступа: http://www.biometrica.tomsk.ru/logit_0.htm (дата звернення 11.09.2021)

10. Пробит-модель регрессии [Електронний ресурс] Режим доступа: https://www.statmethods.ru/statistics-metody/probit-model-regressii/ (дата звернення 11.09.2021)

11. Інформаційні системи та технології в управлінні. Методичні вказівки, теоретичні відомості і завдання до лабораторних робіт для студентів та магістрів денної форми навчання спеціальності 7.803060101 Менеджмент організацій і адміністрування. Частина 3. Класифікація в бізнес-аналітиці. / Укл.: Біла Н.І. – Запоріжжя: ЗНТУ, 2014. – с. 50. [Електронний ресурс] Режим доступа: http://eir.zntu.edu.ua/bitstream/123456789/342/1/met_vk_bila_3.pdf (дата звернення 10.10.2021)

12. Метод опорных векторов [Електронний ресурс] Режим доступа: https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BE%D0%BF%D0%BE%D1%80%D0%BD%D1%8B%D1%85_%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%BE%D0%B2 (дата звернення 10.10.2021)

13. Дискриминантный анализ [Електронний ресурс] Режим доступа: https://www.statmethods.ru/statistics-metody/diskriminantnyj-analiz/ (дата звернення 15.10.2021)

14. Нейронные сети Кохонена [Електронний ресурс] Режим доступа: https://neuronus.com/theory/nn/955-nejronnye-seti-kokhonena.html (дата звернення 15.10.2021)

15. Многослойный персептрон (Multilayered perceptron) [Електронний ресурс] Режим доступа: https://wiki.loginom.ru/articles/multilayered-perceptron.html (дата звернення 23.10.2021)

16. М.А. Новотарський, Б.Б. Нестеренко. Штучні нейронні мережі: обчислення // Праці Інституту математики НАН України. – Т50. – Київ: Ін-т математики НАН України, 2004. – 408 с.

17. Многослойный перцептрон Румельхарта [Електронний ресурс] Режим доступа: https://cybernetics.fandom.com/ru/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D1%81%D0%BB%D0%BE%D0%B9%D0%BD%D1%8B%D0%B9_%D0%BF%D0%B5%D1%80%D1%86%D0%B5%D0%BF%D1%82%D1%80%D0%BE%D0%BD_%D0%A0%D1%83%D0%BC%D0%B5%D0%BB%D1%8C%D1%85%D0%B0%D1%80%D1%82%D0%B0 (дата звернення 25.10.2021)

18. Вероятностная нейронная сеть - Probabilistic neural network [Електронний ресурс] Режим доступа: https://hrwiki.ru/wiki/Probabilistic_neural_network (дата звернення 11.11.2021)

19. Фішер - 1912 р. Математичний енциклопедичний словник, М.: Радянська енциклопедія, 1988

20. Ким О. Дж., Мьюллер Ч.У., Клекка У.Р., и др. Факторный, дискриминантный и кластерный анализ / Под ред. И. С. Енюкова. М.: Финансы и статистика, 1989. 215 с.

21. Що таке Python? [Електронний ресурс] Режим доступа: http://www.plug.org.ua/documentation/about-python (дата звернення 11.11.2021)

22. Введение в анализ данных [Електронний ресурс] Режим доступа: https://mipt-stats.gitlab.io/courses/ad_fivt/python_digest.html (дата звернення 15.11.2021)

23. What is R? [Електронний ресурс] Режим доступа: https://www.r-project.org/about.html (дата звернення 20.11.2021)

24. Програмування в середовищі r. Середа статистичних обчислень R: досвід використання у викладанні. Приклад роботи з пакетом adwordsR [Електронний ресурс] Режим доступа: https://sukachoff.ru/uk/ustrojjstva/programmirovanie-v-srede-r-sreda-statisticheskih-vychislenii-r-opyt/ (дата звернення 28.11.2021)

25. Student Performance Data Set [Електронний ресурс] Режим доступа: https://archive.ics.uci.edu/ml/datasets/Student+Performance (дата звернення 28.11.2021)

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Харківський національний університет імені В.Н. Каразіна

Факультет комп'ютерних наук
Кафедра теоретичної та прикладної системотехніки
Освітньо-кваліфікаційний рівень Магістр
Спеціальність 151 – Автоматизація та комп'ютерно інтегровані технології

**ЗАТВЕРДЖУЮ**

**Завідувач кафедри теоретичної та прикладної системотехніки**

_____ С.І. Шматков
підпис ініціали, прізвище

"____" _____2021 року

# З А В Д А Н Н Я
## НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ

Ткаченко Андрій Максимович
(прізвище, ім'я, по батькові студента)

1. Тема роботи Нейромережеві моделі класифікації стану комп'ютеризованої системи

керівник роботи: Стрілець Вікторія Євгенівна, к.т.н.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від "___"_____2021 року №___

2. Строк подання студентом роботи_____

3. Перелік питань, які потрібно розробити

1. Огляд і аналіз існуючих моделей та методів класифікації стану комп'ютеризованих систем.

2. Побудова нейромережевих моделей класифікації стану комп'ютеризованих систем

3. Програмна реалізація нейромережевих моделей класифікації стану комп'ютеризованої системи.

4. Порівняльний аналіз розглядуваних нейромережевих моделей класифікації

5. Рекомендації щодо використання моделей класифікації стану комп'ютеризованої системи.

4. План роботи

| № з/п | Назви етапів роботи | Дата |
|---|---|---|
| | Дослідження та аналіз існуючих моделей класифікації стану складних систем. | 15.12.2020 – 10.02.2021 |
| | Аналіз нейромереж, які використовуються для розв’язання задач класифікації | 11.02.2021 – 15.03.2021 |
| | Розробка нейромережевих моделей класифікації стану комп’ютеризованої системи. | 25.03.2021 – 30.04.2021 |
| | Програмна реалізація нейромережевих моделей класифікації стану комп’ютеризованої системи. | 01.05.2021 – 18.06.2021 |
| | Тестування та порівняння нейромережевих моделей класифікації стану комп’ютеризованої системи. | 01.09.2021 – 03.10.2021 |
| | Розробка рекомендацій щодо застосування нейромережевих моделей класифікації стану комп’ютеризованої системи. | 10.10.2021 – 15.11.2021 |
| | Оформлення пояснювальної записки. | 16.11.2021– 30.11.2021 |

5. Дата видачі завдання_____

**Студент** _____ А.М. Ткаченко

підпис інніціали, прізвище

**Керівник роботи** _____ В.Є. Стрілець

підпис інніціали, прізвище

**Додаток Б**

Затверджую

_____

«_____»_____2021 р.

**Технічне завдання**
**на розробку програмного виробу**
**«Модель класифікації стану комп'ютеризованої системи оцінювання знань»**

| Назва розділу | Назва і зміст підрозділу |
|---|---|
| 1. Введення | 1) Назва: Модель класифікації стану комп'ютеризованої системи оцінювання знань.<br><br>2) Область застосування: освіта, оцінювання якості освіти учнів. |
| 2. Підстава для розробки | 1) Навчальний план ФКН за спеціальністю 151 – Автоматизація комп'ютерно-інтегровані технології.<br><br>2) Завдання на кваліфікаційну роботу, затверджене наказом № 0210-05/1804 від 10.09.2021р. привести в Додатку А. |
| 3. Призначення розробки | 1) Метою даної роботи є розробка нейромережевих моделей класифікації станів комп'ютеризованої системи.<br><br>2) Призначення розробки: розробити модель, що допоможе виявляти випадки некоректної роботи комп'ютеризованої системи оцінювання знань, шляхом класифікації станів. |

| | |
|---|---|
| | 3) Вихідні дані для розробки: вірно класифіковані стани системи. |
| 4. Вимоги до програмного виробу | 1) Вимоги до функціональних характеристик: програмна модель повинна класифікувати стани системи з максимальною точністю.<br>2) Вимоги до надійності: точність класифікації повинна бути не менше 80%.<br>3) Вимоги до умов експлуатації: для виконання програми потрібно, щоб на ПК було встановлено стабільне інтернет-з'єднання, а також середовище, за допомогою якого можна «відкрити» створену модель.<br>4) Вимоги до складу і параметрів технічних засобів: для виконання програми потрібен комп'ютер, оснащений стабільним інтернет-з'єднанням.<br>5) Вимоги до інформаційної та програмної сумісності: підтримка наступних ОС: Windows, Linux, MacOS.<br>6) Вимоги до маркування та упаковки: відсутні.<br>7) Вимоги до транспортування і зберігання: відсутні.<br>8) Спеціальні вимоги: відсутні. |
| 5. Вимоги до програмної документації. | Програмною документацією до виробу «Модель класифікації стану комп'ютеризованої системи оцінювання знань» вважати:<br>1) Дане Технічне завдання на розробку програмного виробу (представити у вигляді Додатку Б до пояснювальної записки до кваліфікаційної роботи).<br>2) Програму і методику випробувань розробленого програмного виробу (представити у вигляді Додатку В до пояснювальної записки до кваліфікаційної роботи). |

<table>
<tr><td></td><td colspan="2">3) Опис програмного виробу (представити в розділі 3 пояснювальної записки до кваліфікаційної роботи).<br>4) Лістинг програмного коду (представити у Додатку Г).</td></tr>
<tr><td>6. Техніко-економічні показники</td><td colspan="2">1) Орієнтована оцінка ефективності виконуваної роботи: непотрібна.<br><br>2) Терміни і можливі витрати грошових коштів на розробку програмного виробу: не потрібні.<br><br>3)Порівняння економічних показників розробленого програмного виробу з вітчизняними і зарубіжними зразками або аналогами (Представити у розділі 4 пояснювальної записки).</td></tr>
<tr><td rowspan="8">7. Стадії і етапи розробки</td><td>Дата</td><td>Назва етапу</td></tr>
<tr><td>15.04.21 - 5.05.21</td><td>Дослідження та аналіз існуючих моделей класифікації стану складних систем.</td></tr>
<tr><td>16.06.21-16.07.21</td><td>Аналіз нейромереж, які використовуються для розв’язання задач класифікації</td></tr>
<tr><td>17.07.21-17.09.21</td><td>Розробка нейромережевих моделей класифікації стану комп’ютеризованої системи.</td></tr>
<tr><td>17.09.21-17.10.21</td><td>Програмна реалізація нейромережевих моделей класифікації стану комп’ютеризованої системи.</td></tr>
<tr><td>17.10.21-30.10.21</td><td>Тестування та порівняння нейромережевих моделей класифікації стану комп’ютеризованої системи.</td></tr>
<tr><td>01.11.21-15.11.21</td><td>Розробка рекомендацій щодо застосування нейромережевих моделей класифікації стану комп’ютеризованої системи.</td></tr>
<tr><td>15.11.21-30.11.21</td><td>Оформлення пояснювальної записки.</td></tr>
</table>

| | |
|---|---|
| 8. Порядок контролю і приймання | 1) Перевірку ходу розробки програмного виробу Керівнику робіт виконувати 1 раз на тиждень.<br>2) Випробування програмного виробу відповідно до Програми і методики випробувань привести у вигляді додатку В.<br>3) Захист розробленого програмного виробу провести на засіданні атестаційної комісії.<br>4) Пояснювальну записку представити на паперових носіях в одному примірнику, в електронному вигляді - на CD-диску в одному екземплярі. |

Виконавець
студент групи КУ-61
Ткаченко А.М.
_____

Замовник
к. т. н., доцент кафедри ТПС
Стрілець В. Є.
_____

**Додаток В**

**Програма і методика випробувань**
**програмного виробу**
«Модель класифікації стану комп’ютеризованої системи оцінювання знань»

**1 Об'єкт випробувань**

1.1 Найменування випробуваного програмного виробу: «Модель класифікації стану комп’ютеризованої системи оцінювання знань»

1.2 Область його застосування: освіта (оцінювання якості освіти).

1.3 Умовне позначення розробки (при необхідності).

**2. Мета випробувань**

Підтвердження коректності функціонування програмного виробу.

**3. Загальні положення**

**3.1 Підстави для проведення випробувань**

Підставою для проведення випробувань є наказ про призначення атестаційної комісії.

**3.2 Місце і тривалість випробувань**

Приймальні (приймально-здавальні) випробування проводяться дистанційно в період роботи атестаційної комісії.

**3.3 Обсяг випробувань**

Приймальні випробування програмного вироби проводяться в обсязі відповідному цієї Програми і методики випробувань.

**3.4 Організації, які беруть участь у випробуваннях**

Приймальні випробування проводяться атестаційною комісією напередодні засідання за участю Замовника, Виконавця та інших осіб, присутніх на засіданні в дистанційному режимі.

**4. Вимоги до програми або програмного виробу**

Модель повинна:

1) Представляти з себе комп’ютерну реалізацію моделей класифікації станів комп’ютеризованої системи;

2) Забезпечити можливість завантаження користувачем нових даних, для процесу аналізу;

3) Забезпечувати можливість виведення результату на екран за допомогою інтуїтивно зрозумілого інтерфейсу;

4) Забезпечити можливість навчання системи за допомогою нейронних мереж.

4.2. Вимоги до надійності:

1) Програма повинна видавати повідомлення про помилки.

4.3. Вимоги до умов експлуатації:

1) Умови експлуатації співпадають з умовами експлуатації персональної електронно-обчислювальної машини, на якій буде працювати, а також сумісних з нею персональними комп'ютерами.

2) Для користування програмою користувачу необхідно пройти короткий курс навчання роботі з програмою.

4.4. Вимоги до складу і параметрів технічних засобів:

1) Персональний комп'ютер у повній комплектації або ноутбук.

4.5. Вимоги до інформаційної та програмної сумісності:

«Windows» будь-яка версія, залежно від версії RStudio, «Linux» (будь-який дистрибутив), R, RStudio.

6) Вимоги до маркування та упаковки (не висуваються);

7) Вимоги до транспортування і зберігання (не висуваються);

8) Спеціальні вимоги (не пред'являються).

## 5. Вимоги до програмної документації

Склад програмної документації, що подається на випробування, включає:

1) Технічне завдання на розробку програмного вироби (представлено в Додатку Б до пояснювальної записки до дипломної роботи).

2) Ця Програма і методика випробувань розробленого програмного виробу (представлена в Додатку В до пояснювальної записки до дипломної роботи).

3) Опис програмного виробу (представлено в розділі 3 пояснювальної записки до дипломної роботи).

4) Лістинг програми (представлений в Додатку Г до пояснювальної записки до дипломної роботи) .

## 6. Засоби і порядок випробувань

### 6.1 Засоби випробувань

Випробування проводяться на технічних засобах, таких як персональний комп'ютер у повній комплектації або ноутбук.

Випробування проводяться з використанням програмних засобів, таких як операційна система «Windows» будь-яка версія, залежно від версії RStudio, «Linux» (будь-який дистрибутив), R, RStudio.

### 6.2 Порядок проведення випробувань

1) Перевірка програмної документації

1.1. Перевірка складу програмної документації. Перевірку здійснювати за критерієм наявності, представленої в ТЗ документації.

1.2. Критерієм успішності тесту вважати відповідність наявної документації згідно зі списком в ТЗ.

1.3. Перевірка якості програмної документації. Перевірку здійснювати за критерієм відповідності вимогам ЕСПД.

1.4. Модель працює відповідно до умов експлуатації ОС MS Windows будь-якої версії, залежно від версії RStudio, а також сумісних з ним.

1.5. Критерієм успішності тесту вважати відповідність якості наявної документації згідно з вимогами ЕСПД.

2) Для роботи необхідні встановлені програмні засоби R, RStudio .

Тест 1. Перевірка сумісності моделі з ОС та програмним забезпеченням

Критерієм успішності вважати сумісність моделі з ОС, R, RStudio.

Порядок проведення випробувань:

- Запуск програми здійснюється завантаженням в середовище RStudio програмного коду, з підключенням початкових даних.

- Завантажуються додаткові пакети, необхідні для роботи.

- Порядково запускається програмний код, для відстежування роботи програми на кожному кроці.

Висновки: після проведення тесту встановлено, що створена модель успішно функціонує на ОС, на якій тестувалася, завантажується до програмного середовища, проходить процес відладки.

Тест 2. Відповідність створеної моделі до поставленої мети

Для проведення випробувань пропонується провести ті заходи, опис яких містяться в розділі 3. Критерієм успішності є коректні отримані результати на виході.

```
> plot(net.data)
> net.prob <- compute(net.data, testset[-5])$net.result
> pred = c("0",   "1", "2", "3")[apply(net.prob, 1, which.max)]
> (table(Фактично = testset$Class, Зпрогнозовано = pred))
        Зпрогнозовано
Фактично  0  1  2
       0 12  0  0
       1  3 23  0
       2  0  0  5
> rez = mean(pred == testset$Class)
> paste("Розрахована точність =", round(100*rez, 2), "%", sep = "")
[1] "Розрахована точність =93.02%"
```

Рисунок В.1 — Результат виконання Теста 2

Висновок: в результаті даного тесту було встановлено, що побудована модель повністю відповідає поставленій меті. Процес класифікації проходить успішно, результати по точності 93.02, 95.81, 94.88 вказують на відмінний результат.

Висновки: тести 1 та 2 пройдені успішно, отримані результати є коректними, процес класифікації відмінним.

Виконавець

Ткаченко А.М.

## Додаток Г

### Лістинг програмного коду

```
library(pnn)

library(kohonen)

student.mat     <-     read.csv("C:/Users/Asus/Desktop/student/student-
mat.csv", sep=";")

View(student.mat)

head(student.mat)

set.seed(123)

ind <- sample(2, nrow(student.mat), replace = TRUE, prob = c(0.7, 0.3))

trainset <- student.mat[ind == 1, ]

testset <-student.mat[ind == 2, ]

testset

trainset$Good<-trainset$Class=="0"

trainset$Bad<-trainset$Class=="1"

trainset$Problem<-trainset$Class=="2"

trainset

set.seed(1)

net.data<-neuralnet(Good+Bad+Problem~T1+T2+T3+T4, trainset, hidden = 3)

net.data$result.matrix

plot(net.data)

net.prob <- compute(net.data, testset[-5])$net.result

pred = c("0",   "1", "2", "3")[apply(net.prob, 1, which.max)]

(table(Фактично = testset$Class, Зпрогнозовано = pred))

rez = mean(pred == testset$Class)

paste("Розрахована точність =", round(100*rez, 2), "%", sep = "")

modelpnn<-learn(student.mat, category.column = 10)

modelpnn$model

modelpnn$categories

modelpnn$k

modelpnn$n
```

```
modelpnn<-smooth(modelpnn,sigma=0.9)

modelpnn$sigma

set.seed(123)

student.mat1<-som(student.mat)

predict.kohonen()

student.mat
```