###### **МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**КРИВОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

**ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ**

**З В І Т**

**Лабораторна робота №2  
з дисципліни**

**«Сучасні методи та моделі  
інтелектуальних систем керування»**

Виконавець:

аспірант групи АКІТР-23-1а Косей М.П.

Керівник:

викладач Тиханський М. П.

2024

**Лабораторна робота №2**

Тема: Багатошарова нейронна мережа прямого поширення

Мета: Одержати основні навички побудови моделей штучних нейромереж

**ХІД РОБОТИ**

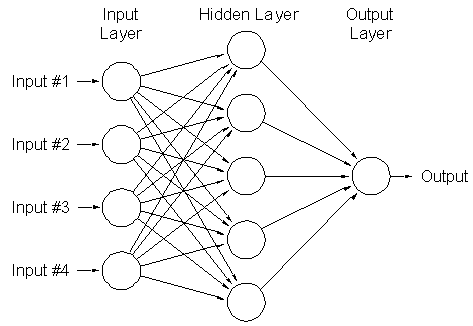
**1) Ознайомитись з теоретичними відомостями до лабораторної роботи**

**БАГАТОШАРОВІ НЕЙРОННІ МЕРЕЖІ**

Багатошарова нейронна мережа прямого поширення (**MLP — Multilayer Perceptron**) є одним із базових типів штучних нейронних мереж. Вона складається з кількох шарів нейронів, з'єднаних таким чином, що інформація проходить тільки в одному напрямку — від вхідного шару до вихідного, через один або більше прихованих шарів.

**MLP** складається з:

1. Вхідного шару: отримує дані на вході.
2. Прихованих шарів: виконують нелінійні перетворення вхідних даних.
3. Вихідного шару: генерує остаточний результат (наприклад, класифікація або передбачення).



**Рисунок 2.1 - Багатошарова нейронна мережа прямого   
поширення (MLP — Multilayer Perceptron)**

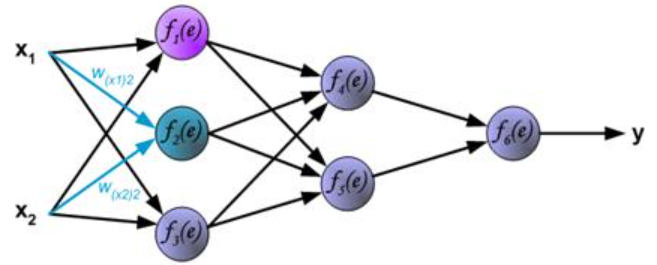
Кожен нейрон в шарі з'єднаний з усіма нейронами попереднього шару, що робить мережу повнозв'язаною.

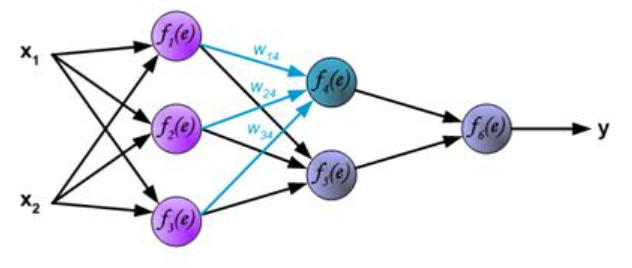
Один з методів навчання нейромереж - зворотнє поширення (**Backpropagation**) - це один з найпоширеніших методів навчання нейронних мереж, який використовується для тренування штучних нейронних мереж з вчителем (supervised learning). Він використовується для покращення ваг (внутрішніх параметрів) мережі на основі помилок між прогнозованими виходами мережі та відповідними цільовими виходами (мітками) у навчальному наборі даних.

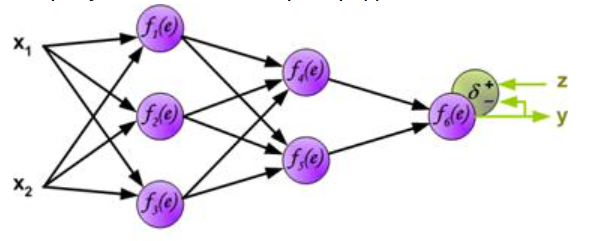
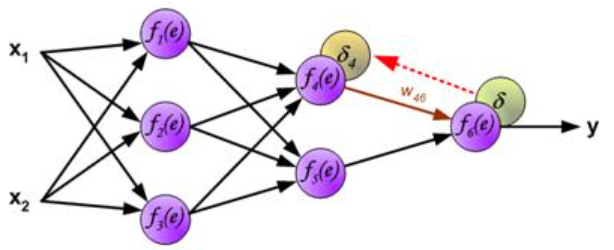
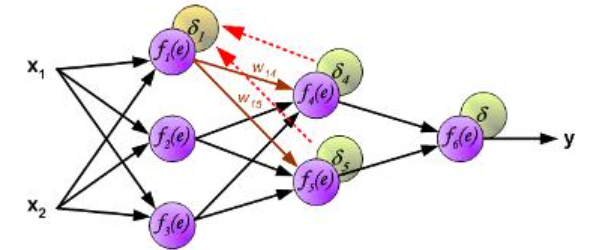
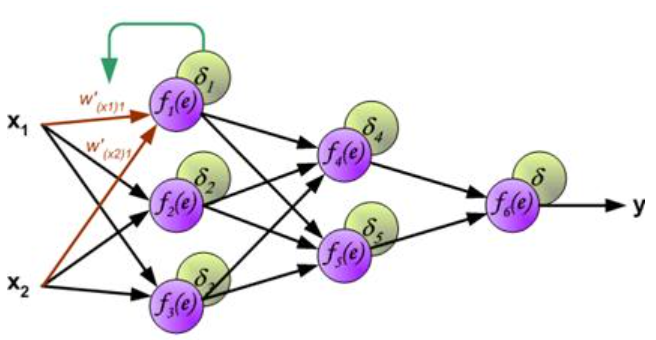
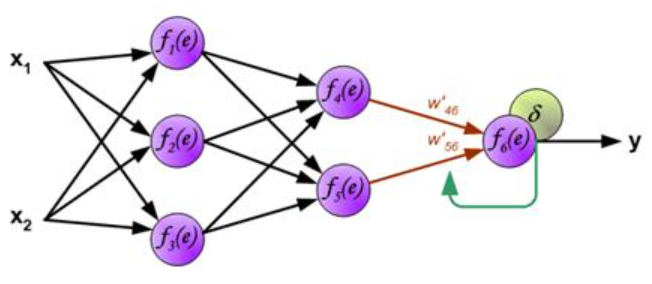
Основна ідея за методом зворотного поширення полягає в пошуку градієнту функції втрати (loss function) відносно ваг мережі, і використання цього градієнту для оновлення ваг таким чином, щоб зменшити помилки прогнозів мережі на навчальному наборі даних. Оновлення ваг відбувається за допомогою алгоритму градієнтного спуску (gradient descent), де ваги оновлюються у напрямку, протилежному до градієнту функції втрати, з метою мінімізації функції втрати.

Процес навчання нейромережі зворотнім поширенням можна узагальнити наступними кроками:

Ініціалізація ваг мережі:

1. Ваги мережі ініціалізуються випадковими значеннями або за допомогою іншого методу.



1. Прямий прохід (forward pass): Вхідні дані проходять крізь мережу від вхідного до вихідного шару, розраховуючи вихідні значення мережі.
2. Розрахунок помилок (loss calculation): Розраховується значення функції втрати, яка відображає різницю між прогнозованими виходами мережі та цільовими виходами (мітками) у навчальному наборі даних.
3. Зворотнє поширення градієнту (gradient backpropagation): За допомогою ланцюгового правила (chain rule) розраховується градієнт функції втрати відносно вихідних значень мережі. Градієнт використовується для розрахунку градієнтів ваг на всіх рівнях мережі, починаючи з вихідного шару і рухаючись назад до вхідного шару.
4. Оновлення ваг (weight update): Ваги мережі оновлюються на основі розрахованого градієнту. Зазвичай використовується алгоритм градієнтного спуску (gradient descent) або його варіанти, які використовують параметр - швидкість навчання (learning rate) для контролю швидкості оновлення ваг.
5. Повторення процесу: Процес прямого проходу, розрахунку помилок, зворотного поширення градієнту та оновлення ваг повторюється для кожного прикладу у навчальному наборі даних протягом кількох епох (повторень) з метою покращення точності мережі.
6. Зупинка навчання: Навчання може бути зупинено за певної умови, такої як досягнення максимальної кількості епох, досягнення заданої точності, або іншої заданої умови.

**2) Практична частина – моделювання багатошарових нейронних мереж прямого поширення (MLP — Multilayer Perceptron).**

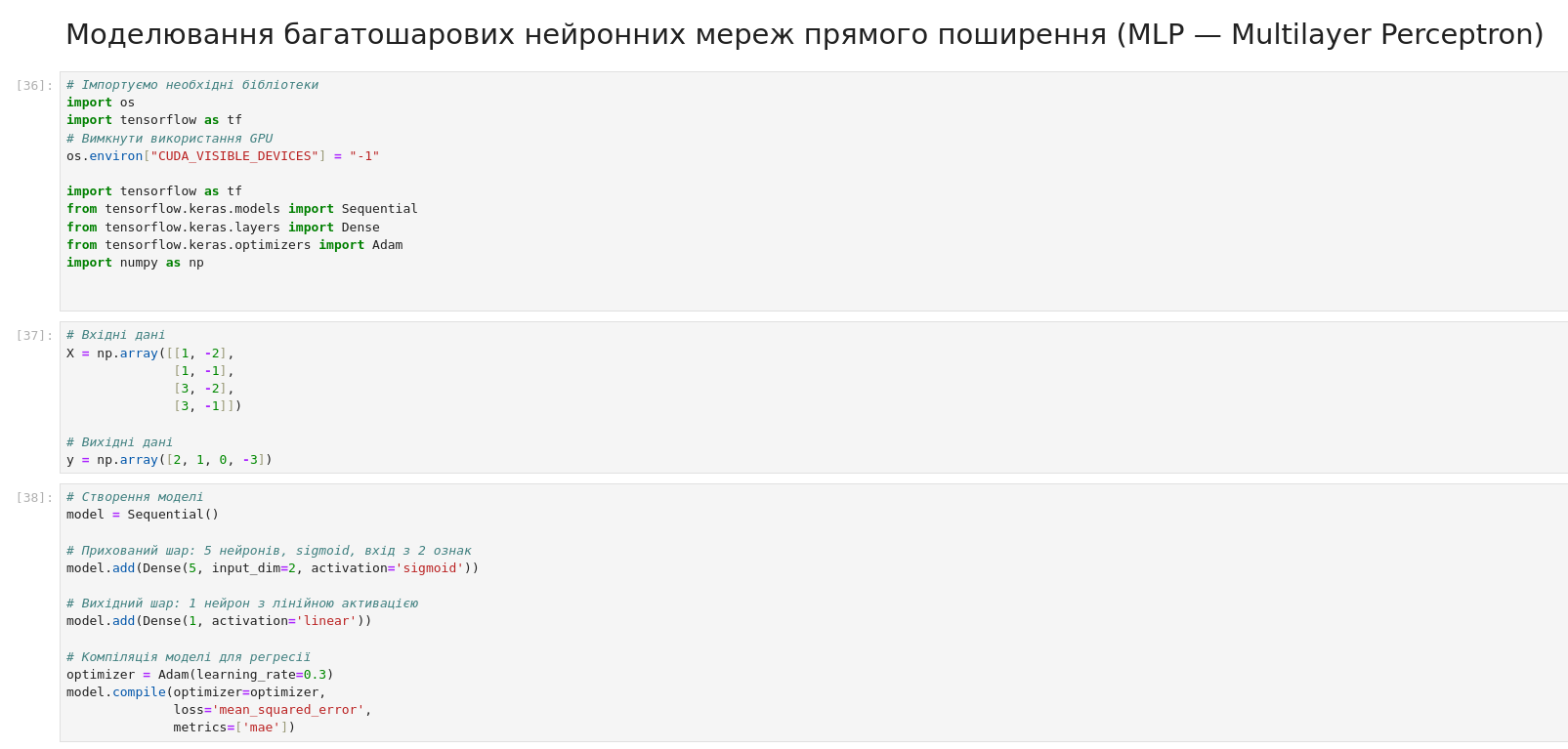
Для моделювання нейронних мереж також можна використовувати наступні інструменти:

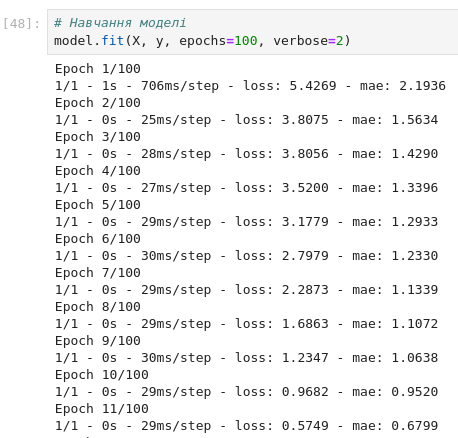
1. **MATLAB** – популярне середовище для математичних обчислень, яке має вбудовані інструменти для побудови та навчання нейронних мереж. MATLAB зручний для наукових досліджень, а його модулі для нейронних мереж полегшують процес моделювання без необхідності глибокого занурення в програмування.
2. **Python** – універсальна мова програмування з багатим набором бібліотек для роботи з нейронними мережами (наприклад, TensorFlow, PyTorch, Keras). Python є найпоширенішою мовою для машинного навчання завдяки простоті та багатофункціональності.
3. **Kaggle** – платформа для змагань з машинного навчання, де можна знайти готові набори даних, різноманітні приклади використання нейронних мереж, а також навчальні матеріали. Kaggle також має вбудоване середовище для запуску Jupyter Notebook, що спрощує експерименти з моделями.
4. **Jupyter Lab** – інтегроване середовище для обчислень, яке дозволяє створювати та виконувати Python-код у вигляді нотаток. Ідеально підходить для моделювання нейронних мереж, тестування моделей, аналізу результатів та візуалізації.

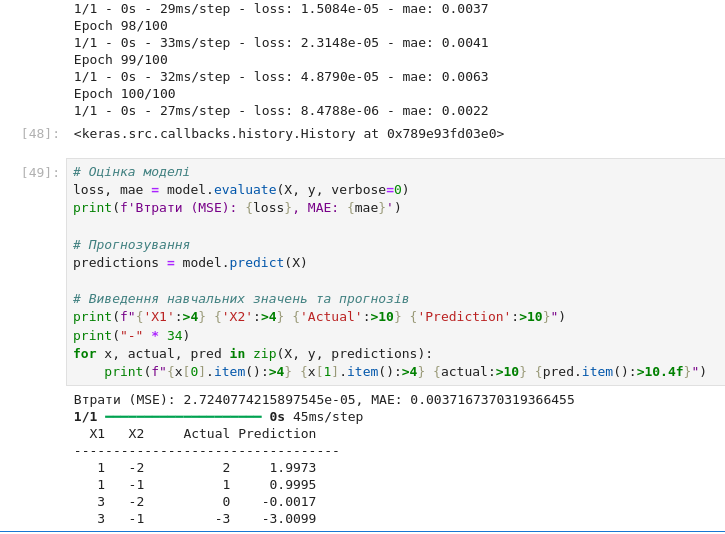
Використовуємо мову програмування **Python** та проект [**JupyterLab**](https://jupyter.org/).



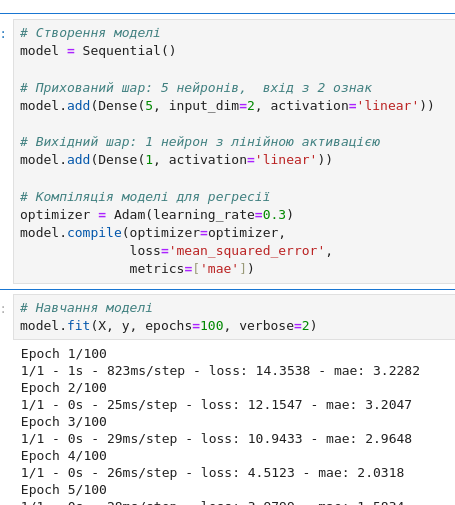
а) Створити двошарову нейронну мережу прямого поширення з сигмоїдальною активаційною функцією 'logsig' з двома входами. Навчити штучну нейронну мережу обчислювати значення функції на основі методу зворотного поширення помилки.

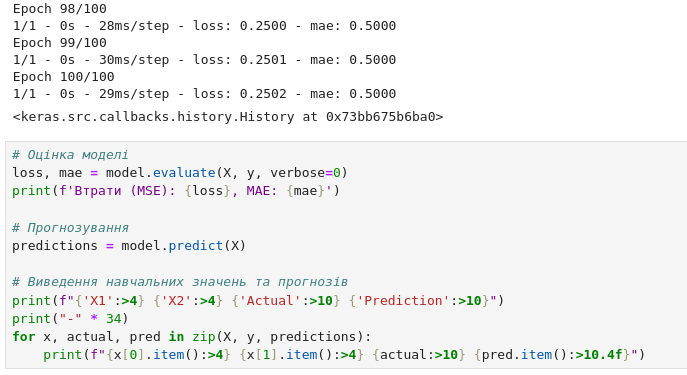


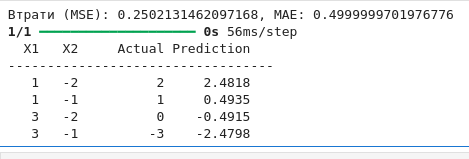


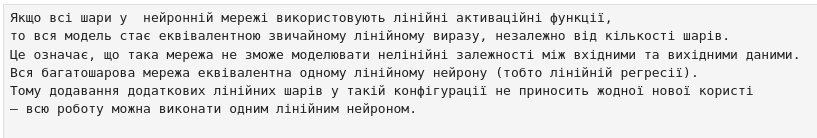


б) Створити двошарову нейронну мережу прямого поширення з лінійною активаційною функцією 'purelin' з двома входами. Навчити штучну нейронну мережу обчислювати значення функції на основі методу зворотного поширення помилки.

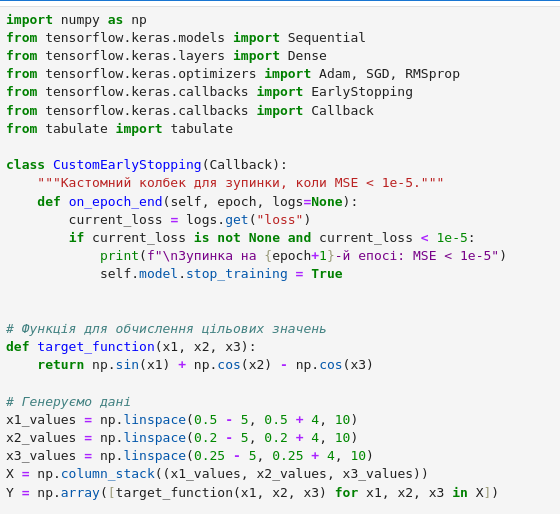


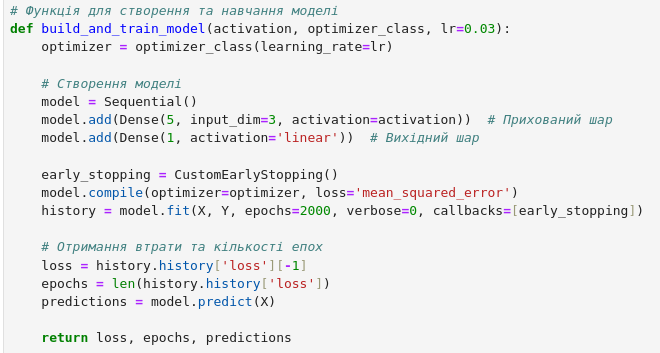


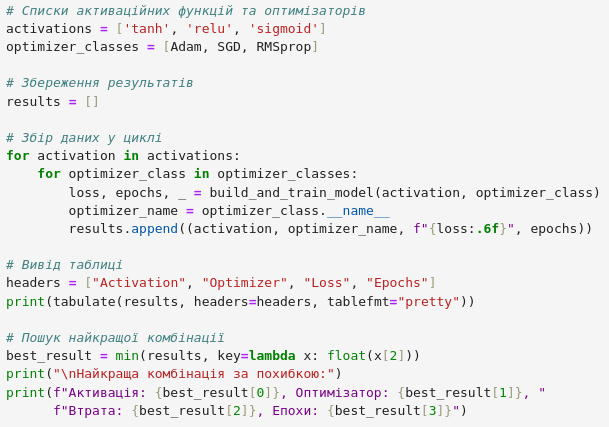


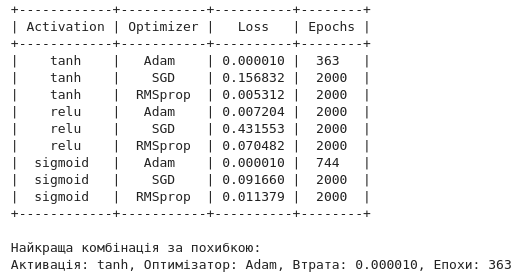


в) Обчислити 10 значень функції sin(x1) + cos(x2) - cos(x3) по заданих значення аргументів x1=1/2, x2=1/5, x3=1/4 та їх варіаціях з кроком ±1.









**ВИСНОВКИ**

**В результаті виконаної лабораторної роботи розроблені моделі багатошарових нейронних мереж прямого поширення.**

**Усі матеріали викладенні у репозіторії GitHub, за посиланням** <https://github.com/Max11mus/-LAB2-Modern-Methods-and-Models-of-Intelligent-Control-Systems>**.**