МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ДЕРЖАВНИЙ ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД

«УЖГОРОДСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ»

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАТИКИ ТА ФІЗИКО-МАТЕМАТИЧНИХ ДИСЦИПЛІН

ЗВІТ

з переддипломної практики

Реєстраційний № \_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Параска Богдан Володимирович

студент 4 курсу

денної форми навчання

залікова книжка № \_\_\_

ЗВІТ ПРО ПРОХОДЖЕННЯ ПЕРЕДДИПЛОМНОЇ ПРАКТИКИ

Спеціальність "Комп’ютерні науки (Інформатика)"

Рекомендовано до захисту з оцінкою:

"\_\_\_\_\_\_\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Підпис керівника практики

"\_\_\_\_" \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 р.

Керівник

практики від ВНЗ

доктор технічних наук,

професор Ніколенко В. В.

Ужгород – 2025

ЗМІСТ

[ВСТУП 2](#_Toc199350743)

[РОЗДІЛ 1. ВІДОМОСТІ ПРО БАЗУ ПРАКТИКИ ТОВ «ЦБО СКАЙТЕК» 3](#_Toc199350744)

[1.1. Відомості про ТОВ «ЦБО СКАЙТЕК» 3](#_Toc199350745)

[РОЗДІЛ 2. ТЕОРЕТИЧНІ ЗАВДАННЯ 6](#_Toc199350746)

[2.1 Поняття штучних нейронних мереж 6](#_Toc199350747)

[2.2 Основи машинного навчання як фундаменту для нейромереж 8](#_Toc199350748)

[2.3 Типи архітектур нейронних мереж у задачах прогнозування 14](#_Toc199350749)

[РОЗДІЛ 3. ПРАКТИЧНЕ ЗАВДАННЯ 19](#_Toc199350750)

[3.1. Вибір програмних засобів 19](#_Toc199350751)

[3.2. Оцінка основних бібліотек та фреймворків 20](#_Toc199350752)

[ВИСНОВКИ 23](#_Toc199350753)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ ТА ДЖЕРЕЛ 25](#_Toc199350754)

[ДОДАТКИ 26](#_Toc199350755)

# ВСТУП

Прогнозування цін є одним із ключових завдань у фінансовій аналітиці, торгівлі та економіці загалом. У сучасних умовах, коли обсяг і складність даних зростають, все більшого значення набувають методи штучного інтелекту, зокрема нейронні мережі. Вони демонструють високу здатність до виявлення складних залежностей у часових рядах і дозволяють досягати точніших результатів у порівнянні з класичними статистичними методами.

Метою цього проєкту є дослідження ефективності різних архітектур нейронних мереж у задачах прогнозування цін (на прикладі криптовалют Bitcoin та Ethereum). Передбачається реалізація, тестування та порівняння таких моделей, як багатошаровий перцептрон (MLP), рекурентна нейронна мережа LSTM, згорткова нейронна мережа (CNN), XGBoost та трансформери. Для досягнення цієї мети заплановано виконання наступних завдань:

**Аналіз вимог до дипломного проєкту:** буде сформульовано чіткі цілі дослідження, обсяг задач і обґрунтовано вибір теми з урахуванням актуальності прогнозування цін у сучасному цифровому середовищі.

**Огляд існуючих підходів та технологій:** проведено аналіз сучасних методів машинного навчання, що застосовуються для аналізу часових рядів, зокрема в контексті прогнозування фінансових показників. Обрані методи буде обґрунтовано на основі їх точності, складності реалізації та практичної придатності.

**Підготовка даних та реалізація моделей:** виконано обробку часових рядів, побудовано архітектури обраних моделей у середовищі Python з використанням бібліотек PyTorch, scikit-learn, XGBoost та інших.

**Навчання та тестування моделей:** буде проведено тренування моделей на історичних даних із подальшим тестуванням їхньої точності, стійкості та здатності до генералізації.

**Візуалізація та порівняльний аналіз результатів:** створено графіки реальних і прогнозованих значень, виконано оцінювання ефективності моделей за метриками RMSE, MAE та R².

# РОЗДІЛ 1. ВІДОМОСТІ ПРО БАЗУ ПРАКТИКИ

## 1.1 Відомості про ТОВ «ЦБО СКАЙТЕК»

Базою проходження практики є Центр бізнес-обслуговування **«СкайТек»** – компанія, що надає широкий спектр ІТ-послуг як для бізнесу, так і для приватних клієнтів. Основні напрямки діяльності центру охоплюють комп’ютерне обслуговування, автоматизацію бізнес-процесів, розробку веб-сайтів, бухгалтерський супровід та створення програмного забезпечення.

**«СкайТек»** відзначається гнучким підходом до клієнтів, індивідуальними консультаціями, а також можливістю виїзду спеціаліста або віддаленого усунення технічних проблем. Працівники компанії мають глибокі знання у сфері електронного документообігу, касового обладнання та інтеграції різноманітних систем обліку.

Рис 1.1 Основні послуги компанії

Центр бізнес-обслуговування "СкайТек" – це компанія, що стрімко розвивається, активно розширюючи свою діяльність у декількох ключових секторах ІТ-індустрії. Вона поєднує в собі різноманітні напрями роботи: від технічного обслуговування комп'ютерів до розробки складних програмних продуктів та реалізації інноваційних рішень для автоматизації бізнес-процесів. Це робить "СкайТек" не просто сервісною компанією, а й надійним ІТ-партнером для малого, середнього та великого бізнесу. Унікальність полягає в наданні повного спектру послуг: від первинної консультації та аналізу потреб клієнта до розробки, впровадження та технічної підтримки ІТ-продуктів або рішень.

Для студентів, які проходять практику в "СкайТек", фірма створює сприятливі умови для отримання практичного досвіду у професійній сфері. Практиканти мають змогу не лише ознайомитися з реаліями корпоративної інфраструктури, але й брати безпосередню участь у вирішенні практичних завдань: налаштуванні та обслуговуванні техніки, роботі над проєктами з розробки веб-сайтів, створенні або тестуванні програмного забезпечення, консультуванні клієнтів, інтеграції облікових систем та багато іншого. Цей досвід дозволяє краще зрозуміти специфіку роботи в ІТ, вдосконалити професійні навички, покращити комунікативні здібності та освоїти практичні інструменти, що використовуються в сучасному ІТ-середовищі.

Багатопрофільність компанії та її орієнтація на реальні бізнес-потреби дають практикантам можливість працювати з актуальними проєктами, вирішувати нестандартні завдання, аналізувати функціонування сучасних систем автоматизації, працювати в команді над проєктами, а також бачити весь процес роботи з клієнтом – від першого звернення до впровадження рішення. Це дозволяє сформувати комплексне розуміння роботи в інформаційних технологіях, покращити професійну підготовку та підвищити конкурентоспроможність на ринку праці після завершення навчання.

# РОЗДІЛ 2. ТЕОРЕТИЧНІ ЗАВДАННЯ

## 2.1 Поняття штучних нейронних мереж

Штучна нейронна мережа (ШНМ) - це математична конструкція, що копіює принципи роботи біологічного мозку. Вона базується на великій кількості взаємопов'язаних вузлів, так званих "нейронів", які організовані у шари: вхідний, один чи декілька прихованих та вихідний. Така мережа здатна вчитися на прикладах, виявляючи закономірності у даних, та згодом робити прогнози або класифікувати нову інформацію.

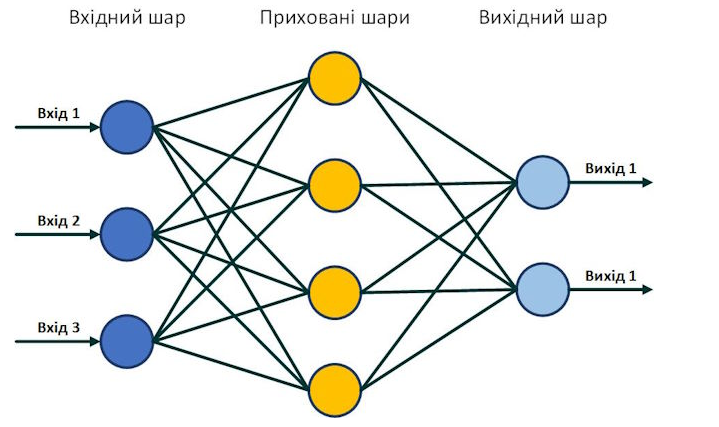
Загалом, структура ШНМ має високу гнучкість і може бути пристосована для різноманітних задач - від розпізнавання зображень до передбачення часових рядів. Її ефективність значною мірою залежить від правильно обраної архітектури та методу навчання. Основними компонентами ШНМ є нейрони, вагові коефіцієнти, функція активації та алгоритм оптимізації. Кожен нейрон отримує сигнали від попередніх шарів, зважує їх, підсумовує та передає результат далі.

Рис. 2.1. Структура найпростішої нейронної мережі

На Рис. 2.1 представлено одношарову нейронну мережу, яка складається з одного входу, одного прихованого шару та одного виходу. Кожен зв'язок має вагу, яка змінюється в процесі навчання.

Навчання ШНМ є процесом налаштування ваг, у ході якого помилка між фактичним та очікуваним результатом мінімізується. Це може досягатися за допомогою різних методів – найпоширенішими з них є градієнтний спуск та його варіації. Під час зворотного поширення помилки (backpropagation) мережа "навчається", змінюючи ваги у напрямку зменшення помилки.

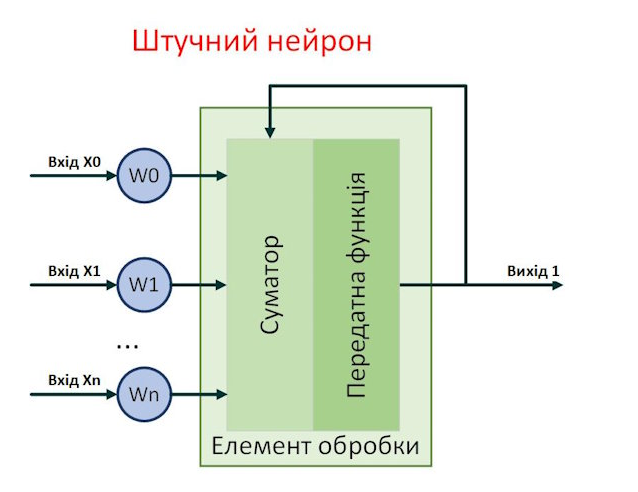
На Рис. 2.2 продемонстровано, як нейронна мережа обчислює похибку, а потім повертається шарами, змінюючи ваги для покращення точності.

Рис. 2.2 – Схематичне зображення процесу навчання мережі за допомогою зворотного поширення помилки

Ключовим аспектом при побудові ефективної моделі є вибір архітектури нейромережі. Просте збільшення кількості шарів або нейронів не завжди приносить кращі результати – важливо уникати як недонавчання (underfitting), так і перенавчання (overfitting). Для цього використовуються спеціальні техніки, серед яких регуляризація, дропаут або крос-валідація.

ШНМ активно застосовуються у задачах прогнозування цін через їх здатність розпізнавати складні нелінійні залежності в даних. Особливо ефективними є такі модифікації як рекурентні нейронні мережі (RNN), довготривала короткострокова пам'ять (LSTM) та трансформери.

Отже, штучні нейронні мережі – потужний інструмент в аналітиці та прогнозуванні. У наступних розділах буде розглянуто класифікацію типів ШНМ, методи їх навчання, а також їх застосування у задачах прогнозування економічних показників.

## 2.2 Основи машинного навчання як фундаменту для нейромереж

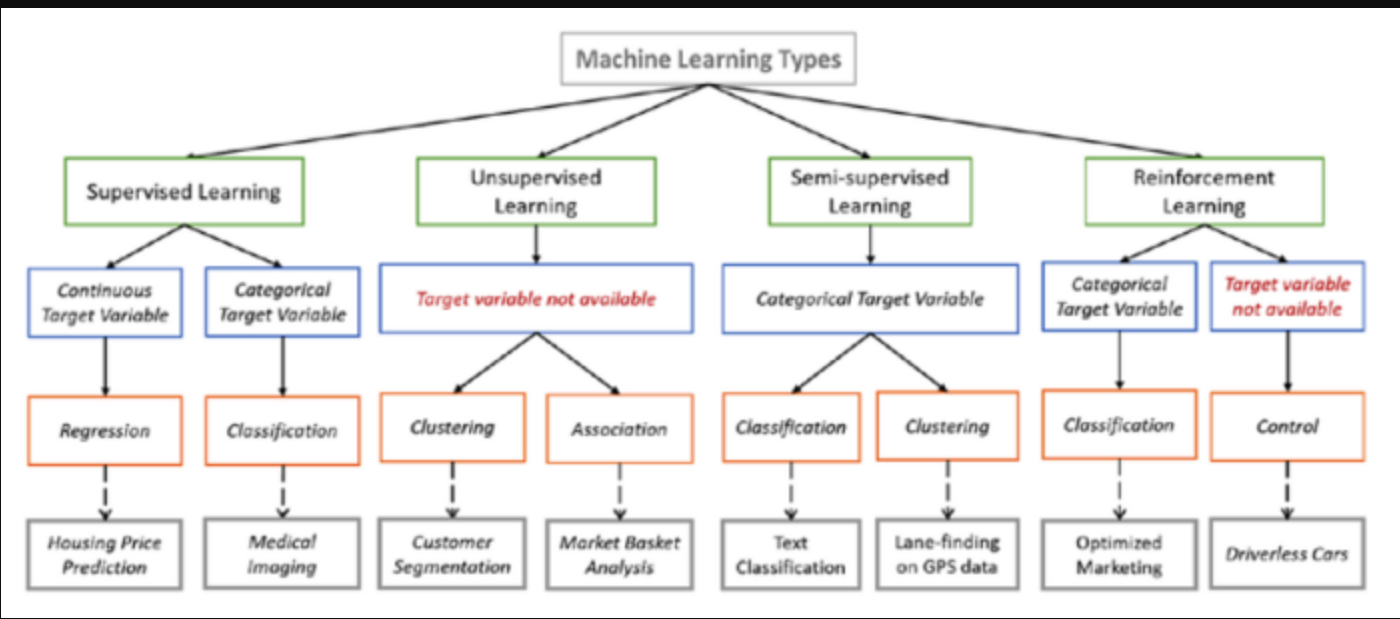
Машинне навчання (ML) є невід’ємною складовою штучного інтелекту та одночасно основою для побудови та навчання штучних нейронних мереж. Цей підхід дозволяє алгоритмам «вчитися» на основі даних, без явного програмування кожного кроку розв’язання задачі. У загальному сенсі, машинне навчання – це здатність комп’ютерних систем знаходити закономірності у даних і використовувати ці закономірності для прийняття рішень, прогнозів чи класифікацій.

машинне навчання визначається як: Метод аналізу даних, який автоматично будує аналітичну модель, використовуючи алгоритми, що ітеративно навчаються на даних. Це дозволяє комп’ютерам знаходити приховані інсайти без явного програмування, де шукати.

Це визначення добре підкреслює ключову перевагу ML – здатність до самостійного вдосконалення на основі досвіду.

Класифікація видів машинного навчання агалом машинне навчання поділяється на три основні типи:

* **Навчання з учителем (Supervised Learning)**
* **Навчання без учителя (Unsupervised Learning)**
* **Підкріплювальне навчання (Reinforcement Learning)**

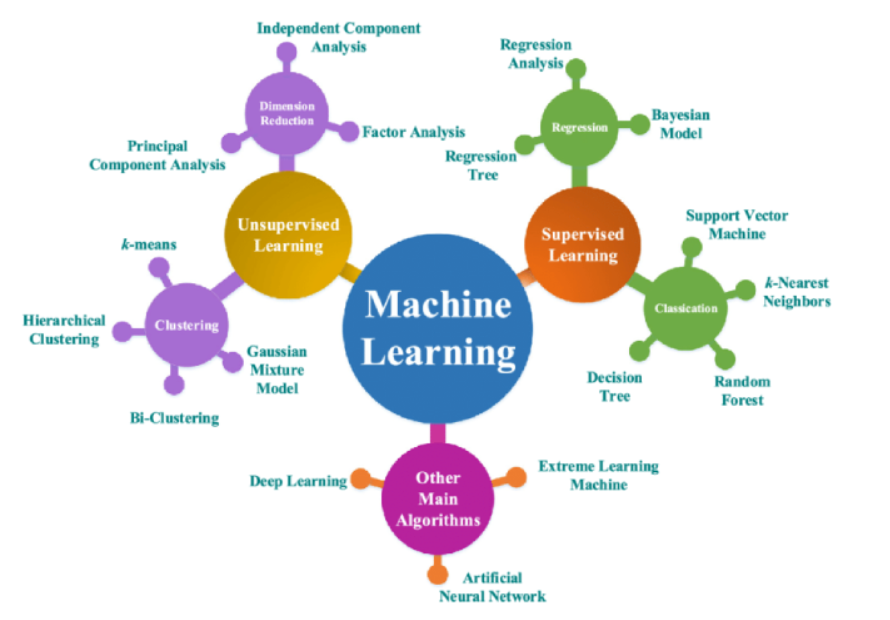
На **Рис. 2.3** зображено порівняння основних підходів ML. У таблиці наведено їхні характеристики, застосування та приклади.

**Рис. 2.3 – Схема типів машинного навчання**

Навчання з учителем. Це найбільш поширений тип ML, при якому алгоритм навчається на заздалегідь маркованих даних. Іншими словами, кожному прикладу у вхідному наборі дано правильну відповідь. Система вивчає зв’язки між вхідними даними та виходами, щоб у майбутньому застосовувати ці зв’язки до нових випадків.

**Приклади задач:**

* Класифікація електронних листів як «спам» або «не спам»
* Прогнозування вартості нерухомості за площею, розташуванням тощо

На **Рис. 2.4** показано, як система використовує позначені дані для побудови моделі.

**Рис. 2.4 – Схема роботи алгоритму з учителем**

Навчання без наглядача (Unsupervised learning). Це різновид машинного навчання, де алгоритм опановує інформацію на основі масиву даних без маркування. Основна ціль алгоритму – самостійно виявити шаблони й віднайти структури в даних, згрупувати об'єкти за їхньою подібністю та, можливо, передбачити майбутні тенденції.

Навчання без наглядача часто використовується для розподілу даних на кластери за спільними характеристиками (кластеризація), скорочення кількості властивостей даних без втрати значущої інформації (зменшення розмірності) та виявлення нетипових даних або відхилень від загальної тенденції (виявлення аномалій).

До алгоритмів навчання без наглядача зараховують:

* Метод k-середніх - алгоритм кластеризації, який розділяє дані на наперед задану кількість груп.
* Алгоритм наближених k-середніх. Більш оптимізована версія попереднього алгоритму, призначена для обробки великих обсягів даних.
* Аналіз головних компонент. Метод зменшення розмірності, що відбирає найважливіші атрибути з масивів даних.

Також можливе поєднання двох вищезгаданих методів (з учителем та без нього). Такий гібрид, умовно, може бути позначений як "напівнаглядне навчання" (або змішане навчання).

У цьому випадку алгоритм вивчає дані, що містять як розмічені, так і не розмічені властивості. Позначені приклади служать для тренування алгоритму, як у випадку навчання з учителем, тоді як не розмічені використовуються для пошуку закономірностей у даних, подібно до навчання без наглядача.

"Напівнаглядне навчання" може показати високу ефективність у випадках, коли наявний незначний набір розмічених даних та великий обсяг не розмічених. Наприклад, під час самонавчання або взаємодії ML-моделей.

**Навчання з підкріпленням (Reinforcement Learning, RL)** – це потужна галузь машинного навчання, де інтелектуальний агент навчається ухвалювати оптимальні рішення шляхом безпосередньої взаємодії з динамічним **довкіллям**. Уявіть собі дресирування домашнього улюбленця: правильна поведінка заохочується ласощами (позитивне підкріплення), а небажана – ігнорується або викликає несхвалення (негативне підкріплення чи його відсутність). Подібним чином, в RL агент виконує певні **дії** у відповідь на поточний **стан** довкілля і отримує чисельний сигнал – **"винагороду"** або **"штраф"**. Головна мета агента – не просто отримати миттєву винагороду, а розробити таку **стратегію (політику)** дій, яка максимізує сукупну, довгострокову винагороду. Цей процес відбувається методом проб і помилок, де агент поступово "розуміє", які дії призводять до кращих результатів у різних ситуаціях. На відміну від інших парадигм машинного навчання, таких як навчання з учителем, де алгоритм отримує готові пари "вхід-правильний вихід", або навчання без учителя, де алгоритм шукає приховані структури в нерозмічених даних, RL агент вчиться самостійно, спираючись виключно на зворотний зв'язок від своїх дій.

Ключові переваги навчання з підкріпленням:

Навчання з підкріпленням пропонує низку унікальних переваг, що роблять його незамінним інструментом для вирішення широкого кола завдань:

* **Здатність до самонавчання та автономність:** Мабуть, найважливішою перевагою є те, що RL-моделі не потребують величезних, ретельно розмічених наборів даних. Агент вчиться, досліджуючи довкілля та отримуючи зворотний зв'язок. Це особливо цінно в ситуаціях, де збір або розмітка даних є надто дорогим, трудомістким або просто неможливим. **Наприклад**, ігровий ШІ може зіграти мільйони партій проти самого себе, поступово вдосконалюючи свою стратегію, не потребуючи даних про ігри людей-чемпіонів (хоча такі дані можуть прискорити навчання). Ще один приклад – робот, що вчиться ходити: він може починати з хаотичних рухів, але з часом, отримуючи позитивну винагороду за кожен успішний крок і негативну за падіння, він самостійно оптимізує свою ходу.
* **Ефективність у надзвичайно складних задачах:** RL демонструє вражаючі результати в задачах, де простір можливих рішень є гігантським, а оптимальну стратегію складно або й неможливо формалізувати за допомогою традиційних алгоритмів. Це стосується як ігрових середовищ, так і реальних застосунків. **Наприклад**, система AlphaGo від DeepMind, навчена за допомогою RL, змогла перемогти найкращих у світі гравців у Го – гру з астрономічною кількістю можливих ходів. Іншим прикладом є оптимізація роботи центрів обробки даних, де RL-агенти можуть керувати системами охолодження для зменшення енергоспоживання, враховуючи безліч динамічних факторів. Також RL використовується в робототехніці для навчання маніпуляторів складним діям, таким як збирання об'єктів або виконання точних операцій.
* **Адаптивність та гнучкість до змін:** RL-алгоритми здатні динамічно адаптуватися до змін у довкіллі або в самій задачі. Якщо умови змінюються, агент може скоригувати свою стратегію на основі нового досвіду, щоб продовжувати досягати максимальної винагороди. Це робить їх ідеальними для систем, що функціонують у непередбачуваних або еволюціонуючих середовищах. **Наприклад**, система управління трафіком на основі RL може адаптуватися до змін у потоках автомобілів протягом дня або до непередбачених подій, таких як дорожні роботи чи аварії, оптимізуючи роботу світлофорів для мінімізації заторів. Персоналізовані системи рекомендацій також можуть використовувати RL, щоб адаптуватися до мінливих інтересів користувача в реальному часі, пропонуючи більш релевантний контент.

Таким чином, навчання з підкріпленням відкриває шлях до створення по-справжньому інтелектуальних систем, здатних навчатися, адаптуватися та вирішувати завдання, які раніше вважалися прерогативою людського інтелекту. Його застосування охоплює все ширший спектр галузей, від ігор та робототехніки до фінансів, медицини та управління складними системами.

Етапи побудови моделі машинного навчання

Якщо узагальнити процес побудови системи машинного навчання, він складається з наступних етапів:

* **Збір даних** – отримання максимально повного та репрезентативного набору даних.
* **Попередня обробка** – очищення, нормалізація, кодування.
* **Вибір моделі** – визначення алгоритму навчання (дерева рішень, нейронні мережі тощо).
* **Навчання моделі** – власне побудова зв’язків.
* **Оцінка точності** – перевірка на тестових даних.
* **Тестування та впровадження** – реальна перевірка роботи системи.

## 2.3 Типи архітектур нейронних мереж у задачах прогнозування

Багатошаровий перцептрон (MLP, Multi-Layer Perceptron) є однією з найпоширеніших архітектур нейронних мереж, яка використовується в задачах класифікації, регресії та прогнозування. Основна ідея цієї моделі полягає у використанні декількох шарів нейронів, кожен з яких поєднаний із попереднім та наступним шарами через вагові коефіцієнти.

На відміну від простого перцептрона, що складається лише з вхідного та вихідного шарів, MLP має принаймні один прихований шар. Кожен нейрон у прихованому шарі застосовує активаційну функцію до зваженої суми вхідних сигналів. Це дозволяє мережі виявляти складні нелінійні залежності між вхідними і вихідними даними.

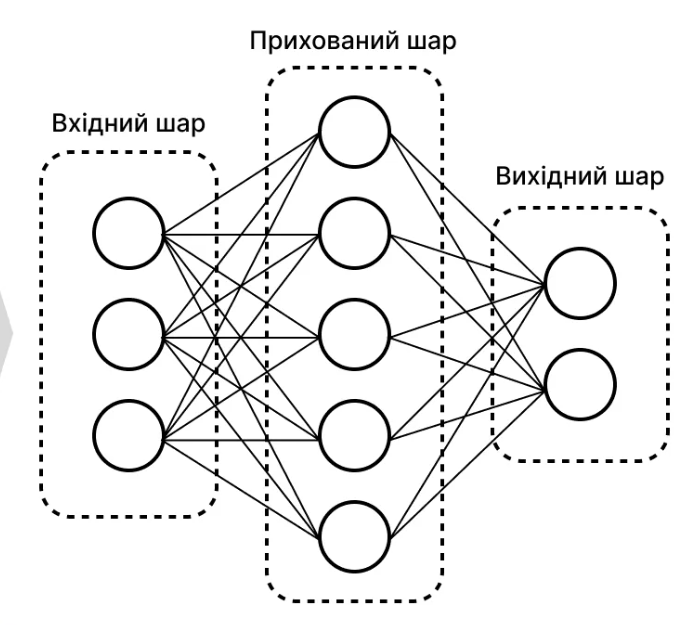
MLP працює за наступним принципом: вхідні дані подаються на перший шар, далі вони проходять через приховані шари, кожен з яких трансформує їх за допомогою активаційних функцій, і, врешті-решт, формуються результати на вихідному шарі.

Рис. 2.5 – Схематичне зображення архітектури MLP з вхідним, прихованим і вихідним шарами.

Основні переваги MLP – це гнучкість у моделюванні даних, здатність до апроксимації довільних функцій, а також відносна простота реалізації.

MLP активно використовується у фінансовому секторі для прогнозування змін курсу валют, у медицині – для класифікації результатів діагностики, у промисловості – для прогнозування зносу деталей або якості продукції.

LSTM – Довготривала короткострокова пам’ять

Модель LSTM (Long Short-Term Memory) є різновидом рекурентних нейронних мереж (RNN), що спеціально розроблена для обробки та аналізу послідовностей даних, які мають довгострокові залежності. Звичайні RNN мають проблему з «забуванням» інформації, що була отримана багато кроків тому, а LSTM вирішує цю проблему за рахунок особливої структури своїх комірок пам’яті.

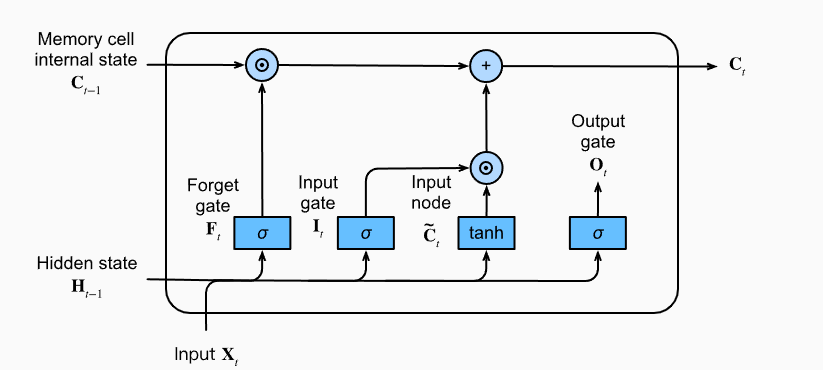
Кожен блок LSTM містить три головні компоненти: вхідний, вихідний і забувальний шлюзи. Ці шлюзи керують тим, яка інформація буде збережена, яка – передана далі, а яка – забута. Таким чином, модель має змогу ефективно зберігати важливу інформацію протягом тривалого часу і водночас відсіювати зайве.

Рис. 2.6 Схематичне зображення роботи LSTM моделі

LSTM широко застосовується в обробці природної мови, зокрема для перекладу текстів, генерації мови, аналізу емоцій, а також у задачах прогнозування часових рядів – таких як прогноз попиту, цін або погодних умов.

Завдяки здатності працювати з тривалими послідовностями, LSTM значно покращує точність моделей, що мають справу зі складними часовими структурами, де врахування далекого контексту критично важливе.

Згорткова нейронна мережа (CNN)

Згорткові нейронні мережі (CNN, Convolutional Neural Networks) є архітектурою, що спеціально пристосована для обробки даних у вигляді сіток, наприклад зображень або багатовимірних тензорів. Основною операцією в CNN є згортка, яка дозволяє автоматично виявляти локальні ознаки – краї, текстури, форми тощо.

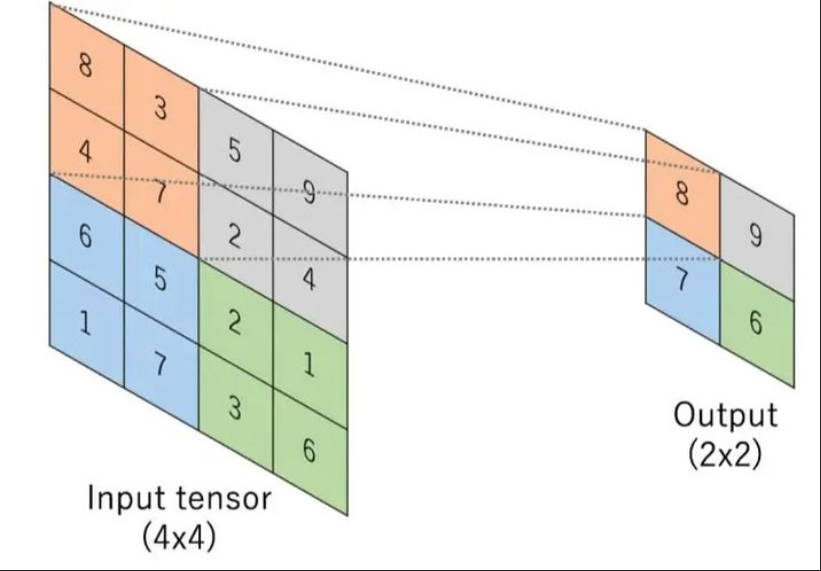
CNN складається з декількох згорткових шарів, шарів підвибірки (pooling) та повнозв’язних шарів. Кожен згортковий шар застосовує фільтри (ядра), що проходять по входу і створюють нові карти ознак, які потім узагальнюються на наступних шарах. Це дає змогу мережі навчатися складним ієрархічним представленням даних.

Рис. 2.7 Схема згортки та роботи з тензорами

У задачах прогнозування CNN може бути використана для обробки структурованих часових рядів, наприклад для виявлення шаблонів, що повторюються у фінансових даних або сенсорних записах. Особливо ефективна CNN у комбінації з іншими архітектурами, наприклад з LSTM.

Transformer

Архітектура Transformer є однією з найважливіших інновацій у сфері глибокого навчання останнього десятиліття. Ця модель була вперше представлена у 2017 році в роботі "Attention is All You Need" і з того часу стала основою для створення сучасних моделей, таких як BERT, GPT, T5 та багатьох інших.

Головною відмінністю Transformer від попередніх моделей є використання механізму self-attention (самоуваги), який дозволяє враховувати вагомість усіх елементів послідовності один щодо одного, незалежно від їхнього порядку. Це дозволяє моделі ефективно захоплювати контекст і взаємозв’язки між словами або значеннями в часових рядах.

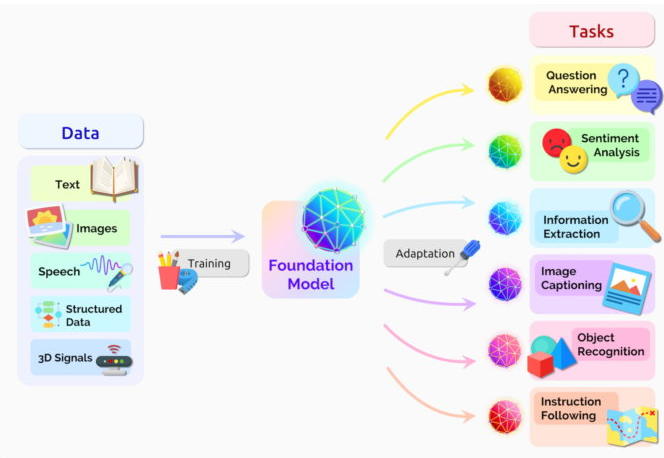
У трансформерах відсутня рекурентність, що значно пришвидшує процес навчання та робить моделі добре масштабованими. Вони демонструють високу продуктивність у перекладі текстів, генерації мови, класифікації документів, а також дедалі частіше застосовуються для прогнозування часових рядів.

Рисунок 2.3 – Зображення можливостей моделей

XGBoost

Модель XGBoost (Extreme Gradient Boosting) – це високоефективна реалізація алгоритму градієнтного бустінгу дерев. Хоча ця модель не є нейронною мережею, її часто використовують у парі з нейромережами як еталонну модель через її точність, швидкість навчання і здатність працювати з розрідженими даними.

XGBoost будує ансамбль рішень, що складається з багатьох слабких моделей (зазвичай, дерев рішень), які поступово навчаються на помилках попередніх моделей. Це дозволяє моделі точно виявляти залежності у даних, в тому числі нелінійні шаблони.

Відмінністю XGBoost є вбудована підтримка регуляризації, що допомагає уникнути перенавчання, а також оптимізація обчислень для роботи з великими обсягами даних. Модель ефективно працює на табличних даних і часто використовується у фінансовому прогнозуванні, аналізі даних, медицині та інших сферах.

# РОЗДІЛ 3. ПРАКТИЧНЕ ЗАВДАННЯ

## 3.1. Вибір програмних засобів

Одним із ключових етапів у реалізації дипломного проєкту є визначення програмного середовища, у якому здійснюється розробка, тестування та аналіз моделей. Зважаючи на зручність, гнучкість та підтримку численних розширень, для реалізації обчислень та написання коду було обрано середовище Visual Studio Code (VS Code).

Visual Studio Code — це сучасний редактор вихідного коду, створений компанією Microsoft для операційних систем Windows, macOS та Linux. Він має відкритий вихідний код і доступний безкоштовно, що робить його привабливим вибором для студентів і розробників. Середовище підтримує велику кількість мов програмування, включаючи Python — основну мову, використану в цьому проєкті. Редактор забезпечує підсвічування синтаксису, автодоповнення коду (завдяки IntelliSense), інтеграцію з Git, налагоджувач, можливість роботи з віртуальними середовищами та зручний інтерфейс.

На відміну від традиційних IDE, VS Code дозволяє швидко створити робоче середовище за допомогою відкриття потрібної папки з файлами, а також легко адаптується до завдань будь-якої складності через розширення. Для роботи з Python використовувалось офіційне розширення Microsoft Python Extension, а також інші додаткові модулі, що полегшують аналіз даних і візуалізацію.

На рисунку 3.1 зображено інтерфейс середовища Visual Studio Code, що використовувався під час розробки проєкту.

*(Рисунок 3.1 – Інтерфейс середовища Visual Studio Code)*

Основною мовою програмування, обраною для реалізації даного дослідження, є Python. Ця мова добре підходить для реалізації моделей машинного навчання та нейронних мереж завдяки наявності великої кількості спеціалізованих бібліотек, зокрема NumPy, Pandas, Matplotlib, PyTorch, scikit-learn та інших. Python має читабельний синтаксис, активну спільноту, велику кількість навчальних ресурсів, а також потужну підтримку для інтеграції з іншими інструментами.

Python є інтерпретованою мовою, що означає виконання коду відбувається безпосередньо під час його запуску, без попередньої компіляції. Це прискорює цикл розробки та полегшує налагодження.

Таким чином, комбінація Visual Studio Code як середовища розробки та Python як основної мови програмування створює оптимальні умови для ефективної реалізації задач дослідження методів навчання нейронних мереж у задачах прогнозування часових рядів.

## 3.2. Оцінка основних бібліотек та фреймворків

Основною бібліотекою для реалізації дипломного проєкту є ReactJS, оскільки це продукт від компанії Facebook, який суттєво полегшує процес конструювання та керування елементами DOM. Спочатку цю бібліотеку створили та використовували винятково для внутрішніх потреб компанії (наприклад, для розробки їх соціальної мережі). З часом вона стала доступною широкому колу розробників.

У своєму класичному вигляді React є всього лиш бібліотекою для створення нескладних клієнтських застосунків, навіть не лише для односторінкових застосунків (Single Page Application), бо вона не має вбудованих механізмів маршрутизації (Routing) та інших додаткових можливостей. Нещодавно React отримав офіційну можливість використовувати Context API для реалізації доволі простого управління станом (state management) [27].

Головні переваги *ReactJS* [27]:

1. Синтаксис React надзвичайно зрозумілий для розробників, які вже знайомі з JavaScript та HTML.
2. Розробка у React підтримує функціональний підхід до написання компонентів.
3. Важливим принципом, що активно використовується у React, є Immutability - заборона безпосередньої зміни стану або даних у компонентах, а замість цього - створення нових екземплярів на основі вже наявних [27].
4. Підхід до написання додатків у React є компоненто-орієнтованим, де кожен елемент – це окремий компонент.
5. Підхід до створення екосистеми додатків у React полягає у поступовому додаванні необхідних бібліотек і пакетів.
6. React зберігає та опрацьовує стан DOM у віртуальному DOM, що робить його швидким при відображенні [27].
7. Розробка у React супроводжується великою кількістю інформації про помилки та ключові проблеми.
8. У React можна інтегрувати будь-яку JavaScript-бібліотеку разом із великою кількістю спеціалізованих бібліотек.
9. React має документацію з багатьма прикладами та детальним описом, включно з українською версією.
10. У React існує активна спільнота розробників.
11. Є багато інструментів, що спрощують процес роботи та відлагодження у React*.*
12. React регулярно оновлюється та удосконалюється.
13. React підтримується Facebook [27].

Недоліки бібліотеки *ReactJS*:

1. *React* "з коробки" не містить практично жодних додаткових можливостей, крім роботи з *DOM*, важливо мати знання або вміти ефективно використовувати пошук потрібних та надійних бібліотек для вирішення різних завдань [27].
2. У простоті React може критися його головний недолік – компонування, структурування файлів у React-додатках, масштабування додатку загалом та інших нюансів, є ризик неправильної реалізації. Це може призвести до складної структури додатку з плином часу (і збільшенням обсягу коду), що вимагатиме багато зусиль для підтримки та подальшого розширення [27].

Для управління станом програми застосовується бібліотека Redux Toolkit.

Redux Toolkit – це офіційний набір інструментів, що допомагає спростити розробку з використанням Redux, надаючи додаткові інструменти та утиліти. Його створили з метою зробити роботу з Redux простішою, продуктивнішою та ефективнішою.

Основні риси та переваги Redux Toolkit:

1. Спрощений синтаксис: Redux Toolkit надає спеціальні функції, як-от createSlice та createAsyncThunk, що допомагають зменшити обсяг коду та полегшити створення reducer.
2. Автоматичне керування станом: Redux Toolkit автоматично генерує ідентифікатори дій та створює редьюсери з вбудованою обробкою сполучень (reducers), що дозволяє зосередитись на логіці додатку, а не на деталях редукторів.
3. Вбудована підтримка асинхронних операцій: Redux Toolkit має вбудовану підтримку асинхронних операцій через функцію createAsyncThunk, що дозволяє виконувати асинхронні запити та опрацьовувати результати [3].
4. Для реалізації маршрутизації в проекті використовується бібліотека React Router. Вона дає змогу керувати навігацією в додатку, визначати, які компоненти мають відображатися під час зміни URL, та робить розробку багатосторінкових додатків в React простішою.

Головні причини використання React Router:

1. Маршрутизація: React Router дає змогу визначити набір маршрутів (routes), пов'язаних з певними компонентами.
2. Навігація: React Router надає компоненти, як-от Link та NavLink, які дають змогу створювати посилання, що відповідають маршрутам в додатку.
3. React Router підтримує параметризовані маршрути, що дають змогу передавати динамічні дані у шляху.
4. Вкладені маршрути: React Router дає змогу створювати вкладені маршрути, коли компоненти можуть мати власну маршрутизацію [6].

При створенні серверної частини використано фреймворк NestJS. Він пропонує розширені можливості для побудови масштабованих та ефективних серверних додатків.

Основні переваги використання NestJS полягають у модульному підході для розробки, що дає змогу розбити функціонал на окремі модулі. Також фреймворк пропонує однорідний стиль коду, що сприяє кращій зрозумілості та співпраці між розробниками. Ще він побудований на основі TypeScript, що дозволяє використовувати строгу типізацію. NestJS має велику кількість розширень, які дозволяють розширювати його функціональність та додавати підтримку різних інструментів та технологій.

# ВИСНОВКИ

Під час проходження проектно-технологічної (виробничої) практики було реалізовано теоретичні та практичне завдання. Було проведено ознайомлення з історією та діяльністю ТОВ «Петтерсонапс», досліджено впроваджені компанією підходи до розробки програмного забезпечення для різних систем, у тому числі для фреймворку ReactJS.

При розробці програмного забезпечення для навчання "сліпого" друку, інтерфейс користувача було створено з використанням ReactJS. Це включало створення складових частин, управління станом застосунку, реалізацію взаємодії з користувачем та інші функції.

Проведено аналіз та порівняння аналогів, визначено переваги та недоліки клавіатурних тренажерів для розробників. Розглянуто існуючі підходи до розв'язання проблеми.

Вибрано програмні засоби для реалізації веб-додатку на мові TypeScript, а саме: React Router, Redux, NestJS, Axios, TailwindCSS, Passport-JWT, Mongoose, та базу даних MongoDB. Також створено Google client ID для реалізації Google авторизації. Архітектура проекту поділяється на серверну та клієнтську частини. У клієнтській частині розміщені основні компоненти, такі як кнопки, поля для введення даних, заголовок застосунку, головна сторінка, сторінка акаунту, профілю, лекцій та вправ. Для серверної частини розроблено компоненти users, auth, attempts, lessons, sections.

Розроблено основні модулі проєкту, такі як перевірка введених даних з клавіатури, функція лічильника часу, розрахунок швидкості та точності набору тексту, реалізація Google авторизації, створення секцій з мовами програмування, створення облікового запису користувача, відображення повної статистики на сторінці профілю, а також реалізація різних рівнів складності вправ. Створено покрокову інструкцію для користувача.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ ТА ДЖЕРЕЛ

1. Штучна нейронна мережа [Електронний ресурс] – Режим доступу: [https://uk.wikipedia.org/wiki/Штучна\_нейронна\_мережа](https://uk.wikipedia.org/wiki/%D0%A8%D1%82%D1%83%D1%87%D0%BD%D0%B0_%D0%BD%D0%B5%D0%B9%D1%80%D0%BE%D0%BD%D0%BD%D0%B0_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D0%B0) – Назва з екрана. – Дата звернення: 27.05.2025.
2. SkyTek – центр бізнес обслуговування [Електронний ресурс] – Режим доступу: <https://skytek.in.ua/> – Назва з екрана. – Дата звернення: 27.05.2025.
3. Штучні нейронні мережі – вступ та основи [Електронний ресурс] – Режим доступу: <https://itmaster.biz.ua/programming/vision/neural-networks.html> – Назва з екрана. – Дата звернення: 27.05.2025.
4. The Transmitted. Що таке MLP у машинному навчанні? [Електронний ресурс]. – Режим доступу: <https://thetransmitted.com/adlucem/shho-take-mlp-u-mashynnomu-navchanni/> – Назва з екрана.
5. IT Wiki. Long Short-Term Memory (LSTM) [Електронний ресурс]. – Режим доступу: <https://itwiki.dev/data-science/ml-reference/ml-glossary/long-short-term-memory-lstm> – Назва з екрана. Дата звернення: 27.05.2025.
6. Speka. Як працює згорткова нейронна мережа: просте пояснення [Електронний ресурс]. – Режим доступу: <https://speka.media/yak-pracyuje-zgortkova-neironna-mereza-proste-poyasnennya-9er7j1> – Назва з екрана. Дата звернення: 27.05.2025.
7. NVIDIA Blog. What is a Transformer Model? [Електронний ресурс]. – Режим доступу: <https://blogs.nvidia.com/blog/what-is-a-transformer-model/> – Назва з екрана. Дата звернення: 27.05.2025.
8. IT Wiki. XGBoost [Електронний ресурс]. – Режим доступу: <https://itwiki.dev/data-science/ml-reference/ml-glossary/xgboost> – Назва з екрана. Дата звернення: 27.05.2025.

# ДОДАТКИ