АНОТАЦІЯ

Інформаційна система для прогнозування вартості житла на основі послідовної моделі машинного навчання з веб-інтерфейсом // Дипломна робота освітнього рівня "Магістр" // Садівник Максим Петрович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп’ютерних наук, група СНм-61 // Тернопіль, 2019 // С. 114, рис. – 32, табл. – 6, додат. – 1, бібліогр. – 50.

Ключові слова: МАШИННЕ НАВЧАННЯ, ШТУЧНИЙ ІНТЕЛЕКТ, БРАУЗЕР, МОДЕЛІ МАШИННОГО НАВЧАННЯ, БІБЛІОТЕКА.

Дана робота присвячена розробці інформаційної системи для прогнозування вартості житла на основі послідовної моделі машинного навчання. Метою роботи є демонстрація головних особливостей машинного навчання у браузері використовуючи мову програмування JavaScript.

В роботі розглянуто основні моделі машинного навчання, проведено їх аналіз та порівняння. На основі аналізу існуючих бібліотек було проведено їх порівняння та вибір бібліотеки для використання. Було розроблено застосунок для демонтстрації роботи обраної бібліотеки.

ABSTRACT

Information system for apartments cost forecasting based on sequential model of web-interface machine study // // Master's Thesis // Sadivnyk Maksym Petrovych // Ternopil I.Pulyu National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, group SNm-61 // Ternopil, 2019 // Pages – 114, drawing – 32 , tables – 6, applications – 1, bibliographic sources – 50 .

KEYWORDS: MACHINE LEARNING, ARTIFICAL INTELLIGENCE, BROWSER, MACHINE LEARNING MODELS, LIBRARY.

This thesis is devoted to the development of information system for apartments cost forecasting based on sequential model of machine learning with web-interface. The aim is to demonstrate the main features of machine learning in browser using programming language JavaScript.

The paper describes the main models of machine learning, was provided their analysis and compare. Based on an analysis of existing libraries was provided their compare and chose of library for use. Developed application for demonstration of chosen library’s work.

ЗМІСТ

Вступ 9

Перелік умовних скорочень 11

1 Штучний інтелект та його роль у світі 12

1.1Поняття «тшучного інтелекту» 12

1.2 Історія розвитку штучного інтелекту 18

1.3 Штучний інтелект в сучасному світі 24

1.4 Висновок до першого розділу 26

2 Машинне навчання та його моделі 27

2.1 Поняття «машинного навчання» 27

2.2 Моделі машинного навчання 36

2.3 Висновок до другого розділу 48

3 Проектування та програмна реалізація інформаційної системи для прогнозування вартості житла на основі послідовної моделі машиного навчання з використанням бібліотеки tensorflow.js 49

3.1 Послідовна модель машинного навчання 49

3.2 Проектування навчальної моделі 55

3.3 Підготовка даних до навчання 61

3.4 Підготовка моделі до навчання та прогнозування 64

3.5 Висновок до третього розділу 68

4 Машинне навчання у браузері. огляд tensorflow.js та інших популярних бібліотек браузерного машинного навчання та обґрунтування вибору 69

4.1 Машинне навчання у браузері 69

4.2 Порівняння TensorFlow.js з іншими бібліотеками браузерного машинного навчання та обґрунтування вибору 70

4.3 Висновок до четвертого розділу 76
5 Обґрунтування економічної ефективності 77
5.1 Розразунок норм часу на виконання науково-дослідницької роботи 77
5.2 Визначення витрат на оплату праці та відрахувань на соціаьні заходи .. 78
5.3 Розразунок матеріальних витрат 81
5.4 Розрахунок витрат на електроенергію 82
5.5 Розрахунок суми амортизаційних відрахувань 83
5.6 Обчислення накладних витрат 84
5.7 Складання кошторису витрат та визначення собівартості роботи 84
5.8 Розрахунок ціни програмного продукту 85
5.9 Визначення економічної ефективності і терміну окупності капітальних вкладень 86
5.10 Висновок до п’ятого розділу 87
6 Охорона праці та безпека в надзвичайних ситуаціях 89
6.1 Охорона праці 89
6.1.1 Види страхових виплат Фонду соціального страхування від нещасних випадків на виробництві та професійних захворювань, на які може розраховувати працівник у разі його травмування, профзахворювання або смерті 89
6.1.2 Ефективність охорони праці у Великобританії у сфері IT 91
6.2 Безпека в надзвичайних ситуаціях 93
6.2.1 Організація цивільного захисту на об’єктах промисовості та виконання заходів щодо запобігання виникненню надзвичайних ситуацій техногенного походження 93

6.2.2 Особливості роботи та розлади здоров’я користувачів комп’ютерів, що формується під впливом роботи за комп’ютером 97
6.3 Висновок до шостого розділу 101
7 Екологія 102
7.1 Енергозбереження і його роль у вирішенні екологічних проблем 102
7.2 Методологія моделювання екологічних проблем 105
7.3 Висновок до сьомого розділу 107
Висновки 108
Список використаних джерел 110
Додатки

ВСТУП

Штучний інтелект - наука і технологія створення інтелектуальних машин та інтелектуальних комп'ютерних програм. Штучний інтелект можна визначити як наукову дисципліну, яка займається моделюванням розумної поведінки. Це визначення має один істотний недолік - поняття інтелекту важко пояснити. Проблема визначення штучного інтелекту зводиться до проблеми визначення інтелекту в цілому: чи є він чимось єдиним, або ж цей термін об'єднує набір розрізнених здібностей. Штучний інтелект надає засіб і випробувальну модель для теорій інтелекту: ці теорії можуть бути сформульовані мовою комп'ютерних програм, а потім - випробувані. На сьогоднішній день існує безліч алгоритмів і підходів до створення штучного інтелекту. Так, штучний інтелект може бути заснований на використанні Булевої алгебри та обчислення предикатів, в якому вона розширена за рахунок введення предметних символів, відносин між ними, кванторів існування та загальності. Іншим поширеним підходом до побудови штучного інтелекту є структурний підхід, який передбачає моделювання структури людського мозку. Однією з перших таких спроб був перцептрон Френка Розенблатта. Основною модельованою структурною одиницею в перцептронах (як і в більшості інших варіантів моделювання мозку) є нейрон [1]. Досить велике поширення отримав і еволюційний підхід. При побудові систем штучного інтелекту по даному підходу основна увага приділяється побудові початкової моделі, і правилам, за якими вона може змінюватися (еволюціонувати). Причому модель може бути складена різними методами, це може бути і штучна нейронна мережа, і набір логічних правил інші моделі. Надалі на підставі перевірки моделей відбираються кращі з них, на підставі яких за певними правилами генеруються нові моделі, з яких знову вибираються кращі і т. д.

Машинним навчанням називається галузь комп’ютерних наук, яка вивчає методи навчання комп’ютеризованих систем на підставі даних без

програмування їх поведінки. Методи машинного навчання (machine-learning methods) відіграють важливу роль у багатьох аспектах сучасного суспільства: від веб-пошуку до фільтрації контенту в соціальних мережах. Системи на базі методів машинного навчання використовуються в системах машинного зору, для ідентифікації об’єктів на зображеннях, аналізу людської мови і текстів тощо.

Метою даної роботи є розробка інформаційної системи для прогнозування вартості житла на основі послідовної моделі машинного навчання з веб-інтерфейсом.

Для досягнення поставленої мети потрібно вирішити такі завдання:

1. Провести огляд літератури, з обраної тематики;
2. Обрати бібліотеку машинного навчання;
3. Створити прототип майбутньої програми;
4. Розробити додаток згідно прототипу;
5. Розробити інтерфейс додатку;
6. Проаналізувати отромані дані.

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ПЗ	програмне забезпечення
UI (User Interface)	інтерфейс користувача
ШІ	штучний інтелект
МН	машинне навчання
API (Application Programming Interface)	набір визначень взаємодії різнотипного програмного забезпечення

1 ШТУЧНИЙ ІНТЕЛЕКТ ТА ЙОГО РОЛЬ В СУЧАСНОМУ СВІТІ

1.1 Поняття «штучного інтелекту»

Штучний інтелект (ШІ) - це моделювання процесів інтелекту людини за допомогою машин, особливо комп'ютерних систем. Ці процеси включають навчання (здобуття інформації та правила користування інформацією), міркування (використання правил для наближення чи певних висновків) та самокорекцію. Окремі програми ШІ включають експертні системи, розпізнавання мови та машинне бачення [2].

Штучний інтелект можна класифікувати як слабкий або сильний. Слабкий штучний інтелект, також відомий як вузький штучний інтелект, - це система, яка розроблена і підготовлена для певного завдання. Віртуальні особисті помічники, такі як Apple Siri, є формою слабкого штучного інтелекту. Сильний штучний інтелект, також відомий як штучний загальний інтелект, - це система з узагальненими когнітивними здібностями людини. Коли постає незнайоме завдання, сильна система здатна знайти рішення без втручання людини.

Оскільки витрати на апаратне забезпечення, програмне забезпечення та персонал для AI можуть бути дорогими, багато виробників включають компоненти штучного інтелекту у свої стандартні пропозиції, а також доступ до платформ штучного інтелекту як сервісу (AIaaS). Штучний інтелект як послуга дозволяє приватним особам та компаніям експериментувати з штучним інтелектом для різних бізнес-цілей та вибирати кілька платформ. До популярних хмарних пропозицій штучного інтелекту належать послуги Amazon AI, помічник IBM Watson, Microsoft Cognitive Services та служби Google AI [3].

Хоча інструменти штучного інтелекту представляють цілий ряд нових функціональних можливостей для бізнесу, використання штучного інтелекту викликає етичні питання [4]. Це пояснюється тим, що алгоритми глибокого навчання, які лежать в основі багатьох найсучасніших інструментів, є настільки ж розумними, як і дані, які вони отримують під час навчання. Оскільки людина вибирає, які дані слід використовувати для підготовки програми, потенціал людської упередженості притаманний і повинен ретельно контролюватися.

Деякі експерти галузі вважають, що термін штучний інтелект занадто тісно пов'язаний із популярною культурою, що спричиняє нереалістичні побоювання громади щодо штучного інтелекту та неймовірних очікувань щодо того, як він змінить робоче місце та життя загалом. Дослідники та маркетологи сподіваються, що розширений інтелект, який має більш нейтральну конотацію, допоможе людям зрозуміти, що штучний інтелект просто покращить продукти та послуги, а не замінить людей, які ними користуються [5].

Аренд Гінце, доцент кафедри інтегративної біології та інформатики та техніки в Мічиганському державному університеті, класифікує штучний інтелект на чотири типи - від виду систем, які існують сьогодні, до розумних систем, які ще не існують. Його категорії такі [6]:

- Реактивні машини. Приклад - Deep Blue, шахова програма IBM, яка перемогла Гаррі Каспарова в 90-х. Deep Blue може визначити фігури на шаховій дошці та зробити прогнози, але вона не має пам’яті та не може використовувати минулий досвід для інформування майбутніх. Він аналізує можливі рухи - свого та опонента - та вибирає найбільш стратегічний крок. Deep Blue та AlphaGO від Google були розроблені для вузьких цілей і не можуть бути легко застосовані до іншої ситуації. На рисунку 1.1 зображений супер комп‘ютер «Deep Blue» [7].
- Обмежена пам'ять. Ці системи штучного інтелекту можуть використовувати минулий досвід для інформування майбутніх рішень. Деякі

функції прийняття рішень у автомобілях, що керуються самостійно, розроблені таким чином. Спостереження інформують про дії, що відбуваються в не надто віддаленому майбутньому, наприклад, проїзд автомобільних смуг. Ці спостереження не зберігаються постійно.

- Теорія розуму. Цей психологічний термін означає розуміння того, що інші мають свої переконання, бажання та наміри, які впливають на рішення, які вони приймають. Такого роду штучного інтелекту ще не існує.
- Самосвідомість. У цій категорії системи штучного інтелекту мають почуття себе, мають свідомість. Машини з самосвідомістю розуміють свій сучасний стан і можуть використовувати інформацію, щоб зробити висновок про те, що відчувають інші. Цей тип штучного інтелекту ще не існує [8].

Рисунок 1.1 – «Шахматний суперкомп‘ютер «Deep Blue»

Штучний інтелект використовується зараз в різних технологіях. Нижче наведено шість прикладів:

1. Автоматизація: що заставляє систему або процес функціонувати автоматично. Наприклад, автоматизовану робототехнічну процес (RPA) можна запрограмувати на виконання багаторазових повторюваних завдань, які

зазвичай виконують людину. RPA відрізняється від автоматизації IT тим, що може адаптуватися до мінливих обставин.

2. Машинне навчання: наука про те, щоб комп'ютер діяв без програмування. Глибоке навчання - це підмножина машинного навчання, яке, дуже просто кажучи, може розглядатися як автоматизація прогнозної аналітики. Існує три типи алгоритмів машинного навчання:

- Контрольоване навчання: Набори даних маркуються таким чином, щоб шаблони можна було виявити та використовувати для позначення нових наборів даних
- Навчання без нагляду: Набори даних не маркуються та сортуються за подібністю чи відмінностями
- Укріплене навчання: набори даних немарковані, але після виконання дії або декількох дій система ШІ отримує зворотній зв'язок

3. Машинний зір: наука про те, щоб комп'ютери могли бачити. Ця технологія фіксує та аналізує візуальну інформацію за допомогою камери, аналого-цифрового перетворення та цифрової обробки сигналу. Його часто порівнюють із зором людини, але машинний зір не пов'язаний з біологією і може бути запрограмований, щоб побачити крізь стіни, наприклад. Він використовується в ряді застосувань - від ідентифікації підпису до аналізу медичних зображень. Комп'ютерний зір, який зосереджений на машинній обробці зображень, часто пов'язаний з машинним зором.

4. Обробка природних мов (NLP): обробка людської, а не комп'ютерної мови комп'ютерною програмою. Один із найстаріших і найвідоміших прикладів NLP - це виявлення спаму, яке розглядає тему та текст електронного листа та вирішує, чи це непотріб. Сучасні підходи до NLP засновані на машинному навчанні. Завдання НЛП включають переклад тексту, аналіз настроїв та розпізнавання мовлення [9].

5. Робототехніка: інженерна галузь, орієнтована на проектування та виготовлення роботів. Роботи часто використовуються для виконання завдань, які людині важко виконувати або виконувати послідовно. Вони

використовуються на складальних лініях для виробництва автомобілів або NASA для переміщення великих предметів у космосі. Дослідники також використовують машинне навчання для створення роботів, які можуть взаємодіяти в соціальних умовах.

6. Самокеровані автомобілі: вони використовують комбінацію комп'ютерного зору, розпізнавання зображень та глибокого навчання для побудови автоматизованих навичок пілотування транспортного засобу під час перебування на заданій смузі руху та уникнення несподіваних перешкод, таких як пішоходи [10].

Окрім цього, штучний інтелект широко використовується в інших галузях людської діяльності:

- Штучний інтелект в охороні здоров'я. Найбільші ставки - на покращення результатів пацієнтів та зниження витрат. Компанії застосовують машинне навчання для постановки кращих і швидших діагнозів, ніж люди. Однією з найвідоміших технологій охорони здоров’я є IBM Watson. Він розуміє природну мову і здатний відповідати на задані нею питання. Система видобуває дані про пацієнтів та інші доступні джерела даних, щоб сформувати гіпотезу, яка потім представляє схему оцінки достовірності. Інші додатки штучного інтелекту включають в себе чати, комп’ютерну програму, яка використовується в Інтернеті для відповіді на запитання та надання допомоги клієнтам, для планування подальших побачень або надання допомоги пацієнтам через процес виставлення рахунків, і віртуальних фельдшерів, які надають основні медичні відгуки.

- Штучний інтелект в бізнесі. Автоматизація робототехнічних процесів застосовується до сильно повторюваних завдань, які зазвичай виконує людина. Алгоритми машинного навчання інтегруються в аналітичні та CRM-платформи, щоб розкрити інформацію про те, як краще обслуговувати клієнтів. Чат-боти були включені в веб-сайти, щоб забезпечити негайне обслуговування клієнтів. Автоматизація робочих місць також стала точкою розмови серед науковців та IT-аналітиків [11].

- Штучний інтелект в освіті. Штучний інтелект може автоматизувати класифікацію, даючи освітянам більше часу. Штучний інтелект може оцінювати учнів та адаптуватися до їх потреб, допомагаючи їм працювати у власному темпі. Викладачі можуть надати студентам додаткову підтримку, гарантуючи, що вони залишаються на шляху. Штучний інтелект міг змінити, де і як навчаються учні, можливо навіть замінивши деяких викладачів.

- Штучний інтелект у фінансах. ШІ у заявах на особисте фінансування, таких як монетний двір або податок на турбо, порушує фінансові установи. Такі додатки збирають особисті дані та надають фінансові поради. Інші програми, такі як IBM Watson, були застосовані до процесу купівлі будинку. Сьогодні програмне забезпечення здійснює велику частину торгів на Уолл-стріт [12].

- Штучний інтелект в праві. Процес відкриття, просіювання документів, у законі часто є надзвичайно важливим для людей. Автоматизація цього процесу - це більш ефективне використання часу. Стартапи також будують комп'ютерні помічники з питаннями та відповідями, які можуть просіювати відповіді на запитання на програму, вивчаючи таксономію та онтологію, пов’язані з базою даних.

- Штучний інтелект у виробництві. Це область, яка стояла на передньому плані з включення роботів у робочий процес. Промислові роботи використовувались для виконання окремих завдань і були відокремлені від людських робітників, але в міру розвитку технології це змінилося.

При застосуванні штучного інтелекту в області самокерованих автомобілів виникають проблеми безпеки та етичні питання [13]. Автомобілі можуть бути зламані, і коли автономний транспортний засіб бере участь у ДТП, відповідальність не є чіткою. Автономні транспортні засоби також можуть бути поставлені в положення, коли аварія неминуча, змушуючи програму приймати етичне рішення про те, як мінімізувати шкоду.

Ще одна велика стурбованість - потенціал для зловживань інструментами штучного інтелекту. Хакери починають використовувати складні інструменти машинного навчання для отримання доступу до чутливих систем, ускладнюючи питання безпеки поза його сучасним станом.

Засоби поглиблення відео та аудіозапису на основі глибокого навчання також представляють ризик для відомих людей інструментами, які використовуються для створення так званих глибоких фейків, переконливо сфабрикованих відеозаписів громадських діячів, які говорять чи роблять речі, які ніколи не відбувалися.

Незважаючи на ці потенційні ризики, існує мало норм, що регулюють використання інструментів штучного інтелекту, і там, де закони існують, як правило, стосуються штучного інтиелекту лише опосередковано. Наприклад, федеральні правила справедливого кредитування вимагають від фінансових установ пояснювати кредитні рішення потенційним клієнтам, які обмежують ступінь, в якій кредитори можуть використовувати алгоритми глибокого навчання, які за своєю суттю зазвичай непрозорі. Європейський GDPR встановлює суворі обмеження щодо того, як підприємства можуть використовувати дані споживачів, що перешкоджає навчанню та функціональності багатьох застосунків, пов'язаних із споживачами.

У 2016 році Національна рада з питань науки та технологій випустила звіт, в якому вивчала потенційну роль урядового регулювання у розвитку штучного інтелекту , але він не рекомендував розглянути конкретні законодавчі акти. З того часу питання мало приділено уваги з боку законодавців [14].

1.2 Історія розвитку штучного інтелекту

Штучний інтелект (AI) - це молода дисципліна шістдесяти років, що представляє собою сукупність наук, теорій і прийомів (включаючи математичну логіку, статистику, ймовірності, обчислювальну нейробіологію,

інформатику), яка має на меті наслідувати когнітивні здібності людини. Розроблений в диханні Другої світової війни [15], його розробки тісно пов'язані з технологіями обчислювальної техніки та змусили комп'ютери виконувати все більш складні завдання, які раніше могли бути делеговані лише людині.

Однак ця автоматизація залишається далеко не людським інтелектом у строгому розумінні, що робить назву відкритою для критики з боку деяких експертів. Кінцева стадія їх досліджень ("сильний" штучний інтелект, тобто здатність контекстуалізувати дуже різні спеціалізовані проблеми абсолютно автономно) абсолютно не порівнянна з поточними досягненнями ("слабкий" або "сильний" штучний інтелект надзвичайно ефективний в їх навчальному полі). "Сильний" штучний інтелект, який ще лише здійснився в науковій фантастиці, потребує прогресу в фундаментальних дослідженнях (а не лише підвищення ефективності), щоб мати змогу моделювати весь світ.

Однак з 2010 року ця дисципліна пережила новий бум, головним чином завдяки значному вдосконаленню обчислювальної потужності комп'ютерів та доступу до величезної кількості даних. Обіцянки, поновлення та занепокоєння, які часом фантазуються, ускладнюють об'єктивне розуміння явища. Короткі історичні нагадування можуть допомогти розташувати дисципліну та повідомити про поточні дебати. На рисунку 1.2 наведена інфографіка історії розвитку штучного інтелекту.

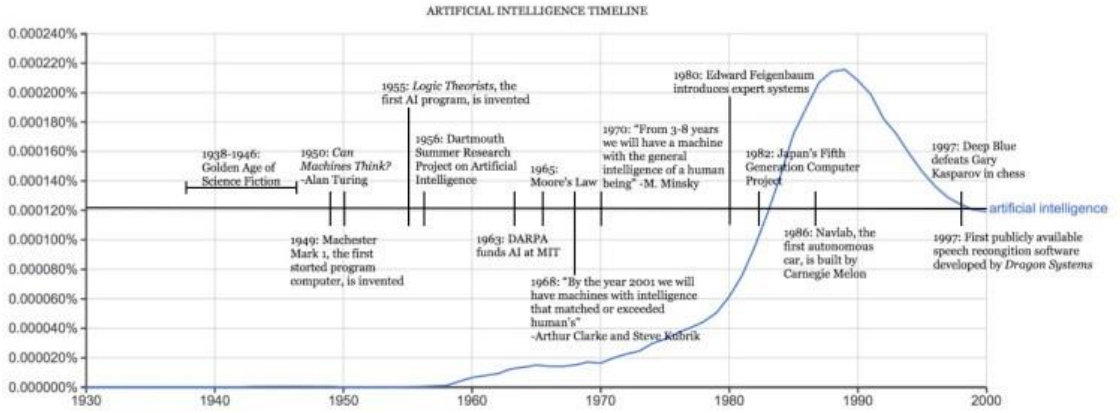


Рисунок 1.2 – Інфографіка розвитку штучного інтелекту

1940-1960: народження штучного інтелекту та пробудження кібернетики. 2520/5000 Період між 1940 та 1960 роками був сильно ознаменований поєднанням технологічних розробок (прискорювачем яких була Друга світова війна) та прагненням зрозуміти, як поєднати функціонування машин та органічних істот. Для Норберта Вінера, піонера кібернетики, його метою було об'єднати математичну теорію, електроніку та автоматику як "цілу теорію управління та зв'язку як у тварин, так і в машинах". Незадовго перша математична та комп'ютерна модель біологічного нейрона (формальний нейрон) була розроблена Уорреном Маккаллохом та Уолтером Пітсом ще в 1943 році [16].

На початку 1950 р. Джон Фон Нойман і Алан Тьюрінг не створили термін AI, але були батьками-засновниками технології, що знаходиться за ним: вони здійснили перехід від комп'ютерів до десяткової логіки 19 століття (яка, таким чином, стосувалася значень від 0 до 9) і машини для бінарної логіки (яка покладається на булеву алгебру, що має справу з більш-менш важливими ланцюжками 0 або 1). Таким чином, два дослідники формалізували архітектуру сучасних комп'ютерів і продемонстрували, що це універсальна машина, здатна виконувати те, що запрограмовано. Тьюрінг, з іншого боку, вперше підняв питання про можливий інтелект машини в своїй знаменитій статті 1950 року "Обчислювальна техніка та інтелект" і описав "гру в наслідування", де людина повинна бути в змозі розрізнити телелітичний діалог, чи спілкується він з людиною чи машиною. Як би ця стаття не була суперечливою (цей "тест Тьюрінга" [17], як видається, не підходить для багатьох експертів), її часто називають джерелом допиту кордону між людиною і машиною.

Термін "ШІ" можна було б віднести до Джона Маккарті з MIT (Массачусетський технологічний інститут) [18], який Марвін Мінський (Університет Карнегі-Меллона) визначає як "побудову комп'ютерних програм, які займаються завданнями, які в даний час більш задовільно виконуються людьми тому що вони потребують розумових процесів високого рівня, таких

як: перцептивне навчання, організація пам'яті та критичні міркування. Літня конференція 1956 року в Дартмутському коледжі (фінансується Інститутом Рокфеллера) вважається основоположником цієї дисципліни. Успіх того, що було не конференцією, а швидше семінаром, Лише шість людей, включаючи Маккарті та Міньського, постійно залишалися постійно присутніми в цій роботі (яка по суті спиралася на розробки, засновані на формальній логіці).

Незважаючи на те, що технології залишаються захоплюючими та перспективними (див., Наприклад, статтю Ріда К. Лоулора 1963 року, члена Каліфорнійської колегії під назвою "Що можуть робити комп'ютери: аналіз та прогнозування судових рішень"), популярність технології зменшилася на початку 1960-х. Машини мали дуже мало пам'яті, що ускладнювало використання комп'ютерної мови. Однак сьогодні вже існують деякі основи, такі як дерева рішень для вирішення проблем: IPL, мова обробки інформації, таким чином дала можливість написати ще в 1956 р. Програму LTM (теоретик теорії логіки), яка мала на меті продемонструвати математичну теореми.

Герберт Саймон, економіст і соціолог, пророкував у 1957 році, що штучний інтелект в наступні 10 років вдасться перемогти людину в шахах. Бачення Саймона виявилося правильним аж через 30 років [19].

1980-1990 та експертні системи. У 1968 році Стенлі Кубрик зняв фільм «Космічна одісея 2001 року», де комп'ютер - HAL 9000 (лише один лист від IBM) узагальнює в собі всю суму етичних питань, поставлених A.I.: чи це буде представляти високий рівень витонченості, добро для людства чи небезпека? Вплив фільму, природно, не буде науковим, але він сприятиме популяризації теми, як і автор наукової фантастики Філіп К. Дік, який ніколи не перестане замислюватися, чи одного разу машини переживуть емоції.

Саме з появою перших мікропроцесорів наприкінці 1970 року ШІ знову знявся і вступив у золоту еру експертних систем. Цей шлях був фактично відкритий в MIT в 1965 році з DENDRAL (експертна система, що спеціалізується на молекулярній хімії), і в Стенфордському університеті в 1972 році з MYCIN (система, що спеціалізується на діагностиці захворювань

крові та ліків, що відпускаються за рецептом). Ці системи базувалися на "двигуні висновку", який був запрограмований як логічне дзеркало людських міркувань. Вводячи дані, двигун надав відповіді високого рівня знань.

Обіцянки передбачали масштабний розвиток, але манія знову впаде наприкінці 1980-х, на початку 1990-х років. Програмування таких знань насправді вимагало великих зусиль і від 200 до 300 правил, там був ефект "чорної скриньки" там, де це було Не зрозуміло, як аргументувала машина. Таким чином, розробка та обслуговування стали вкрай проблематичними, і, перш за все, швидшими, та були можливі багато інших менш складних і менш дорогих способів. Слід нагадати, що у 90-х роках термін штучний інтелект майже став табу, і більш скромні варіації навіть увійшли до університетської мови, наприклад "сучасні обчислення".

Успіх Deep Blue (експертна система IBM) у травні 1997 р. У шаховій грі проти Гаррі Каспарова виконав пророцтво Герберта Саймона у 1957 р. 30 років пізніше, але не підтримав фінансування та розвиток цієї форми ШІ. Операція Deep Blue грунтувалася на алгоритмі систематичної грубої сили, де всі можливі рухи оцінювались і зважувались. Поразка людини залишалася дуже символічною в історії, але Глибокому Блакиті насправді вдалося розглянути лише обмежений периметр (той, що стосується правил шахової гри), дуже далекий від здатності моделювати складність світу.

З 2010 року почався новий розквіт, заснований на великій кількості даних для обробки та нових обчислювальних можливостях [20]. Два чинники пояснюють новий бум у цій дисципліні близько 2010 року:

- Перш за все, доступ до величезних обсягів даних. Наприклад, щоб мати можливість використовувати алгоритми для класифікації зображень та розпізнавання котів, раніше потрібно було самостійно здійснити вибірку. Сьогодні за допомогою простого пошуку в Google можна знайти мільйони.
- Тоді виявлення дуже високої ефективності процесорів комп'ютерних відеокарт для прискорення обчислення алгоритмів навчання. Цей процес був дуже ітеративним, для обробки всього зразка може

знадобитися тижнів до 2010 року. Обчислювальна потужність цих карт (здатна здійснювати більше тисячі мільярдів транзакцій в секунду) дозволила досягти значного прогресу при обмежених фінансових витратах (менше 1000 євро на карту).

Це нове технологічне обладнання дало змогу досягти значних успіхів у суспільстві та збільшило фінансування: у 2011 році Ватсон, IA IBM, виграє ігри проти 2-х чемпіонів по небезпеці! ». У 2012 році Google X (пошукова лабораторія Google) зможе мати AI-розпізнавання котів на відео. Для цього останнього завдання було використано понад 16000 процесорів, але потенціал надзвичайний: машина вчиться щось відрізняти. У 2016 році AlphaGO (AI, який спеціалізується на Іграх Google), обіграє чемпіонку Європи (Fan Hui) та чемпіонку світу (Lee Sedol), а потім себе (AlphaGo Zero). Зазначимо, що гра Go має комбінаторику набагато важливішою, ніж шахи (більше, ніж кількість частинок у Всесвіті), і що не можна отримати таких значних результатів у сильній силі (як у Deep Blue у 1997 р.).

Звідки взялося це диво? Повна зміна парадигми від експертних систем. Підхід став індуктивним: це вже не питання кодування правил, як для експертних систем, а дозволення комп'ютерам відкривати їх поодинці за допомогою кореляції та класифікації на основі величезної кількості даних.

Серед методів машинного навчання глибоке навчання видається найбільш перспективним для ряду програм (включаючи розпізнавання голосу чи зображення). У 2003 році Джеффрі Хінтон (Університет Торонто), Йошуа Бенджо (Університет Монреаля) та Ян Лекун (Університет Нью-Йорка) вирішили розпочати дослідницьку програму, щоб оновити нейронні мережі. Експерименти, проведені одночасно в Microsoft, Google та IBM за допомогою лабораторії в Торонто в Хінтоні, показали, що цей тип навчання вдався вдвічі зменшити кількість помилок для розпізнавання мовлення. Аналогічних результатів досягла команда з розпізнавання зображень Гінтона [21].

За ніч значна частина дослідницьких команд звернулася до цієї технології з незаперечними перевагами. Цей тип навчання також дозволив

досягти значного прогресу в розпізнаванні тексту, але, на думку експертів, таких як Ян Лекун, ще потрібно довгий шлях до створення систем розуміння тексту. Розмовні агенти добре ілюструють цей виклик: наші смартфони вже вміють транскрибувати інструкцію, але не можуть повністю контекстуалізувати її та проаналізувати наші наміри. На рисунку 1.3 зображена історія штучного інтелекту

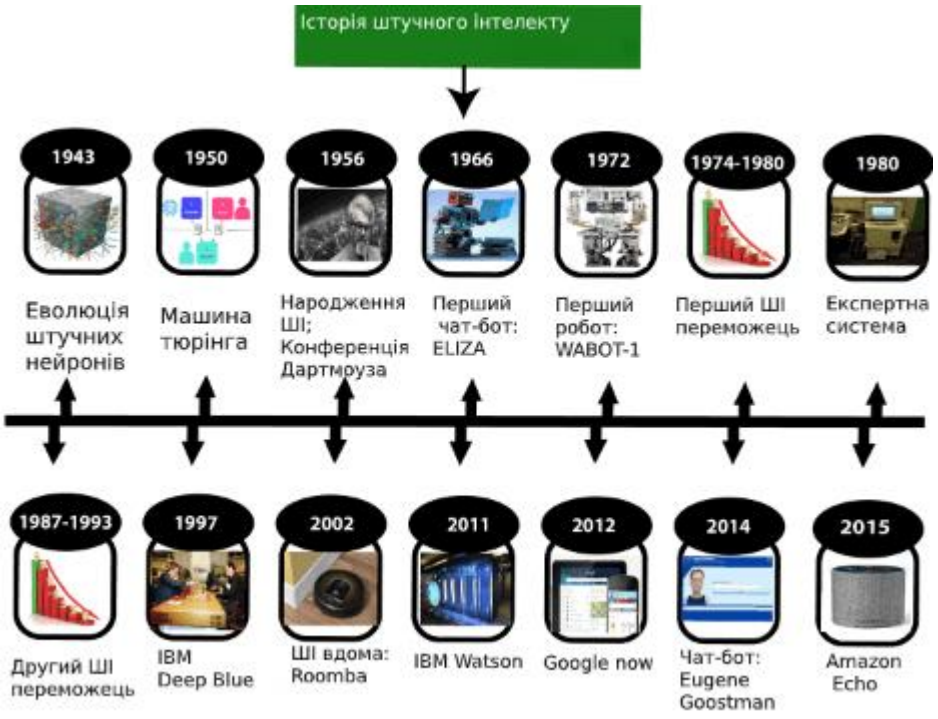


Рисунок 1.3 – Історія штучного інтелекту

1.3 Штучний інтелект в сучасному світі

Штучний інтелект - галузь обчислювальних наук, яка наголошує на розвитку машин інтелекту, мислення та роботи, як люди. Наприклад, розпізнавання мовлення, вирішення проблем, навчання та планування. 32% керівників стверджують, що розпізнавання голосу - це найпоширеніша технологія штучного інтелекту у їхньому бізнесі сьогодні.

Сьогодні штучний інтелект - дуже популярний предмет, який широко обговорюється в технологічних та ділових колах. Багато експертів та

галузевих аналітиків стверджують, що штучний інтелект або машинне навчання - це майбутнє, але якщо ми оглянемось, ми переконані, що це не майбутнє - це сучасність.

З розвитком технологій ми вже так чи інакше пов'язані з штучним інтелектом - будь то Сірі, Ватсон чи Алекса [22]. Так, технологія знаходиться на початковій фазі, і все більше компаній інвестують ресурси в машинне навчання, що свідчить про стійке зростання продуктів та додатків AI в найближчому майбутньому.

Наступна статистика надасть уявлення про зростання:

- У 2014 році в стартапи AI було вкладено понад 300 мільйонів доларів, що демонструє зростання на 300% порівняно з попереднім роком (Bloomberg)
- До 2018 року 6 мільярдів підключених пристроїв будуть проактивно просити про підтримку. (Гартнер)
- До кінця 2018 року "цифрові помічники клієнтів" розпізнають клієнтів по обличчю та голосу через канали та партнерів (Gartner)
- Штучний інтелект замінить 16% американських робочих місць до кінця десятиліття (Forrester)
- 15% користувачів власників телефонів Apple використовують функції розпізнавання голосу Siri. (BGR)

На відміну від загального сприйняття, штучний інтелект не обмежується лише IT чи технологічною галуззю; натомість він широко використовується в інших сферах, таких як медицина, бізнес, освіта, право та виробництво.

1.4 Висновок до першого розділу

Штучний інтелект (ШІ), машинне навчання і нейронні мережі - терміни, використовувані для опису потужних технологій, які базуються на машинному навчанні, здатні вирішити безліч завдань з реального світу.

У той час, як роздум, прийняття рішень і т.д. порівняно зі здібностями людського мозку у машин далекі від ідеалу (не ідеальні вони, зрозуміло, і у людей), в недавній час було зроблено кілька важливих відкриттів в області технологій ШІ і пов'язаних з ними алгоритмів. Важливу роль відіграє збільшення кількості доступних для навчання ІІ великих вибірок різноманітних даних.

Область ШІ перетинається з багатьма іншими областями, включаючи математику, статистику, теорію ймовірностей, фізику, обробку сигналів, машинне навчання, комп'ютерне зір, психологію, лінгвістику і науку про мозок. Питання, пов'язані із соціальною відповідальністю та етикою створення ШІ притягують цікавляться людей, що займаються філософією.

2 МАШИННЕ НАВЧАННЯ ТА ЙОГО МОДЕЛІ

2.1 Поняття «машинного навчання»

Машинне навчання (англ. *machine learning*) — це підгалузь штучного інтелекту в галузі інформатики, яка часто застосовує статистичні прийоми для надання комп'ютерам здатності «навчатися» (тобто, поступово покращувати продуктивність у певній задачі) з даних, без того, щоби бути програмованими явноОсновною передумовою машинного навчання є побудова алгоритмів, які можуть приймати вхідні дані та використовувати статистичний аналіз для прогнозування виходу при оновленні результатів, коли нові дані стануть доступними [23].

Процеси, що беруть участь у машинному навчанні, аналогічні процесам видобутку даних та прогнозування моделювання. Обидва вимагають пошуку даних, щоб шукати шаблони та відповідно коригувати дії програми. Багато людей знайомі з машинним навчанням за допомогою покупок в Інтернеті та подають рекламу, пов’язану з їх покупкою. Це відбувається тому, що механізми рекомендацій використовують машинне навчання, щоб персоналізувати доставку реклами в режимі реального часу. Крім персоналізованого маркетингу, інші поширені випадки використання машинного навчання включають виявлення шахрайства, фільтрацію спаму, виявлення загрози мережевій безпеці, прогнозне обслуговування та створення новин.

Непідтримувані алгоритми не потребують навчання з бажаними даними про результати. Натомість вони використовують ітеративний підхід, який називається глибоким навчанням, для перегляду даних та отримання висновків. Непідконтрольні алгоритми навчання - їх також називають нейронними мережами - використовуються для складніших завдань обробки, ніж керовані системи навчання, включаючи розпізнавання зображень, мовлення в текст та створення природних мов. Ці нейронні мережі працюють,

комбінуючи мільйони прикладів навчальних даних та автоматично ідентифікуючи часто тонкі кореляції між багатьма змінними [24]. Після навчання, алгоритм може використовувати свій банк асоціацій для інтерпретації нових даних. Ці алгоритми стали здійсненними лише в епоху великих даних, оскільки для них потрібні великі обсяги навчальних даних. На рисунку 2.1 зображений один з алгоритмів машинного – кластеризація, яка використовується для пошуку патернів в данних.

Рисунок 2.1 – Принцип кластеризації

Далі будуть наведені приклади використання машинного навчання. Машинне навчання сьогодні використовується в широкому діапазоні застосувань. Один з найвідоміших прикладів - News Feed Facebook. News Feed використовує машинне навчання, щоб персоналізувати канал кожного користувача. Якщо учасник перестає прокручувати сторінку, щоб прочитати або вподобати публікаціям певного друга, News Feed почне показувати більше активності цього друга у стрічці користувача. Програмне забезпечення просто використовує статистичний аналіз та прогностичну аналітику для виявлення шаблонів даних користувача та використання цих моделей для заповнення новістної сторінки. Якщо член більше не перестане читати, сподобатися чи коментувати публікації свого друга, нові дані будуть включені до набору даних, а новістна сторінка буде відповідно коригуватися.

Машинне навчання також входить до масиву корпоративних додатків. Системи управління взаємовідносинами з клієнтами (CRM) використовують моделі навчання для аналізу електронної пошти та оперативних членів торгової групи, щоб спочатку відповісти на найбільш важливі повідомлення. Більш просунуті системи можуть навіть рекомендувати потенційно ефективні відповіді. Постачальники бізнес-аналітики та аналітики використовують машинне навчання у своєму програмному забезпеченні, щоб допомогти користувачам автоматично ідентифікувати потенційно важливі точки даних. Системи людських ресурсів використовують моделі навчання для визначення характеристик ефективних працівників та спираються на ці знання, щоб знайти найкращих претендентів на відкриті посади.

Машинне навчання також відіграє важливу роль у самокерованих автомобілях. Нейронні мережі глибокого навчання використовуються для ідентифікації об'єктів та визначення оптимальних дій для безпечного керування транспортним засобом вниз по дорозі [25].

Технологія віртуального асистента також працює за допомогою машинного навчання. Розумні помічники поєднують кілька моделей глибокого навчання для інтерпретації природного мовлення, приведення у відповідний контекст - наприклад, особистий графік користувача або попередньо визначені вподобання - та вживають дій, як бронювання рейсу або витягування маршруту водіння.

Реалізації машинного навчання класифікуються на чотири основні категорії, залежно від характеру навчального “сигналу” чи “відповіді”, доступного для системи навчання, які є наступними [26]:

- Контрольоване навчання: коли алгоритм вчиться на прикладі даних та пов'язаних з ними цільових відповідей, які можуть складатися з числових значень або міток рядків, таких як класи або теги, щоб пізніше передбачити правильну відповідь, коли вони поставлені з новими прикладами, підпадає під категорію контрольованого навчання . Цей підхід дійсно схожий на навчання людини під наглядом вчителя. Учитель надає хороші приклади

для запам'ятовування учня, а потім учень виходить із цих конкретних прикладів загальних правил.

- Навчання без нагляду: Коли алгоритм вчиться на простих прикладах без будь-якої пов’язаної відповіді, залишаючи алгоритм визначати шаблони даних самостійно. Цей тип алгоритму має тенденцію до реструктуризації даних у щось інше, наприклад, нові функції, які можуть представляти клас або нову серію некорельованих значень. Вони є досить корисними для надання людям уявлення про значення даних та нових корисних даних для керованих алгоритмів машинного навчання. Як різновид навчання, він нагадує методи, якими користується людина, щоб з'ясувати, що певні предмети чи події є з одного класу, наприклад, шляхом спостереження за ступенем подібності між об'єктами. Деякі системи рекомендацій, які ви знайдете в Інтернеті у формі автоматизації маркетингу, засновані на такому типі навчання.

- Підсилення навчання: Коли ви представляєте алгоритм з прикладами, на яких відсутні мітки, як у навчанні без дозволу. Однак ви можете супроводжувати приклад позитивними чи негативними зворотними зв’язками відповідно до рішення, яке пропонує алгоритм, підпадає під категорію навчального підкріплення, яка підключена до програм, щодо яких алгоритм повинен приймати рішення (тому виріб є рецептурним, а не просто описовим, як у непідконтрольному навчанні), і рішення несуть наслідки. У людському світі це подібно до навчання шляхом спроб та помилок [27].Помилки допомагають вам навчитися, оскільки на них додається штраф (вартість, втрата часу, жаль, біль тощо), навчаючи вас, що певний спосіб дії є менш шансовим досягти успіху, ніж інші. Цікавий приклад підкріплення навчання відбувається, коли комп’ютери навчаються грати у відеоігри самостійно.У цьому випадку додаток представляє алгоритм із прикладами конкретних ситуацій, таких як геймер застряг у лабіринті, уникаючи ворога. Додаток дозволяє алгоритму знати результат дій, які він вживає, і навчання відбувається, намагаючись уникнути того, що виявляється небезпечним і

досягти виживання. Ви можете ознайомитись з тим, як компанія Google DeepMind створила навчальну програму підкріплення, яка відтворює відеоігри старих Atari. Переглядаючи відео, зауважте, як програма спочатку незграбна і некваліфікована, але стабільно покращується з навчанням, поки не стане чемпіоном.

- Навчання під наглядом: де подається неповний сигнал навчання: навчальний набір, у якому відсутні деякі (часто багато) цільових результатів. Існує особливий випадок цього принципу, відомий як Transduction, коли весь набір проблемних випадків відомий під час навчання, за винятком того, що частина цілей відсутня.

Ще одна категоризація завдань машинного навчання виникає, коли варто враховувати бажані вихідні дані машинної системи навчання:

- Класифікація: Коли дані поділяються на два або більше класів, і учень повинен створити модель, яка призначає невидимі входи до одного або декількох (класифікація на багато міток) цих класів. Зазвичай це вирішується під наглядом. Фільтрація спаму є прикладом класифікації, де вхідними повідомленнями є повідомлення електронної пошти (або інші), а класи - "спам" та "не спам".

- Регресія: що також є контрольованою проблемою, випадок, коли виходи є безперервними, а не дискретними. На рисунку 2.2 зображені класифікація та регресія.

- Кластеризація: Коли набір входів слід розділити на групи. На відміну від класифікації, групи заздалегідь не відомі, що робить це, як правило, непідвладним завданням. На рисунку 2.3 зображений принцип кластеризації [28].

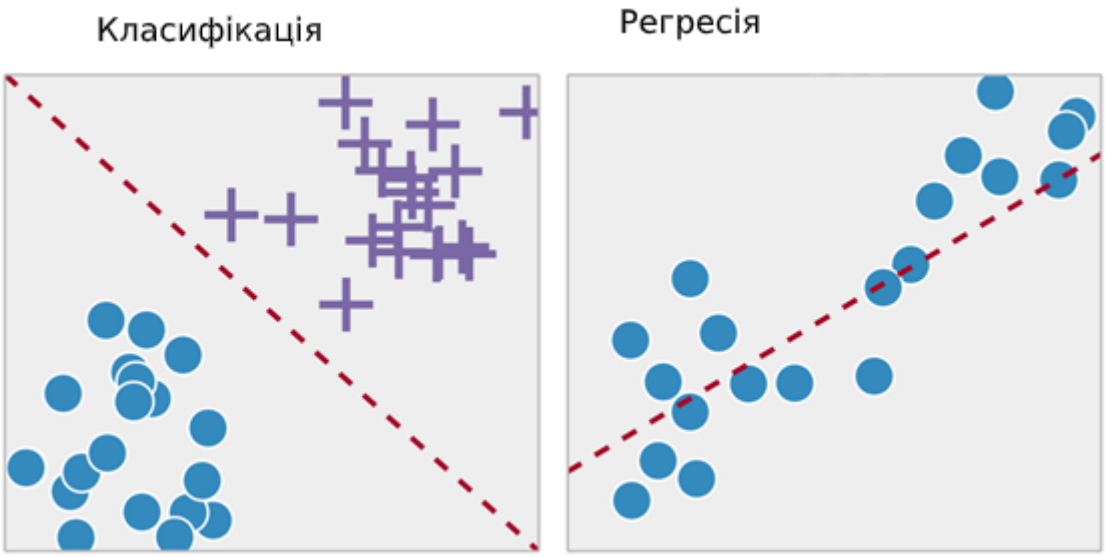


Рисунок 2.2 – Класифікація та Регресія

Машинне навчання застосовується тоді, коли проблеми неможливо вирішити за допомогою типових підходів.

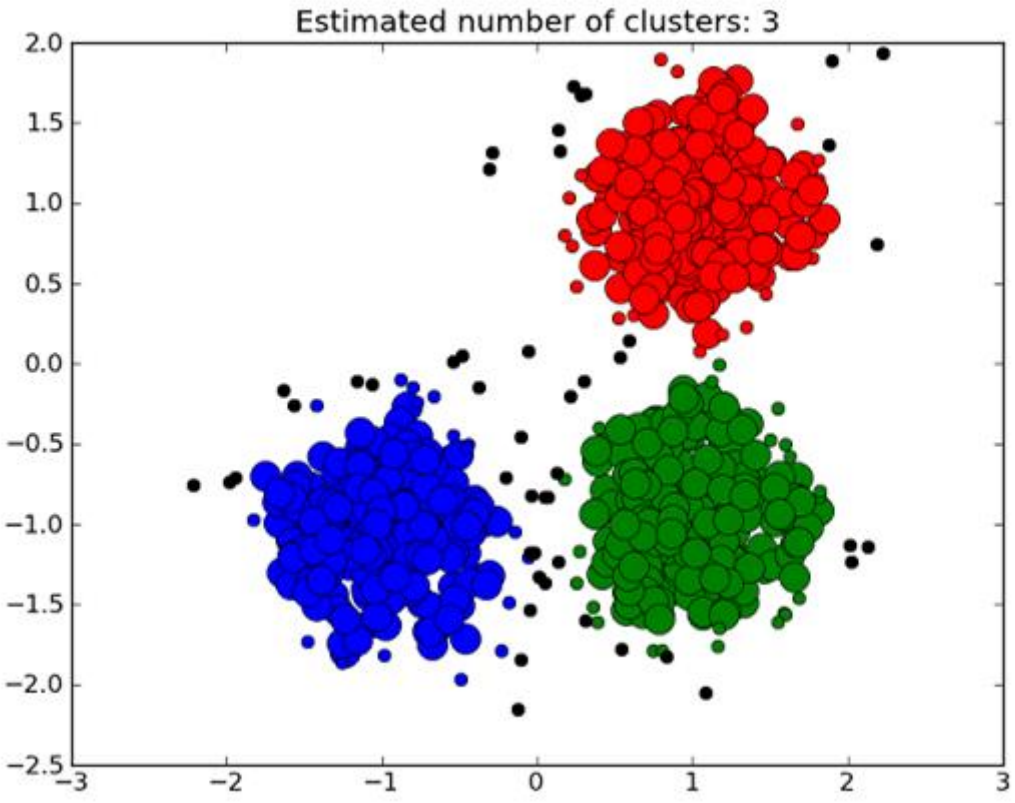


Рисунок 2.3 – Кластеризація з трьома кластерами

Існують різні підходи аби запустити процес машинного навчання, від використання дерев базових рішень до кластеризації до шарів штучних нейронних мереж (останній з яких поступився місцем глибокому навчанню), залежно від того, яке завдання ви намагаєтеся виконати, типу та кількість наявних у вас даних. Ця динаміка розглядається в програмах настільки ж різними, як медична діагностика або автошкола.

Дослідження, проведені під час роботи над реальними додатками, часто сприяють прогресу в цій галузі, а причини двоякі:

- Схильність до виявлення меж та обмежень існуючих методів.
- Дослідники та розробники, що працюють з експертами в галузі домену та використовують час та досвід для підвищення продуктивності системи.

Іноді це також відбувається «випадково». Ми можемо розглядати модельні ансамблі або комбінації багатьох алгоритмів навчання для підвищення точності. Команди, які змагалися за ціну Netflix 2009 року, виявили, що вони отримали найкращі результати, поєднуючи своїх учнів з учнями іншої команди, в результаті чого вдосконалився алгоритм рекомендацій (читайте в блозі Netflix докладніше про те, чому вони не закінчилися використовувати цей ансамбль).

Один важливий момент (заснований на інтерв'ю та бесідах з експертами в цій галузі), що стосується застосування в бізнесі та інших місцях, - це те, що машинне навчання - це не просто, а то й навіть, автоматизація, часто неправильно зрозуміла концепція. Якщо ви думаєте таким чином, ви зобов’язані пропустити цінні уявлення, які можуть надати машини, і отримані можливості (переосмислення цілої бізнес-моделі, наприклад, як у галузях промисловості, таких як виробництво та сільське господарство).

Машини, які навчаються, корисні людям, оскільки, використовуючи всю потужність обробки, вони можуть швидше виділити або знайти зразки у великих (або інших) даних, які інакше люди не пропустили б. Машинне навчання - це інструмент, який можна використовувати для розширення

здібностей людини до вирішення проблем та обґрунтованих висновків щодо широкого спектру проблем, починаючи від допомоги діагностиці захворювань до розробки глобальних змін клімату.

"Машинне навчання не може отримати щось із нічого ... те, що воно робить,стає більшим з меншого". - Доктор Педро Домінго, Університет Вашингтона [29].

Дві найбільші, історичні (і тривалі) проблеми машинного навчання включали переозброєння (в якому модель виявляє упередженість щодо навчальних даних і не узагальнює нові дані та / або дисперсію, тобто вивчає випадкові речі під час навчання нових даних) та розмірність (алгоритми з більшою кількістю функцій працюють у вищих / декількох вимірах, що ускладнює розуміння даних). Наявність доступу до достатньо великого набору даних у деяких випадках також була основною проблемою.

Однією з найпоширеніших помилок серед початківців машинного навчання є успішна перевірка даних про навчання та наявність ілюзії успіху; Домінго (та інші) наголошують на важливості зберігання деяких наборів даних під час тестування моделей і лише використання цих зарезервованих даних для тестування обраної моделі з подальшим вивченням усього набору даних.

Коли алгоритм навчання (тобто, який навчається) не працює, часто швидший шлях до успіху полягає в тому, щоб подати машині більше даних, доступність яких на сьогоднішній день відома як головний рушій прогресу в алгоритмах машинного та глибокого навчання останнім часом років; однак це може призвести до проблем зі масштабованістю, за яких у нас є більше даних, але час, щоб дізнатися, що дані залишаються проблемою.

З точки зору мети, машинне навчання - це не мета і рішення саме по собі. Крім того, спроба використовувати його як покривне рішення, тобто "BLANK" не є корисною вправою; натомість, підходити до столу з проблемою або метою, найчастіше, керується більш конкретним запитанням - "BLANK".

Глибоке навчання передбачає вивчення та проектування машинних алгоритмів для вивчення хорошого представлення даних на кількох рівнях

абстрагування (способи організації комп'ютерних систем). Нещодавнє оприлюднення глибокого навчання через DeepMind, Facebook та інші установи виділило його як "наступний кордон" машинного навчання.

Міжнародна конференція з машинного навчання (ICML) вважається однією з найважливіших у світі. Цьогоріч відбувся в червні в Нью-Йорку, і він об’єднав дослідників з усього світу, які працюють над вирішенням сучасних проблем глибокого навчання:

- Безперервне навчання в невеликих наборах даних
- На основі імітаційного навчання та передачі в реальному світі

Системи глибокого навчання за останні десятиліття досягли значних успіхів у таких сферах, як виявлення та розпізнавання викидів, переклад тексту в мовлення, пошук інформації та інші. Зараз дослідження орієнтовані на розробку технологій машинного навчання з ефективними даними, тобто системи глибокого навчання, які дозволять навчатися ефективніше, з однаковою ефективністю за менший час та з меншою кількістю даних, у передових областях, таких як персональне охорона здоров'я, навчання посилення роботів, аналіз настроїв та ін. інші. Нижче наводиться підбірка найкращих практик та концепцій застосування машинного навчання.

- Напевно, найважливішим фактором успішних проектів машинного навчання є функції, які використовуються для опису даних (які залежать від домену), і наявність адекватних даних для тренування моделей.
- Існують випадки, коли алгоритми працюють некоректно, це пов’язано з проблемою з навчальними даними (тобто недостатньою кількістю / викривленими даними; шумними даними або недостатньою характеристикою, що описує дані для прийняття рішень.
- "Простота не передбачає точності" – немає зв'язку між кількістю параметрів моделі та тенденцією до перевиконання.
- Отримання експериментальних даних (на відміну від даних спостережень, над якими ми не маємо контролю) слід робити, якщо це

можливо (наприклад, дані, отримані від надсилання різних варіантів електронного листа до вибіркової вибірки випадкової аудиторії).

- Незалежно від того, чи ми позначаємо дані причинними чи співвідносними, важливішим моментом є прогнозування наслідків наших дій.
- Завжди відкладайте частину вашого набору даних про навчання для перехресної перевірки; ви хочете, щоб обраний вами класифікатор або алгоритм навчання добре працював на нових даних.

2.2 Моделі машинного навчання

Існує таке поняття, як «No Free Lunch» теорема. Її суть полягає в тому, що немає такого алгоритму, який був би кращим вибором для кожного завдання, що особливо стосується навчання з учителем.

Наприклад, не можна сказати, що нейронні мережі завжди працюють краще, ніж дерева рішень, і навпаки. На ефективність алгоритмів впливає безліч факторів на кшталт розміру і структури набору даних.

З цієї причини доводиться пробувати багато різних алгоритмів, перевіряючи ефективність кожного на тестовому наборі даних, і потім вибирати кращий варіант. Само собою, потрібно вибирати серед алгоритмів, відповідних поставленій задачі. Якщо проводити аналогію, то при прибиранні будинку, швидше за все, буде використовуватись пилосос, мітлу або швабру, але ніяк не лопату.

Алгоритми машинного навчання можна описати як навчання цільової функції f, яка найкращим чином співвідносить вхідні змінні X і вихідну змінну Y: Y = f (X). Не відомо, що з себе представляє функція f. Адже якби було відомо, то використовували б її безпосередньо, а не намагалися навчити за допомогою різних алгоритмів.

Найбільш поширенним завданням в машинному навчанні є передбачення значень Y для нових значень X. Це називається прогностичним моделюванням, і основна мета - зробити якомога більш точне передбачення.

Нижче наведений топ-10 популярних алгоритмів, які використовуються в машинному навчанні [30].

1. Лінійна регресія - мабуть, один з найбільш відомих і зрозумілих алгоритмів в статистиці і машинному навчанні. Прогностичне моделювання в першу чергу стосується мінімізації помилки моделі або, іншими словами, як можна більш точного прогнозування. Будуть запозичені алгоритми з різних областей, включаючи статистику, і використання їх в цих цілях. Лінійну регресію можна представити у вигляді рівняння, яке описує пряму, найбільш точно ніколи взаємозв'язок між вхідними змінними X і вихідними змінними Y. Для складання цього рівняння потрібно знайти певні коефіцієнти B для вхідних змінних. Це зображено на рисунку 2.4.

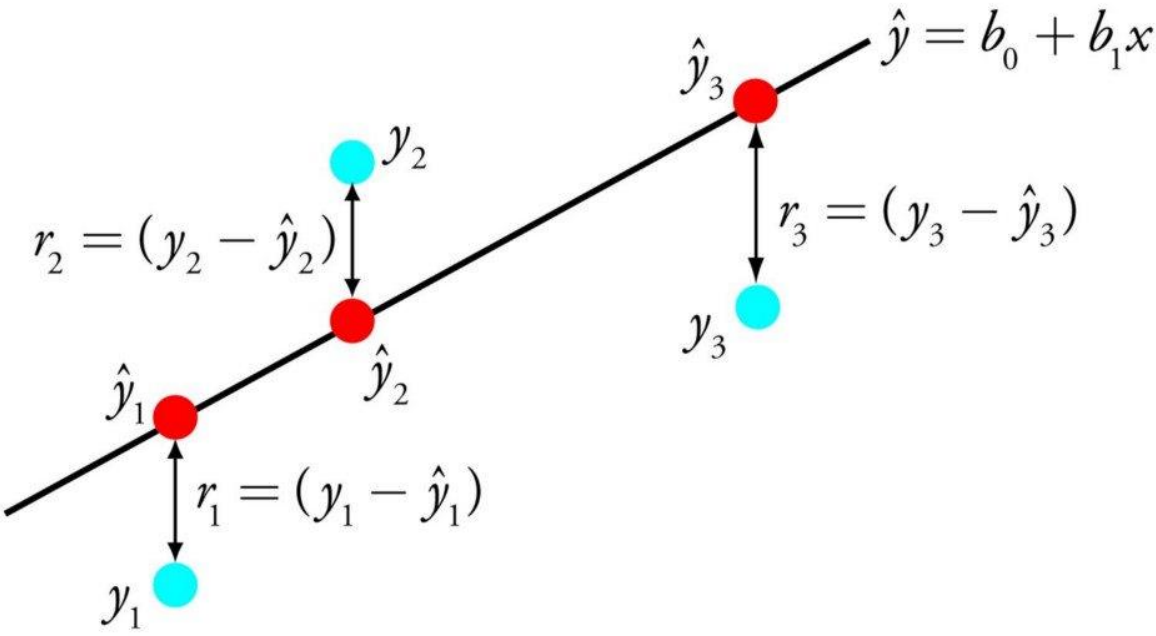


Рисунок 2.4 – Лінійна регресія

Наприклад: Y = B0 + B1 * X. Знаючи X, потрібно знайти Y, і мета лінійної регресії полягає в пошуку значень коефіцієнтів B0 і B1. Для оцінки регресійній моделі використовуються різні методи на кшталт лінійної алгебри або методу найменших квадратів. Лінійна регресія існує вже понад 200 років, і за цей час її встигли ретельно вивчити. Так що ось пара практичних правил

по використанню лінійної регресії: прибрати схожі (корелюючі) змінні і позбутись від шуму в даних, якщо це можливо. Лінійна регресія - швидкий і простий алгоритм, який добре підходить в якості першого алгоритму для вивчення.

2. Логістична регресія - ще один алгоритм, який прийшов в машинне навчання прямо з статистики. Її добре використовувати для завдань бінарної класифікації (це завдання, в яких на виході ми отримуємо один з двох класів). Логістична регресія схожа на лінійну тим, що в ній теж потрібно знайти значення коефіцієнтів для вхідних змінних. Різниця полягає в тому, що вихідне значення перетвориться за допомогою нелінійної або логістичної функції. Логістична функція виглядає як велика буква S і перетворює будь-яке значення в число в межах від 0 до 1. Це дуже корисно, тому що можемо застосувати правило до виходу логістичної функції для прив'язки до 0 і 1 (наприклад, якщо результат функції менше 0.5, то на виході отримуємо 1) і передбачення класу. Це зображено на рисунку 2.5

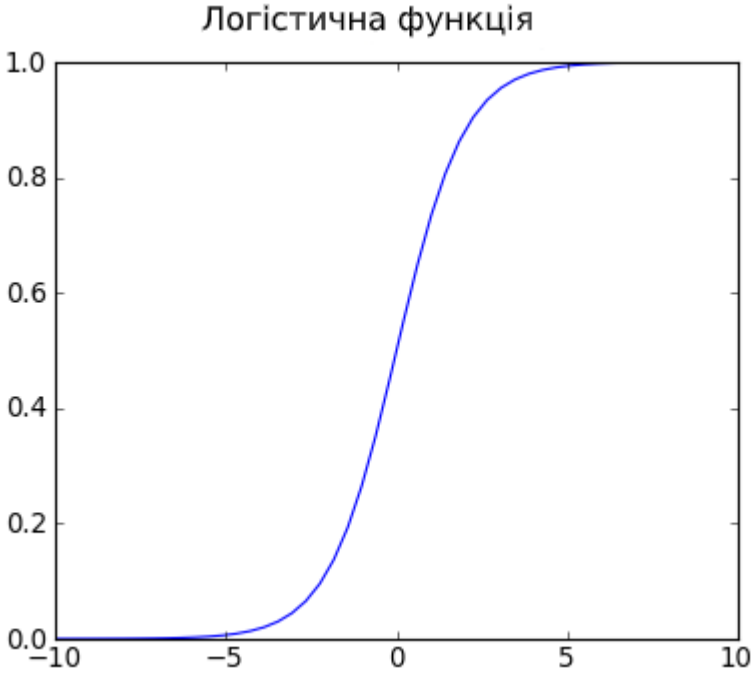


Рисунок 2.5 – Логістична функція.

Завдяки тому, як навчається дана модель, передбачення логістичної регресії можна використовувати для відображення ймовірності приналежності зразка до класу 0 або 1. Це корисно в тих випадках, коли потрібно мати більше обґрунтувань для прогнозування.

Як і у випадку з лінійної регресією, логістична регресія виконує своє завдання краще, якщо прибрати зайві і схожі змінні. Модель логістичної регресії швидко навчається і добре підходить для задач бінарної класифікації.

3. Якщо класів більше, ніж два, то краще використовувати алгоритм LDA (Linear discriminant analysis). Адже логістична регресія використовується, коли потрібно віднести зразок до одного з двох класів. Подання LDA досить просте. Це зображено на рисунку 2.6. Воно складається зі статистичних властивостей даних, розрахованих для кожного класу. Для кожної вхідної змінної це включає:

- Середнє значення для кожного класу;
- Дисперсію, розраховану по всім класам

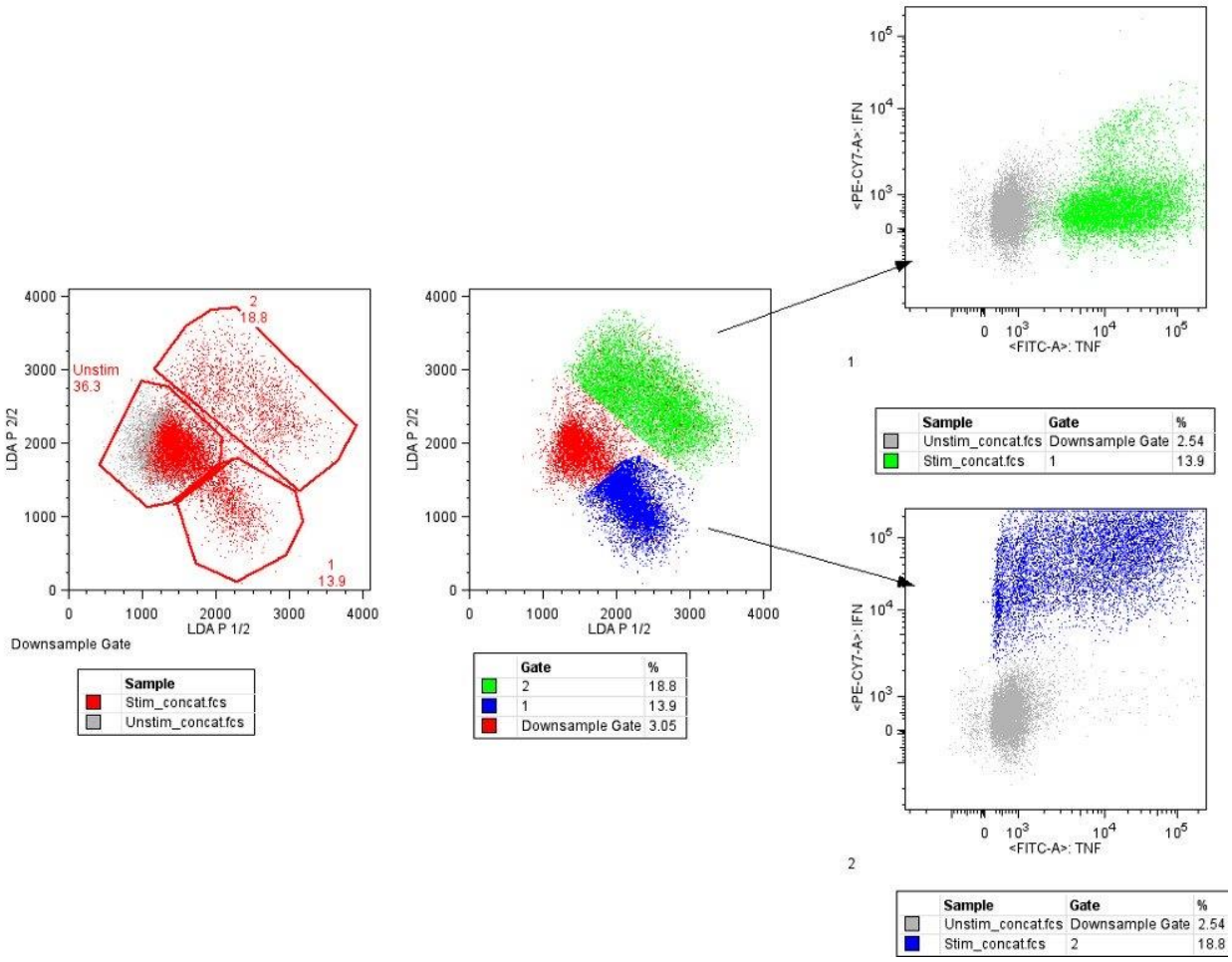


Рисунок 2.6 – Лінійний дискримінантний аналіз

Передбачення виробляються шляхом обчислення дискриминантного значення для кожного класу і вибору класу з найбільшим значенням. Передбачається, що дані мають нормальний розподіл, тому перед початком роботи рекомендується видалити з даних аномальні значення. Це простий і ефективний алгоритм для задач класифікації.

4. Дерево прийняття рішень можна представити у вигляді двійкового дерева, знайомого багатьом по алгоритмам і структурам даних. Кожен вузол являє собою вхідну змінну і точку поділу для цієї змінної (за умови, що змінна - число). На рисунку 2.7 зображено приклад дерева рішень.

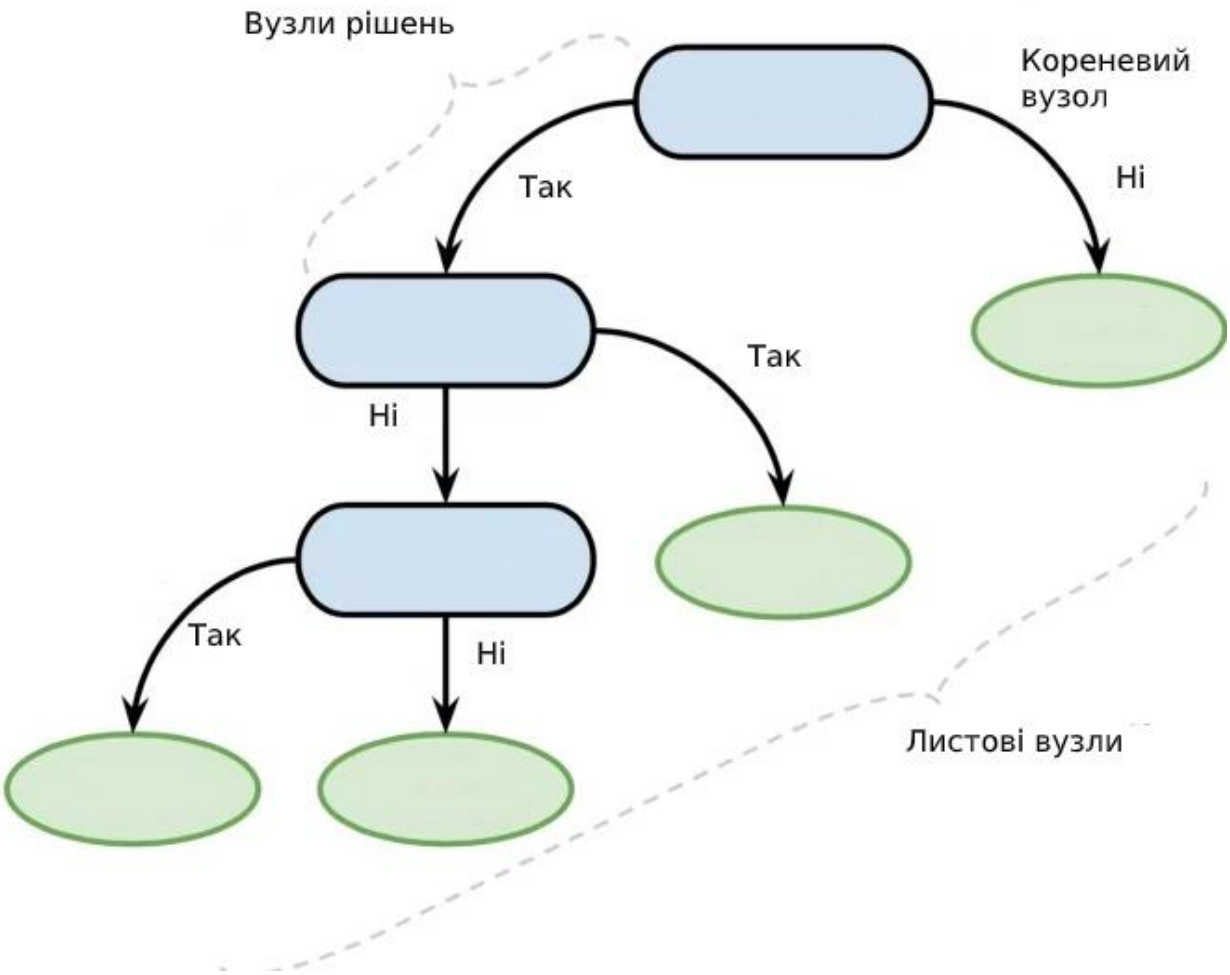


Рисунок 2.7 – Дерево рішень

Листові вузли – це вихідна змінна, яка використовується для передбачення. Пророцтва виробляються шляхом проходу по дереву до листовому вузлу і виведення значення класу на цьому вузлі. Дерева швидко навчаються і роблять прогнози. Крім того, вони точні для широкого кола завдань і не вимагають особливої підготовки даних.

5. Наївний Байєсовий класифікатор - простий але ефективний алгоритм. Модель складається з двух типів ймовірностей, які розраховуються за допомогою даних:

- Ймовірність кожного класу;
- Умовна ймовірність моделі для кожного класу при кожному значенні X.

Після розрахунку ймовірностей моделі, її можна використовувати для передбачення з новими даними за допомогою теореми Байєса. Якщо дані вагомі, то використовуючи нормальний розподіл, розрахувати ці ймовірності не є трудозатратною задачаю. На рисунку 2.8 зображена візуалізація наївного Байєсового класифікатора.

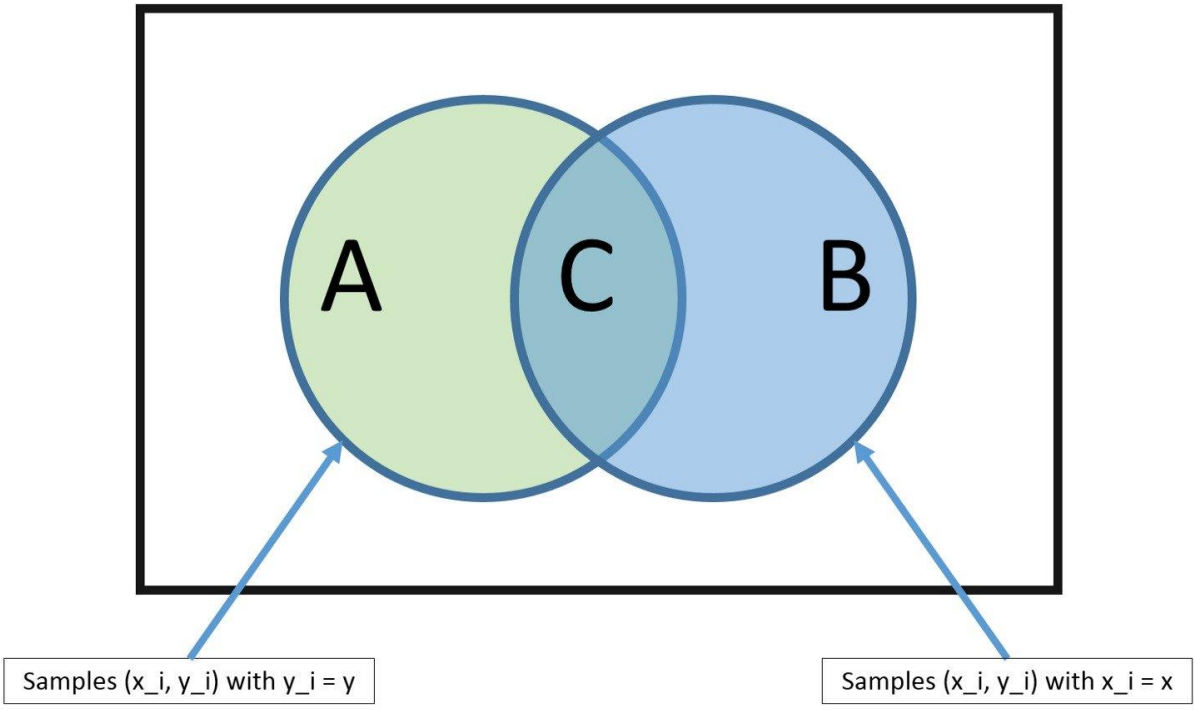


Рисунок 2.8 – Наївний Байєсовий класифікатор

Наївний Байєсовий класифікатор називається наївним, тому що алгоритм припускає, що кожна змінна незалежна одна від одної. Це сильне твердження, яке не відповідає реальним даним. Тим не менш, даний алгоритм вельми ефективний для цілого ряду важких задач, наприклад, класифікації спаму чи розпізнавання рукописних цифр.

6. K-найближчих сусідів (KNN) - дуже простий і дуже ефективний алгоритм. Модель KNN (K-nearest neighbors) представлена всім набором тренувальних даних. Передбачення для нової точки робиться шляхом пошуку K найближчих сусідів в наборі даних і підсумовування вихідної змінної для цих K примірників.

Питання лише в тому, як визначити схожість між екземплярами даних. Якщо всі ознаки мають один і той же масштаб (наприклад, сантиметри), то найпростіший спосіб полягає у використанні евклидово] відстані - числа, яке можна розрахувати на основі відмінностей з кожної вхідної змінної. На рисунку 2.9 зображений принцип К-найближчих сусідів.

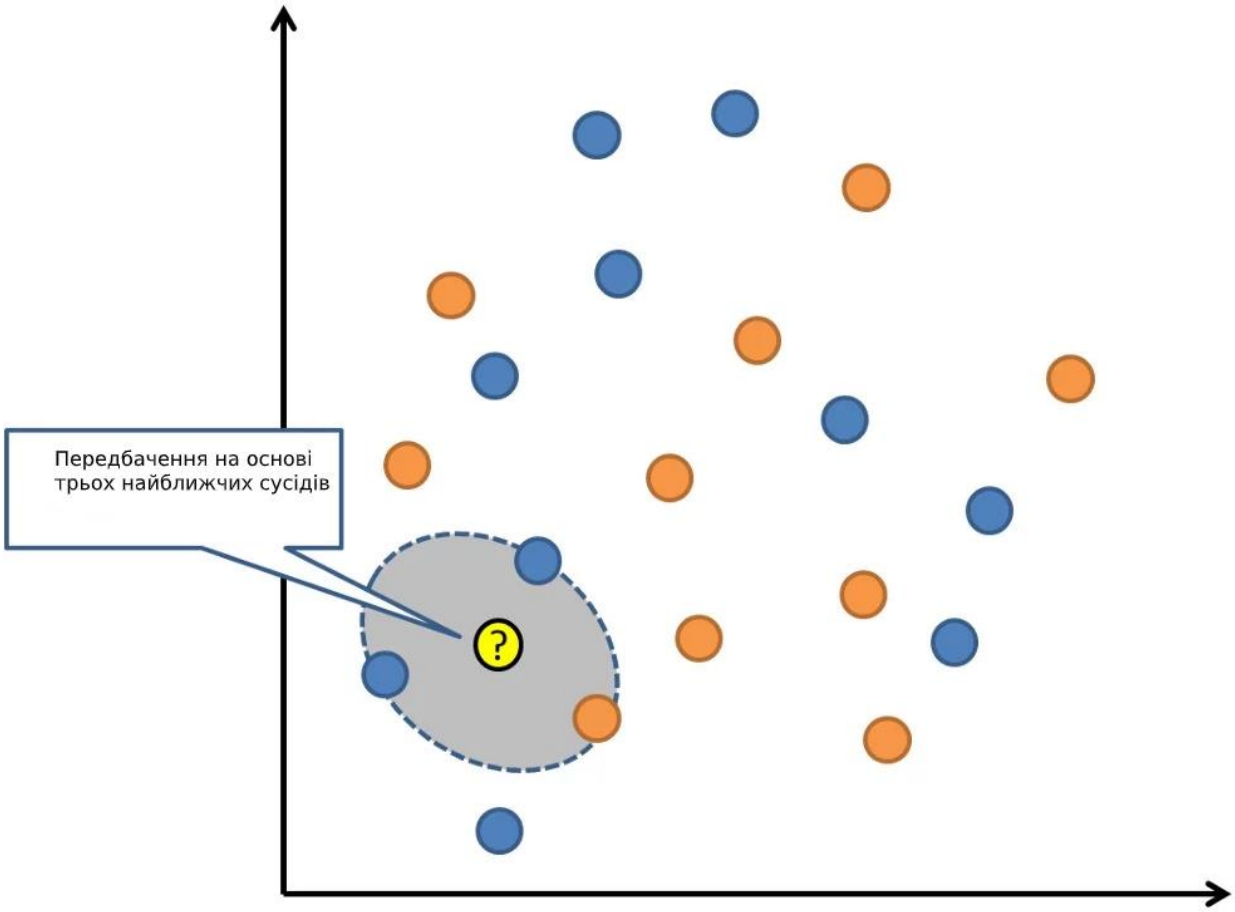


Рисунок 2.9 – К-найближчих сусідів (KNN)

KNN можевимагати багато пам'яті для зберігання всіх даних, але зате швидко зробить прогноз. Також навчальні дані можна оновлювати, щоб передбачення залишалися точними з плином часу. Ідея найближчих сусідів може погано працювати з багатовимірними даними (безліч вхідних змінних), що негативно позначиться на ефективності алгоритму при вирішенні задачі. Це називається прокляттям розмірності. Іншими словами, варто використовувати лише найбільш важливі для передбачення змінні.

7. Мережі векторного квантування (LVQ). Недолік KNN полягає в тому, що потрібно зберігати весь тренувальний набір даних. Якщо KNN добре себе показав, то є сенс спробувати алгоритм LVQ (Learning vector quantization), який позбавлений цього недоліку.

LVQ являє собою набір кодових векторів. Вони вибираються на початку випадковим чином і протягом певної кількості ітерацій адаптуються так, щоб найкращим чином узагальнити весь набір даних. Після навчання ці вектори можуть використовуватися для передбачення так само, як це робиться в KNN. Алгоритм шукає найближчого сусіда (найбільш підходящий кодовий вектор) шляхом обчислення відстані між кожним кодовою вектором і новим екземпляром даних. Потім для найбільш підходящого вектора як передбачення повертається клас (або число в разі регресії). Кращого результату можна досягти, якщо всі дані будуть знаходитися в одному діапазоні, наприклад від 0 до 1. На рисунку 2.10 зображений метод мережі векторного квантування.

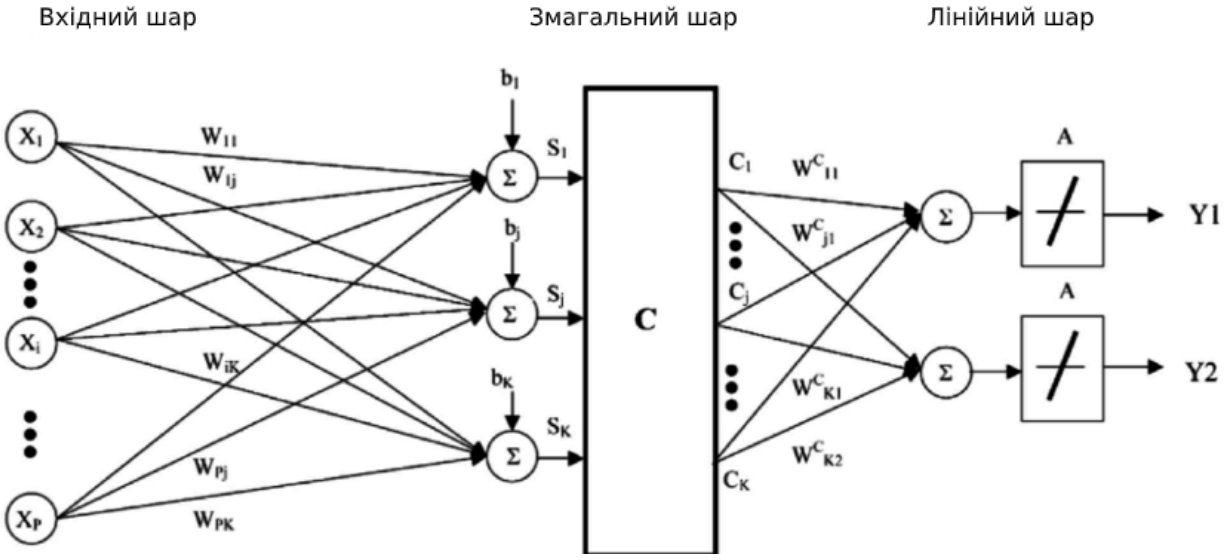


Рисунок 2.10 – Мережа векторного квантування

8. Метод опорних векторів (SVM), ймовірно, один з найбільш популярних і обговорюваних алгоритмів машинного навчання. Гіперплощина - це лінія, що розділяє простір вхідних змінних. У методі опорних векторів гіперплощина вибирається так, щоб найкращим чином розділяти точки в

площині вхідних змінних по їх класу: 0 або 1. В двовимірної площині це можна уявити як лінію, яка повністю поділяє точки всіх класів. Під час навчання алгоритм шукає коефіцієнти, які допомагають краще розділяти класи гиперплоскостью. На рисунку 2.11 зображений метод опорних векторів.

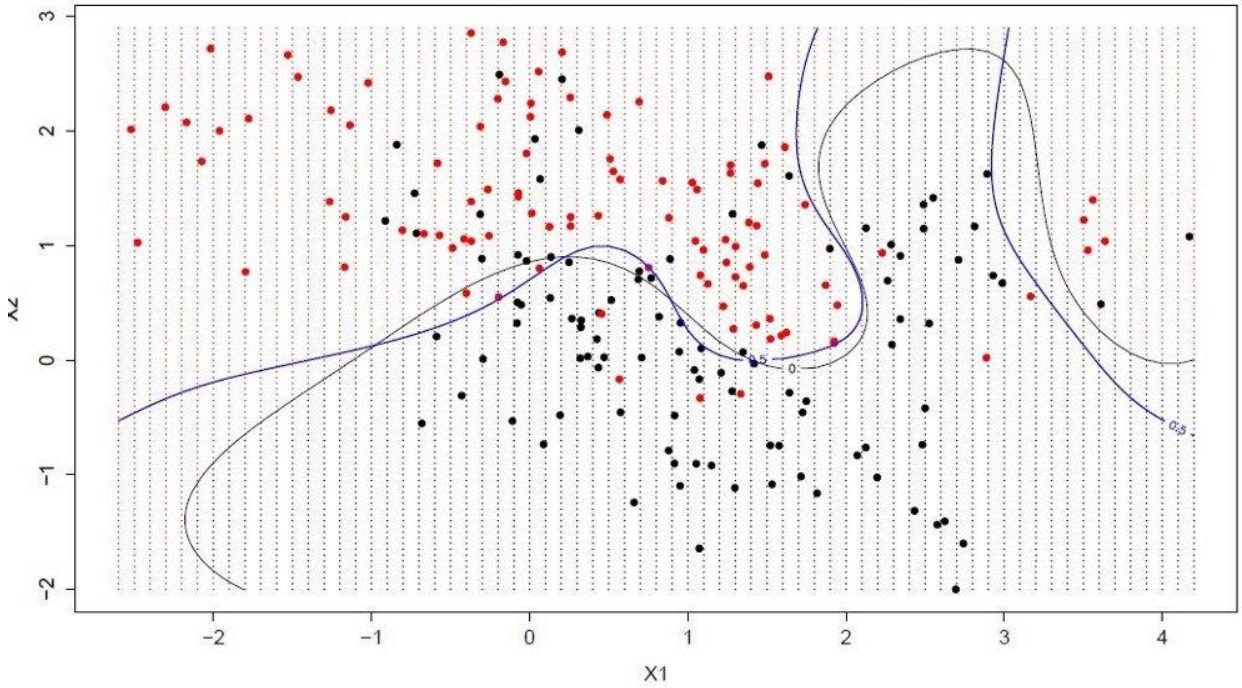


Рисунок 2.11 – Метод опорних векторів

Відстань між гіперплощиною і найближчими точками даних називається різницею. Краща або оптимальна гіперплощина, що розділяє два класи, - це лінія з найбільшою різницею. Тільки ці точки мають значення при визначенні гиперплощини і при побудові класифікатора. Ці точки називаються опорними векторами. Для визначення значень коефіцієнтів, що максимізує різницю, використовуються спеціальні алгоритми оптимізації. Метод опорних векторів, напевно, один з найефективніших класичних класифікаторів, на який безперечно варто звернути увагу.

9. Беггінг і випадковий ліс - дуже популярний і ефективний алгоритм машинного навчання. Це різновид ансамблевого алгоритму, який називається беггінгом. Бутстреп є ефективним статистичним методом для оцінки будь-якої величини на зразок середнього значення. Берете безліч підвибірок з ваших

даних, вважаєте середнє значення для кожної, а потім усереднюються результати для отримання кращої оцінки дійсного середнього значення. У беггінгу використовується той же підхід, але для оцінки всіх статистичних моделей найчастіше використовуються дерева рішень. Тренувальні дані розбиваються на безліч вибірок, для кожної з яких створюється модель. Коли потрібно зробити прогноз, то його робить кожна модель, а потім передбачення усереднюються, щоб дати кращу оцінку значенню. На рисунку 2.12 зображений метод беггінгу та випадкових дерев.

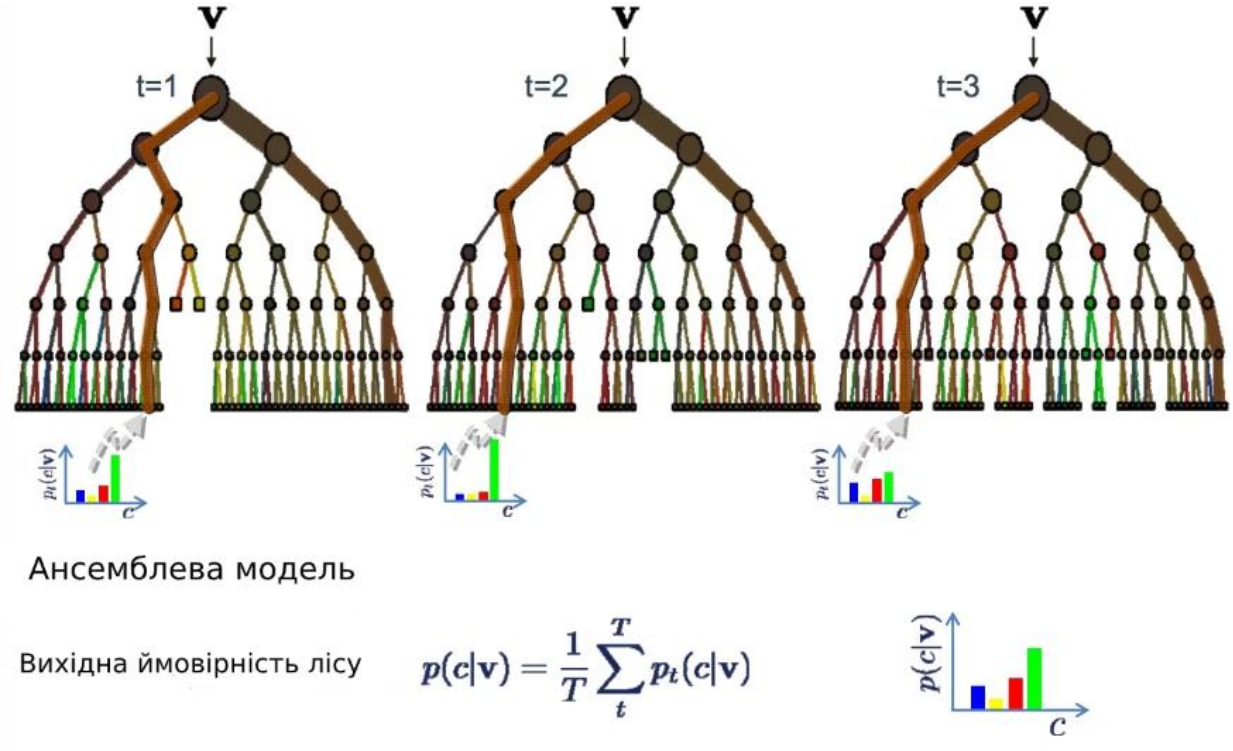


Рисунок 2.12 – Метод беггінгу та випадкових дерев.

В алгоритмі випадкового лісу для всіх вибірок з тренувальних даних будуються дерева рішень. При побудові дерев для створення кожного вузла вибираються випадкові ознаки. Окремо отримані моделі не дуже точні, але при їх об'єднанні якість передбачення значно покращується. Якщо алгоритм з високою дисперсією, наприклад, дерева рішень, показує хороший результат з вхідними даними, то цей результат часто можна покращити, застосувавши беггінг.

10. Бустинг і AdaBoost - це сімейство ансамблевих алгоритмів, суть яких полягає в створенні сильного класифікатора на основі декількох слабких. Для цього спочатку створюється одна модель, потім інша модель, яка намагається виправити помилки в першій. Моделі додаються до тих пір, поки тренувальні дані не будуть ідеально пророкувати або поки не буде перевищено максимальну кількість моделей. AdaBoost був першим дійсно успішним алгоритмом бустінга, розробленим для бінарної класифікації. Саме з нього найкраще починати знайомство з бустінгом. Сучасні методи на кшталт стохастичного градієнтного бустінга грунтуються на AdaBoost. На рисунку 2.13 зображений метод бустингу і AdaBoost [31].

11.

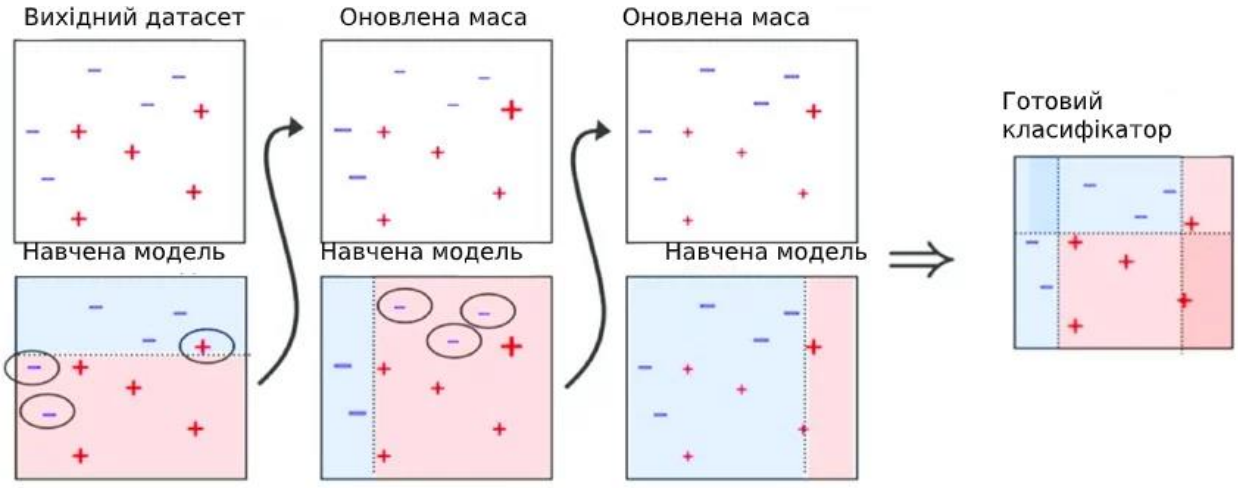


Рисунок 2.13 – Бустинг і AdaBoost

AdaBoost використовують разом з короткими деревами рішень. Після створення першого дерева перевіряється його ефективність на кожному тренувальному об'єкті, щоб зрозуміти, скільки уваги має приділити наступне дерево всім об'єктам. Тим даними, які складно передбачити, дається більшу вагу, а тим, які легко передбачити, - менший. Моделі створюються послідовно одна за одною, і кожна з них оновлює ваги для наступного дерева. Після побудови всіх дерев робляться прогнози для нових даних, і ефективність кожного дерева залежить від того, наскільки точним воно було на тренувальних даних.

Так як в цьому алгоритмі велика увага приділяється виправленню помилок моделей, важливо, щоб в даних були відсутні аномалії.

2.3 Висновок до другого розділу

Машинне навчання - (Machine Learning) великий підрозділ штучного інтелекту, що вивчає методи побудови алгоритмів, які здатні навчатися. Розрізняють два типи навчання: навчання по прецедентах, або індуктивне навчання, засноване на виявленні закономірностей в емпіричних даних; дедуктивне навчання передбачає формалізацію знань експертів і їх перенесення в комп'ютер у вигляді бази знань. Дедуктивне навчання прийнято відносити до області експертних систем, тому терміни машинне навчання і навчання по прецедентах можна вважати синонімами. Багато методів індуктивного навчання розроблялися як альтернатива класичним статистичним підходам.

3 ПРОЕКТУВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ ПРОГНОЗУВАННЯ ВАРТОСТІ ЖИТЛА НА ОСНОВІ ПОСЛІДОВНОЇ МОДЕЛІ МАШИННОГО НАВЧАННЯ З ВИКОРИСТАННЯМ БІБЛІОТЕКИ TENSORFLOW.JS

3.1 Послідовна модель машинного навчання

Послідовності - структура даних, де кожен приклад може розглядатися як серія із точок даних. Наприклад, дане речення: "Я в даний час читаю розділ про послідовне моделювання нейронних мереж", є прикладом, який складається з декількох слів, і слова залежать один від одного. Це саме стосується також медичних записів. Одна єдиний медичний запис складається з багатьох вимірювань за певну одиницю часу. Це - те ж для мовних форм хвилі.

По факту, алгоритми машинного навчання вимагають, аби вхідний текст був представлений вектором з фіксованою довжиною. Модель (мережа) повинна бути натренованою та вираженою за допомогою різних алгебраїчних операцій над матрицею вхідних данних.

Напевно, найбільш поширеною репрезентацією вектора з фіксованою довжиною для тексту є модель «торба слів» завдяки своїй простоті, ефективності та зазвичай вражаючій точності. Тим не менш, «торба слів» має і свої недоліки:

1. По-перше, втрачається порядок слів, і тому різні речення можуть мати точно однакове представлення, якщо вживаються одні й ті ж слова. Приклад: "Їжа була гарною, зовсім не поганою." Проти "Їжа була поганою, зовсім не гарною". Незважаючи на те, що «сумка» з n-грамів розглядає порядок слів у короткому контексті, він страждає від розрідженості даних та високої розмірності.
2. Крім того, «мішок із слів» і «мішок з n-грамів» мають дуже мало знань про семантику слів або, більш формально, відстані між словами. Це

означає, що слова "потужний", "сильний" і "Париж" однаково віддалені, незважаючи на те, що семантично "потужний" повинен бути ближче до "сильного", ніж "Париж".

3. Люди не починають своє мислення з нуля щосекунди. Читаючи цю інформацію, читач розуміє кожне слово, грунтуючись на розумінні попередніх слів. Традиційні нейронні мережі не можуть цього зробити, і це здається головним недоліком. «Мішок слів» і «мішок-n-грамів», як текстові подання, не дозволяють відслідковувати довгострокові залежності всередині одного і того ж речення або абзацу.

4. Ще одним недоліком моделювання послідовностей з традиційними нейронними мережами (наприклад, Feedforward Neural Networks) є факт нерозподілення параметрів протягом часу. Візьмемо для прикладу ці два речення: "У понеділок сніг" та "Був сніг у понеділок". Ці речення означають те саме, хоча деталі знаходяться в різних частинах послідовності. Насправді, коли ми подаємо ці два речення в нейронну мережу Feedforward для завдання передбачення, модель призначатиме різні ваги "Понеділку" в кожен момент часу. Параметри обміну надають мережі можливість шукати задану функцію скрізь у послідовності, а не лише у певній області.

Аби побудувати послідовність потрібен специфічний фреймворк машинного навчання, який зможе:

1. Справлятись з послідовностями різної довжини;
2. Підтримувати порядок послідовності;
3. Слідкувати за довгостроковими залежностями, а не просто скорочувати вхідні дані;
4. Обмінюватись параметрами по всій послідовності (аби повторно не вивчати речі)

Рекурентна нейронна мережа може справитись із цим заданням. Вона являє собою мережу із циклів, які можуть зберігати інформацію. Рекурентна нейронна мережа побудована так само як і «традиційна» нейронна мережа. Є

деякі входи, є декілька прихованих шарів і є декілька виходів. Це зображено на рисунку 3.1.

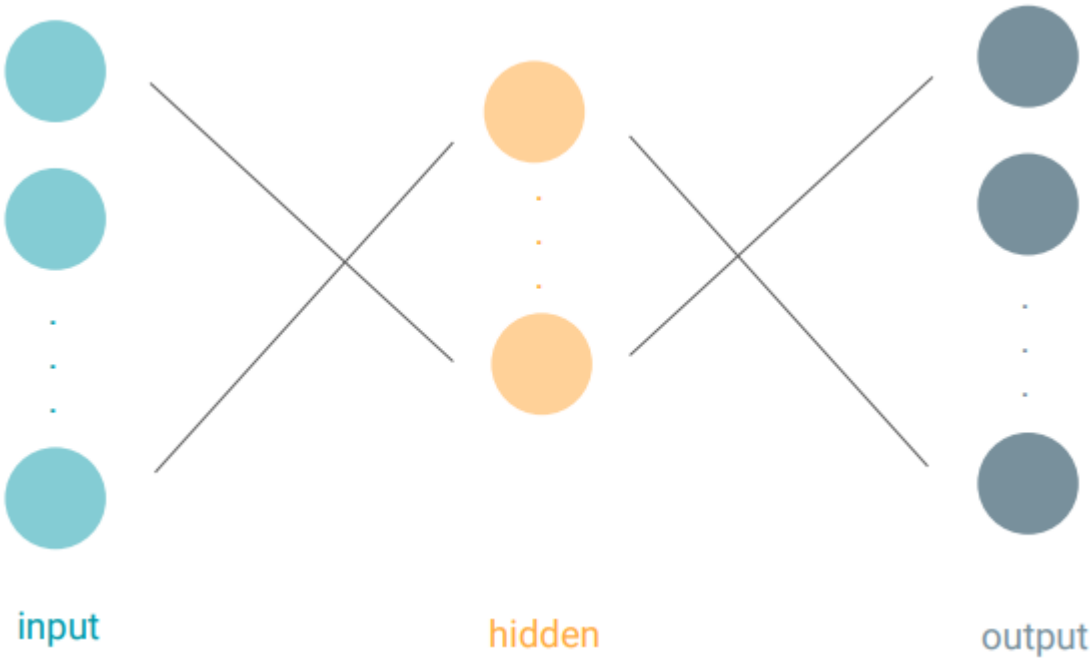


Рисунок 3.1 – Базова схема нейронної мережі

Єдина відмінність полягає в тому, що кожен прихований блок виконує дещо іншу функцію. Періодичний прихований блок обчислює функцію входу і власний попередній вихід, також відомий як стан комірки. Для текстових даних введенням може бути вектор, що представляє слово x(i) у реченні з n слів (також відоме як вбудовування слова). На рисунку 3.2 зображена функція прихованого блоку.

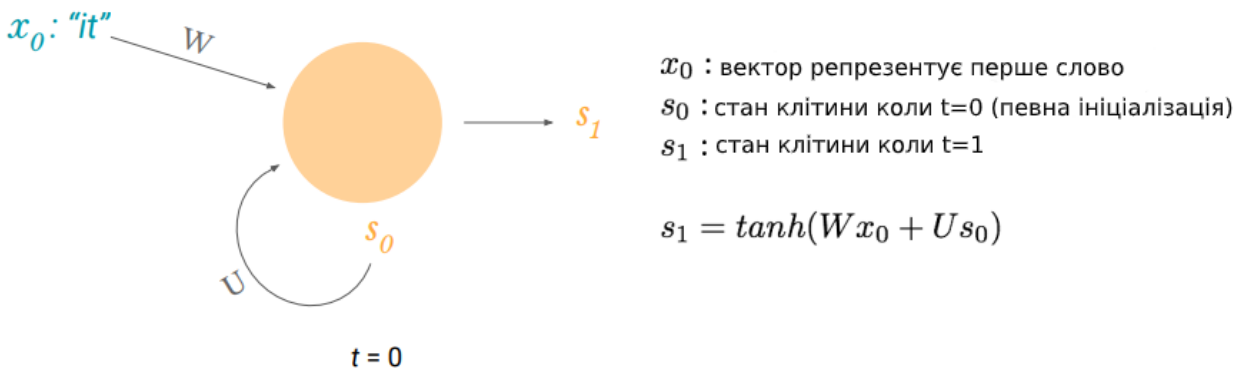


Рисунок 3.2 – Функція прихованого блоку

W та U є ваговими матрицями, а tanh є гіперболічною дотичною функцією. Аналогічно, на наступному кроці він обчислює функцію нового вводу та його попередній стан комірки: s2 = tanh (Wx1 + Us1. Така поведінка схожа на приховану одиницю в мережі передачі каналу. Різниця, властива послідовностям, полягає в тому, що додається додатковий термін, щоб включити його власний попередній стан.

Поширений спосіб перегляду періодичних нейронних мереж - розгортання їх у часі. Можна помітити, що використовуються однакові вагові матриці W і U протягом усієї послідовності. Це вирішує проблему спільного використання параметрів. Немає нових параметрів для кожної точки послідовності. Таким чином, як тільки ми щось дізнаємось, це може застосовуватися в будь-якій точці послідовності. На рисунку 3.3 зображено розгортання нейронної мережі у часі.

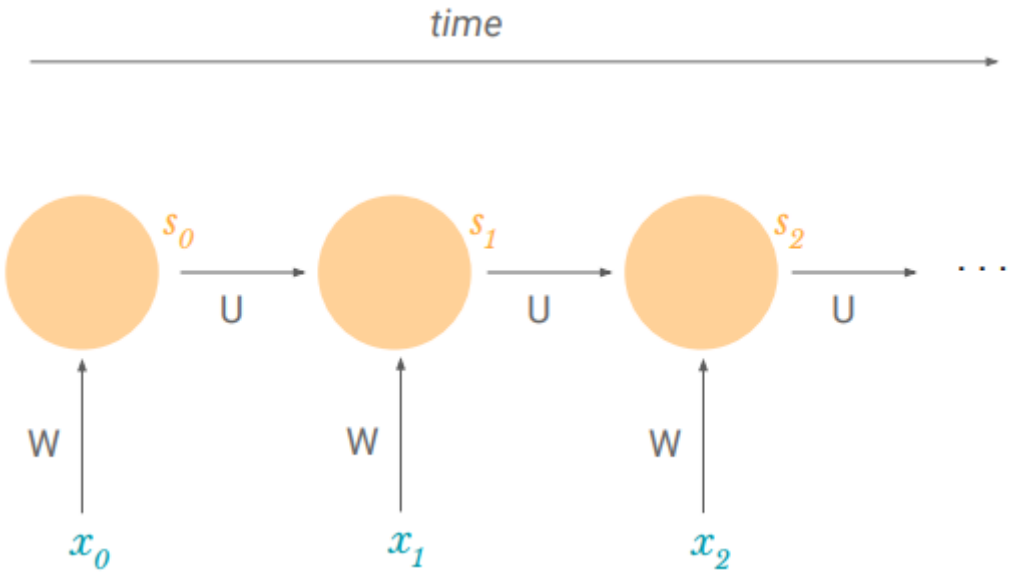


Рисунок 3.3 – Розгортання нейронної мережі у часі

Факт відсутності нових параметрів для кожної точки послідовності також допомагає мати справу з послідовностями змінної довжини. У випадку послідовності, яка має довжину 4, можна розкрутити цей RNN до чотирьох часових кроків. В інших випадках можна розкрутити його до десяти часових кроків, оскільки довжина послідовності не визначена в алгоритмі. Під

розгортанням, просто мається на увазі, що виписується мережа для повної послідовності. Наприклад, якщо послідовність, яка розглядається, - це речення з 5 слів, мережа буде розкручена в 5-шарову нейронну мережу, по одному шару для кожного слова. Тепер, коли відомо, як працює одна прихована одиниця, слід розібратися, як тренувати цілу періодичну нейронну мережу, що складається з безлічі прихованих одиниць і навіть багатьох шарів багатьох прихованих одиниць.

Послідовні моделі при контрольованому навчанні можуть використовуватися для вирішення різноманітних питань, включаючи прогнозування фінансових часових рядів, розпізнавання мови, створення музики, класифікацію настроїв, машинний переклад та розпізнавання відео активності. Єдине обмеження полягає в тому, що або вхід, або вихід має бути послідовним. Іншими словами, можна використовувати послідовні моделі для вирішення будь-якого типу проблеми контрольованого навчання, яка містить часовий ряд або в вхідному, або у вихідному шарах.

Традиційні нейронні мережі, що подаються, не мають функцій у різних положеннях мережі. Іншими словами, ці моделі припускають, що всі входи (і виходи) не залежать один від одного. Ця модель не працюватиме в порядку передбачення послідовності, оскільки попередні входи мають важливе значення для прогнозування наступного результату. Наприклад, якби читач передбачав наступне слово в потоці тексту, він б хотів знати хоча б пару слів перед цільовим словом. На рисунку 3.4 зображена схема алгоритму послідовної моделі. Традиційні нейронні мережі вимагають, щоб довжини послідовностей вводу та виводу були постійними для всіх прогнозів.

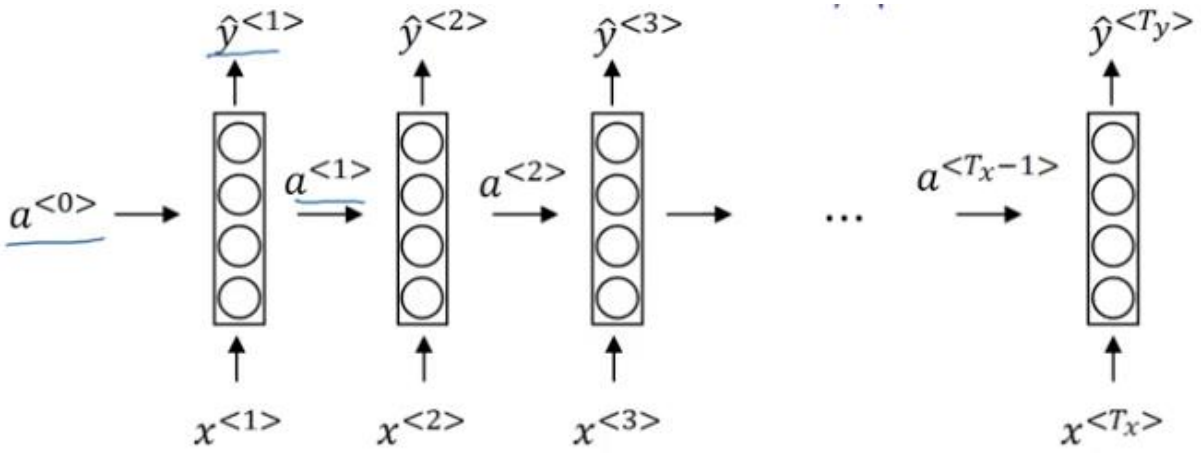


Рисунок 3.4 – Схема алгоритму послідовної моделі

Як було сказано вище, послідовна модель машинного навчання може вирішувати абсолютно різноманітні завдяки своєму прогнозуванню послідовностей. Існують різні типи мереж послідовної моделі машинного навчання,наприклад, мережі "один до одного", "один до багатьох", "багато до одного" і "багато до багатьох". На рисунку 3.5 зображені дані типи мереж.

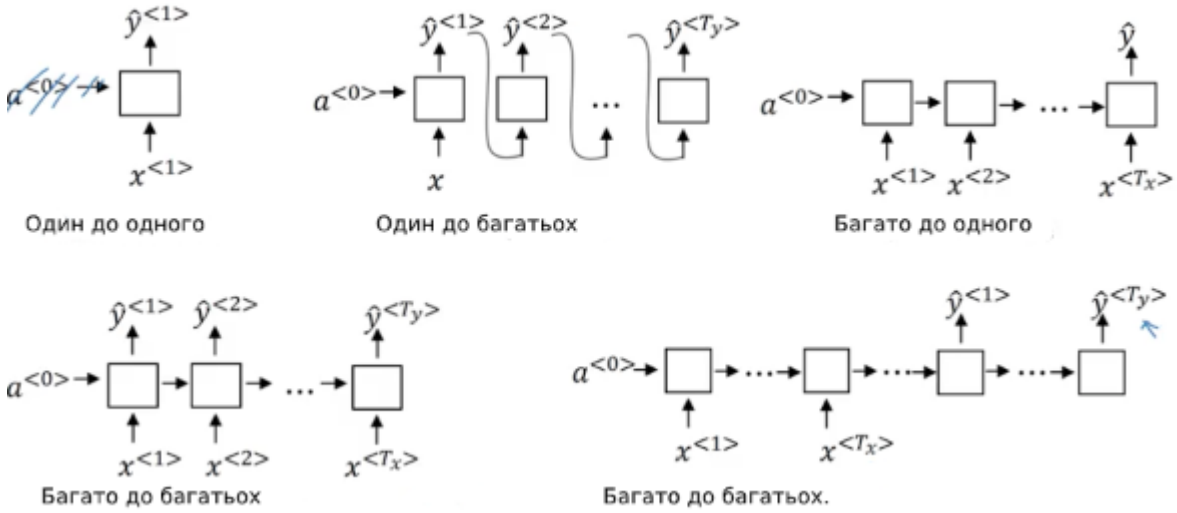


Рисунок 3.5 – Типи мереж у послідовній моделі машинного навчання

У музичному поколінні вхід може бути порожнім набором, і продукція може бути піснею (один до багатьох). У високочастотному фінансовому прогнозуванні мінливості вхід може бути потоком цитат за минулі 3 хвилини, і продукція була б нестабільним прогнозом (багато до одного). Найбільш цікаво, що «багато до багатьох» архітектура може використовуватись

застосунками, де довжини послідовності входу та виходу не одинакові, та з використанням енкодера / декодера, це показано в правому нижньому куті рисунка 3.5. Моделі «багато до багатьох» зазвичай згадуються як моделі типу «послідовності до послідовності» в літературі.

Лінгвістичні моделі роблять прогнози, оцінюючи ймовірність наступного слова, беручи до уваги слова, які передують йому. Після того, як одна лінгвістична модель є навченою, умовні розподіли, які були оцінені моделлю, можуть бути використані новими типовими послідовностями. На рисунку 3.6 зображена RNN модель (рекурентна нейронна мережа), яка використовується у лінгвістичному прогнозуванні.

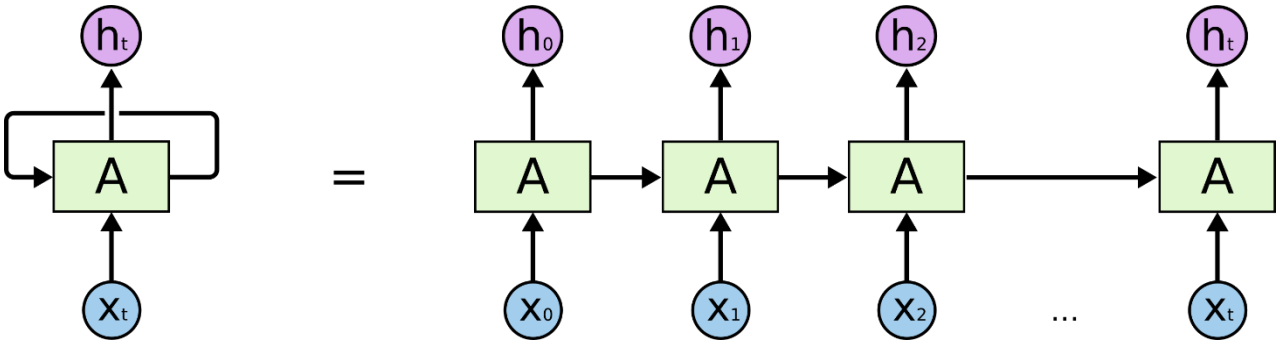


Рисунок 3.6 – Схема рекурентної нейронної мережі

У рекурентній нейронній мережі можуть бути градієнти, які зникають еспоненційно швидко, заважаючи цим нейронній мережі навчатись на довготривалих залежностях. Зникаючі градієнти також важко найти, що спричиняє певну небезпеку при розгортанні системи в продукцію.

3.2 Проектування навчальної моделі.

Хоча мова програмування Python або R має відносно просту криву навчання, веб-розробники із задоволенням роблять усе, що знаходиться в зоні комфорту JavaScript. Враховуючи тенденцію, яку розпочав Node.js - застосовувати JavaScript будь де, де тільки можна, я вирішив зрозуміти концепт машинного навчання за допомогою JS. Python став популярним через

велику кількість доступних пакетів та бібліотек, але спільнота JS нічим не поступається.

TensorFlow.js - це бібліотека JavaScript, розроблена Google для навчання та використання моделей машинного навчання (ML) у браузері. Це супутня бібліотека до TensorFlow, популярної бібліотеки ML для Python. Читайте далі, щоб дізнатися про його особливості, майбутнє та про те, як це може вам допомогти.

У даному розділі буде продемонтсровано процес розробки веб-застосунку, який використовує TensorFlow.js, аби навчити модель у браузері. Модель навчиться передбачати ціну будинку в залежності від кількості кімнат. Для цього потрібно:

- Завантажити дані і підготувати їх до навчання;
- Визначити архітектуру моделі;
- Навчити модель та спостерігати за її розвитком під час тренування;
- Оцінити навчену модель, зробивши певні прогнози.

Для початку слід створити кореневий HTML файл і підключити до нього JavaScript-файл script.js.

```
<!DOCTYPE html>
<html
<head>
  <title>TensorFlow.js Tutorial</title>  <!-- Import
TensorFlow.js -->
  <script
src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@1.0.0/dist/tf.mi
n.js"></script>
  <!-- Import tfjs-vis -->
  <script
src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs-
vis@1.0.2/dist/tfjs-vis.umd.min.js"></script>  <!-- Import the main
script file -->
  <script src="script.js"></script></head><body>
```

```
</body>
</html>
```

Самі дані для тренування будуть братись з JSON-файлу.

```
{
   "Avg. Area Income": 79545.45857,
   "Avg. Area House Age": 5.682861322,
   "AvgAreaNumberofRooms": 7.009188143,
   "Avg. Area Number of Bedrooms": 4.09,
   "Area Population": 23086.8005,
   "Price": 1059033.558,
   "Address": "208 Michael Ferry Apt. 674\nLaurabury, NE
37010-5101"
 },
 {
   "Avg. Area Income": 79248.64245,
   "Avg. Area House Age": 6.002899808,
   "AvgAreaNumberofRooms": 6.730821019,
   "Avg. Area Number of Bedrooms": 3.09,
   "Area Population": 40173.07217,
   "Price": 1505890.915,
   "Address": "188 Johnson Views Suite 079\nLake Kathleen,
CA 48958"
 }
```

Вище наведено приклад даних, які вміщує даний JSON-файл. Серед цих всіх методів у нашому випадку будуть використовуватись методи «AvgAreaNumberofRooms» та «Price», які охарактеризовують кількість кімнат та ціну відповідно.

До файлу script.js додається асинхронна функція getData()

```
async function getData() {
const houseDataReq = await
fetch('https://raw.githubusercontent.com/meetnandu05/ml1/master/house.jso
```

```
n');
const houseData = await houseDataReq.json();
const cleaned = houseData.map(house => ({
price: house.Price,
rooms: house.AvgAreaNumberofRooms,
}))
.filter(house => (house.price != null && house.rooms != null));
return cleaned;
}
```

Дана функція видалить всі записи, у яких не визначається ціна, чи кількість кімнат. Далі слід внести дані, які будуть використані для тренування, на точкову діаграму. Для цього у script.js файл добавляється ще одна асинхронна функція run().

```
async function run() {
  // завантаження данних на точкову діаграму
  const data = await getData();
  const values = data.map(d => ({
    x: d.rooms,
    y: d.price,
  }));  tfvis.render.scatterplot
    {name: 'No.of rooms v Price'},
    {values},
    {
      xLabel: 'No. of rooms',
      yLabel: 'Price',
      height: 300
    }
  );  // нижче буде додано більше коду
}document.addEventListener('DOMContentLoaded', run);
```

Після оновлення сторінки у браузері появляється панель з точковою діаграмою. Це зображено на рисунку 3.7.

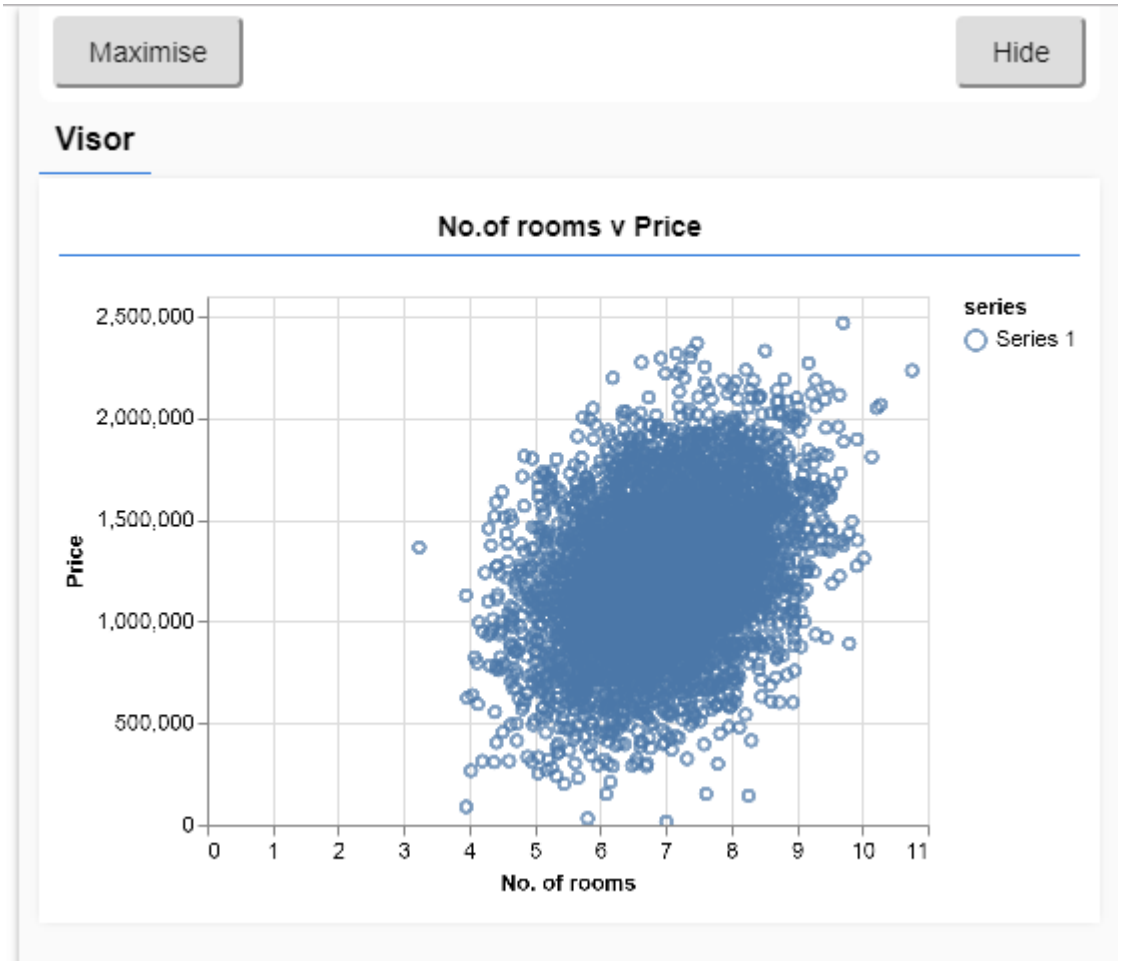


Рисунок 3.7 – Точкова діаграма

Як правило, під час роботи з даними корисно знайти можливість візуалізувати дані та при необхідності очистити їх. Візуалізація даних може дати зрозуміти, чи є якась структура даних, яку модель може вивчити. На рисунку 3.7 видно, що існує позитивна кореляція між кількістю кімнат та ціною, тобто якщо кількість кімнат збільшується, то ціни на будинки в цілому зростають.

Модель машинного навчання отримає вхідні дані, а потім видасть вихідні. Але спершу для цього слід побудувати нейронну мережу. Для цього слід у файл script.js добавити функцію createModel()

```
function createModel() {
  // створення послідовної моделі
  const model = tf.sequential();
```

```
        // Добавлення одиночного прихованого шару
        model.add(tf.layers.dense({inputShape: [1], units: 1,
useBias: true}));
        // Добавлення вихідного шару
        model.add(tf.layers.dense({units: 1, useBias: true}));
return model;
      }
```

Це одна з видів моделей, яку можна визначити за допомогою TensorFlow.js

```
      const model = tf.sequential();
```

Це створює об'єкт tf.Model. Ця модель є послідовною, оскільки її входи надходять прямо до її виходу. Інші моделі можуть мати гілки, або навіть кілька входів і виходів, але у багатьох випадках моделі будуть послідовними.

```
      model.add(tf.layers.dense({inputShape: [1], units: 1,
useBias: true}));
```

Це додає прихований шар до мережі. Оскільки це перший рівень мережі, необхідно визначити вхідну форму.inputShape є [1], тому що ми маємо 1 число як наш вхід (кількість кімнат даного будинку).

Units встановлює, наскільки великою буде матриця ваги в шарі. Встановивши його тут на 1, мається на увазі, що буде 1 вага для кожної з вхідних функцій даних.

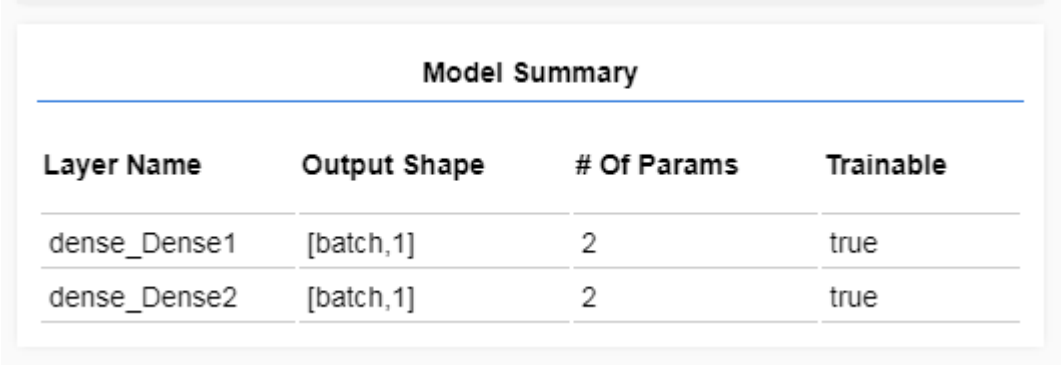
```
      model.add(tf.layers.dense({units: 1}));
```

Даний метод створює вихідний рівень. Встановлюємо, що units є 1, оскільки потрібно вивести одне число.

Основну функцію run() слід доповнити наступним кодом

```
      // створення моделі
const model = createModel();
tfvis.show.modelSummary({name: 'Model Summary'}, model);
```

Це створить екземпляр моделі та покаже зведення шарів на веб-сторінці. Екземпляр моделі зображений на рисунку 3.8.



Model Summary

Layer Name	Output Shape	# Of Params	Trainable
dense_Dense1	[batch,1]	2	true
dense_Dense2	[batch,1]	2	true

Рисунок 3.8 – Екземпляр моделі

3.3 Підготовка даних до навчання

Для отримання переваг продуктивності TensorFlow.js, які роблять практичні моделі машинного навчання практичними, потрібно перетворити дані в тензори. Для цього у файл script.js додається функція convertToTensor()

```
function convertToTensor(data) {
  return tf.tidy(() => {
    // Step 1. Змішання даних
    tf.util.shuffle(data);
    // Step 2. Конвертування даних в тензори
    const inputs = data.map(d => d.rooms)
    const labels = data.map(d => d.price);
    const inputTensor = tf.tensor2d(inputs, [inputs.length,
1]);
    const labelTensor = tf.tensor2d(labels, [labels.length,
1]);    //Step 3. Нормалізація даних за в ряд 0 - 1 з використанням
min-max масштабування
    const inputMax = inputTensor.max();

    const inputMin = inputTensor.min();
    const labelMax = labelTensor.max();
    const labelMin = labelTensor.min();
```

```
        const normalizedInputs =
inputTensor.sub(inputMin).div(inputMax.sub(inputMin));
          const normalizedLabels =
labelTensor.sub(labelMin).div(labelMax.sub(labelMin));
         return {
            inputs: normalizedInputs,
            labels: normalizedLabels,
            //Повернення  min/max границь для подальшого
використання.
            inputMax, inputMin,
            labelMax, labelMin,
          }
        });
      }
```

Під час навчання моделі набір даних поділяється на менші набори, кожен набір відомий як пакет.

```
      // Step 1. Змішання даних
  tf.util.shuffle(data);
```

Ці партії потім подаються на модель для тренувань. Перемішування даних важливо, оскільки модель не повинна отримувати однакові дані знову і знову. Якщо модель отримує однакові дані знову і знову, модель не зможе узагальнити дані та дати результати, визначені на входах, отриманих під час навчання. Перемішування допоможе, маючи різноманітні дані в кожній партії.

```
      // Step 2. Конвертування даних в тензори
          const inputs = data.map(d => d.rooms)
          const labels = data.map(d => d.price);
          const inputTensor = tf.tensor2d(inputs, [inputs.length,
1]);
    const labelTensor = tf.tensor2d(labels, [labels.length, 1]);
```

Тут створюються два масиви, один для вхідних прикладів (кількість записів номерів) та інший для справжніх вихідних значень (які відомі як мітки

в машинному навчанні, а в данному випадку - ціна кожного будинку). Потім перетворюємо кожен масив даних у 2d тензор.

```
      //Step 3. Нормалізація даних за в ряд 0 – 1 з використанням
min-max масштабування
          const inputMax = inputTensor.max();
          const inputMin = inputTensor.min();
          const labelMax = labelTensor.max();
          const labelMin = labelTensor.min();
        const normalizedInputs =
inputTensor.sub(inputMin).div(inputMax.sub(inputMin));
    const normalizedLabels =
labelTensor.sub(labelMin).div(labelMax.sub(labelMin));
   return {
      inputs: normalizedInputs,
      labels: normalizedLabels,
```

Далі слід нормалізувати дані. Тут дані нормалізовуються в числовий діапазон 0-1, використовуючи мінімальне масштабування. Нормалізація важлива, оскільки внутрішність багатьох моделей машинного навчання, які будуються за допомогою tensorflow.js, розроблена для роботи з не надто великими числами. Загальні діапазони для нормалізації даних включають 0 до 1 або -1 до 1.

```
      return {
      inputs: normalizedInputs,
      labels: normalizedLabels,
      //Повернення  min/max границь для подальшого використання.
      inputMax,
      inputMin,
      labelMax,
      labelMin,
    }
```

Слід зберегти значення, які використовувались для нормалізації під час тренінгу, щоб мати змогу «ненормалізувати» результати, щоб повернути їх у

початковий масштаб і дозволити нормалізувати майбутні вхідні дані таким же чином.

3.4 Підготовка моделі до навчання та прогнозування

Слід запустити тренувальний цикл. Це модна зробити наступним чином:

```
      return model.fit(inputs, labels, {
batchSize,
epochs,
callbacks: tfvis.show.fitCallbacks(
{ name: 'Training Performance' },
['loss', 'mse'],
{
height: 200,
callbacks: ['onEpochEnd']
}
)
});
```

model.fit - це функція, яку викликається для запуску навчального циклу. Це асинхронна функція, тому слід повертаємо «обіцянку», яку вона дає, щоб абонент міг визначити, коли навчання закінчено.

Для відстеження прогресу навчання деякі зворотні дзвінки передаються до model.fit. Використовуємо tfvis.show.fitCallbacks для створення функцій, які будують графіки для метрики "втрата" та "mse", яка була вказана раніше.

Тепер слід звернутись до функцій, які були визначені у функції run()

```
      // Конвертування даних до форми для тренування
const tensorData = convertToTensor(data);
const {inputs, labels} = tensorData;
// Навчання моделі
```

```
await trainModel(model, inputs, labels);
console.log('Done Training');
```

Після оновлення сторінки, з затримкою в декілька секунд появляться графіки продуктивності навчання, які зображені на рисунку 3.9.

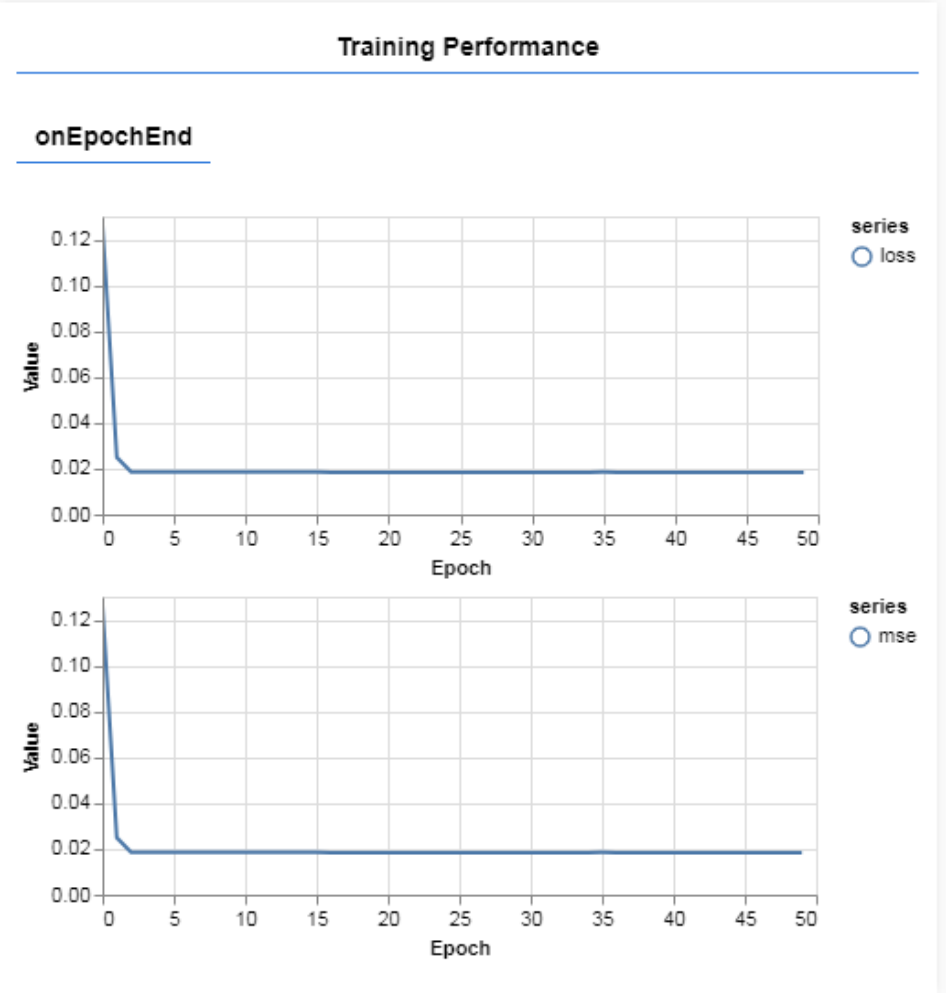


Рисунок 3.9 – Графіки продуктивності навчання

Вони створюються зворотними дзвінками, які були створені раніше. Дані графіки відображають втрати (на останній партії) та mse (на весь набір даних) в кінці кожної епохи. Тренуючи модель, було б бажано, щоб втрати зменшувалися. У цьому випадку, оскільки наша метрика є мірою помилки, також хочеться, щоб вона знижувалася.

Тепер, коли модель навчена, слід робити деякі прогнози. Спершу слід оцінити модель, побачити, що вона прогнозує для рівномірного діапазону номерів

від низької до великої кількості кімнат. Для цього потрібно використати функцію testModel()

```
function testModel(model, inputData, normalizationData) {
const {inputMax, inputMin, labelMin, labelMax} = normalizationData;
// Генерування передбачень для ряду цифр між 0 та 1

// Ненормалізація даних інверсією min-max масштабуванням
const [xs, preds] = tf.tidy(() => {
const xs = tf.linspace(0, 1, 100);
const preds = model.predict(xs.reshape([100, 1]));
const unNormXs = xs
.mul(inputMax.sub(inputMin))
.add(inputMin);
const unNormPreds = preds
.mul(labelMax.sub(labelMin))
.add(labelMin);
//Ненормалізація даних
return [unNormXs.dataSync(), unNormPreds.dataSync()];
});
const predictedPoints = Array.from(xs).map((val, i) => {
return {x: val, y: preds[i]}
});
const originalPoints = inputData.map(d => ({
x: d.rooms, y: d.price,
}));
tfvis.render.scatterplot(
{name: 'Model Predictions vs Original Data'},
{values: [originalPoints, predictedPoints], series: ['original',
'predicted']},
{
xLabel: 'No. of rooms',
yLabel: 'Price',
height: 300
});}
```

Декілька заміток стосовно доданого коду.

```
        const xs = tf.linspace(0, 1, 100);
const preds = model.predict(xs.reshape([100, 1]));
```

Тут генерується 100 нових "прикладів" для подачі моделі. Model.predict - це те, як подаються ці приклади в модель. Зауважте, що вони повинні мати подібну форму ([num_examples, num_features_per_example]).

```
        //Ненормалізація даних
const unNormXs = xs
.mul(inputMax.sub(inputMin))
.add(inputMin);
const unNormPreds = preds
.mul(labelMax.sub(labelMin))
.add(labelMin);
```

Щоб повернути дані до початкового діапазону (а не 0–1), використовувались значення, які обчислювались під час нормалізації, простою інвертацією операції.

```
        return [unNormXs.dataSync(), unNormPreds.dataSync()];
```

.dataSync() - це метод, який можна використати для отримання типу набору значень, що зберігаються в тензорі. Це дозволяє обробляти ці значення у звичайному JavaScript. Це синхронна версія методу .data (), яка, як правило, є кращою.

Нарешті, можна використовуємо tfjs-vis для побудови вихідних даних та прогнозів з моделі.

В кінці слід добавити наступний код для запуску run() функції.

```
        testModel(model, data, tensorData);
```

Після цього слідоновити сторінку, і можна побачити передбачення, як тільки модель закінчить навчання. Це зображено на рисунку 3.10.

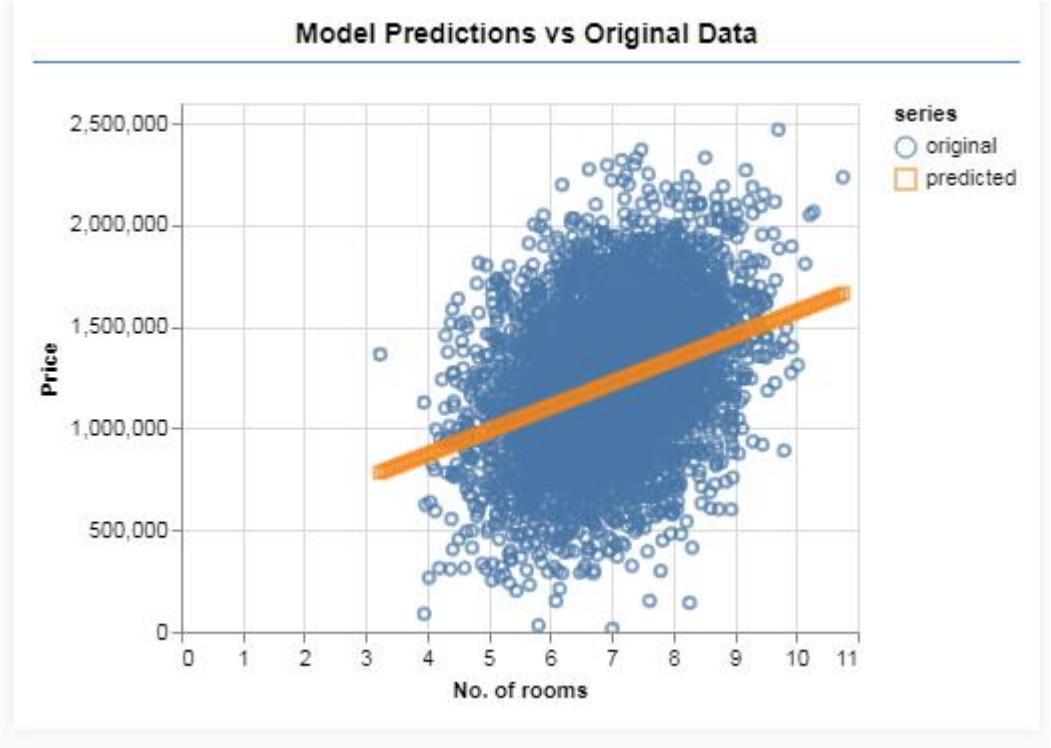


Рисунок 3.10 – Прогнозована модель

3.5 Висновок до третього розділу

Запуск машинного навчання в браузері розглядає, що з точки зору користувача немає необхідності встановлювати які-небудь бібліотеки або драйвери. Просто варто відкрити веб-сторінку із програмою і вона вже готова до запуску. Крім того, все готово для роботи з прискорювачем на GPU. TensorFlow.js автоматично підтримує WebGL і сам використовує код, якщо є графічний процесор. Користувачі також можуть відкрити свою веб-сторінку з мобільного пристрою, і в цьому випадку модель може використовувати дані датчиків, наприклад, гіроскопа або акселерометра. Всі дані залишаються на стороні клієнта, що робить TensorFlow.js пригожим для виводу з низькою затримкою, а також для застосунків, які зберігають конфіденційність.

4 МАШИННЕ НАВЧАННЯ У БРАУЗЕРІ. ОГЛЯД TENSORFLOW.JS ТА ІНШИХ ПОПУЛЯРНИХ БІБЛІОТЕК БРАУЗЕРНОГО МАШИННОГО НАВЧАННЯ ТА ОБҐРУНТУВАННЯ ВИБОРУ

4.1 Машинне навчання у браузері

Останнім часом все більше уваги приділяється штучному інтелекту та машинному навчанню. На перший погляд ці концепції абсолютно не пов'язані з веб-розробкою та технологіями JavaScript. Зазвичай вони асоціюються з середовищем Python / R або навіть бібліотеками C ++. Одним з найпопулярніших фреймворків, яким користується все більша кількість розробників, є TensorFlow. Він був розроблений в Google в 2011 році. Він створений на мові C ++ і може використовуватись різними мовами, такими як Python, R або Java.

Тривалий час, використовуючи JavaScript, машинне навчання здійснювалося на стороні сервера. Навчена модель була розгорнута на сервері і, наприклад, доступна через протокол HTTP. Веб-додаток надсилав запит з необхідними даними за допомогою JS для отримання результату від сервера. У березні 2018 року з’явився TensorFlow.js і з його допомогою можна писати застосунки для машинного навчання / глибокого навчання за допомогою JavaScript, не використовуючи програм, які обробляються на стороні сервера. TensorFlow.js можна використовувати для визначення, навчання та запуску моделей машинного навчання повністю у браузері. З точки зору користувача, це дуже просто і зручно; не потрібно встановлювати бібліотеки чи драйвери. Слід просто відкрити веб-сторінку і програма готова до запуску.

Як відомо, JavaScript є однопоточним і виконується лише на центральному процесорі, який призначений для переключення між програмами та задач з великою затримкою, а не для високої пропускної здатності. Графічний процесор розроблений для високих навантажень та для високої пропускної здатності. З точки зору розробки, різниця між цими

процесорами у тому, що на графічному процесорі робота буде виконуватись паралельно та багатопоточно і завдяки цьому обчислення, які обробляються графічним процесором є у багато раз швидшими, ніж якби вони оброблялись центральним процесором. Tensorflow.js, як і інші бібліотеки машинного навчання у браузері, автоматично підтримує WebGL, який є браузерним інтерфейсом для OpenGl, і дозволяє виконання JavaScript коду за допомогою графічного процесора.

Також машинне навчання у браузері має низку переваг. Оскільки, програма не відсилає ніяких запитів до сервера для обробки, користувач може не перейматись за коонфіденційність своїх даних. Застосунки можуть бути відкриті у будь-якому сучасному браузері та на будь-якій операційній системі. Також завдяки відсутності обробки даних на сервері, різні затримки по типу «запит – відповідь» будуть відсутні, що збільшує швидкість обробки даних.

Javascript зараз використовується скрізь - від корпоративних до персональних проектів, він є також покращений такою підмножиною мов як Typescript та ReasonML. Не зважаючи на те, що Javascript займає третє місце за популярністю серед мов програмування, які використовуються для машинного навчання, багато розробників надають перевагу мові програмування Python. Javascript сильно розвинулась в якості мови машинного навчання в період з 2018 року, хоча безліч проектів залишаються непоміченими серед спільноти.

4.2 Порівняння TensorFlow.js з іншими бібліотеками браузерного машинного навчання та обґрунтування вибору

Brain.js - бібліотека JavaScript, яка працює на Node.js та в браузерах. Вихідний код brain.js розміщується на GitHub і випускається під ліцензією MIT. Поточна версія (1.6.1) забезпечує підтримку повторюваних нейронних мереж (RNN), довготривалу короткочасну пам'ять та закриту рекурентну мережу. На рисунку 4.1 зображений логотип brain.js [32].

Рисунок 4.1 – Логотип brain.js

Навчальна програма спочатку потребує підключення необхідних бібліотек, а потім викликаються утилітні методи, щоб отримати зображення та мітки для наборів даних про навчання та тестування набору даних.

Дану бібліотеку створювали, маючи на меті зменшити складність машинного та глибокого навчань. Brain,js позиціонується як найпростіша бібліотека машинного навчання.

Machinelearn.js - нова бібліотека для машинного навчання на базі мови програмування Javascript. Дана бібліотека по своєму функціоналу та можливостям схожа на бібліотеку ScikitLearn, яка використовує мову програмування Python. Вона надає прості та критичні для поставленого завдання моделі та утиліти для контрольованих та непідконтрольних проблем. Орієнтуючись на простоту та доволі широкий функціонал для розробників Javascript та Typescript, вона забезпечує кластеризацію, декомпозицію,беггінг, лінійні моделі, виділяння ознак, тощо. На рисунку 4.2 зображений логотип Machinelearn.js [33].

Рисунок 4.2 – Логотип Machinelearn.js

Дана бібліотека використовує вже існуючі популярну бібліотеку Tensorflow.js для своїх основних розрахунків, яка дозволяє прискорити використання рідного інтерфейсу C ++, CUDA та WebGL.

Math.js - це бібліотека для всіх математичних потреб у Javascript з великими API лінійної алгебри, включаючи матричні операції та основну математику. Він досить легкий, оскільки не залежить від інших прискорювальних прийомів, таких як WebAssembly або WebGL. На рисунку 4.3 зображений логотип Math.js [34].

Рисунок 4.3 – Логотип Math.js.

У галузі машинного навчання Javascript Math.js достатньо, щоб виконати роль утиліти Numpy (розширення мови Python), оскільки вона надає всі необхідні утиліти, необхідні для вирішення проблем машинного навчання.

Face-api.js – це бібліотека, яка включає в себе реалізацію відомих моделей виявлення та розпізнавання облич, попередньо навчених на широкому спектрі наборів даних. Це дає готове до використання API, яке можна безпосередньо підключити до будь-якого середовища Node.js та

браузера. На рисунку 4.4 зображено приклад використання бібліотеки Face-api.js.

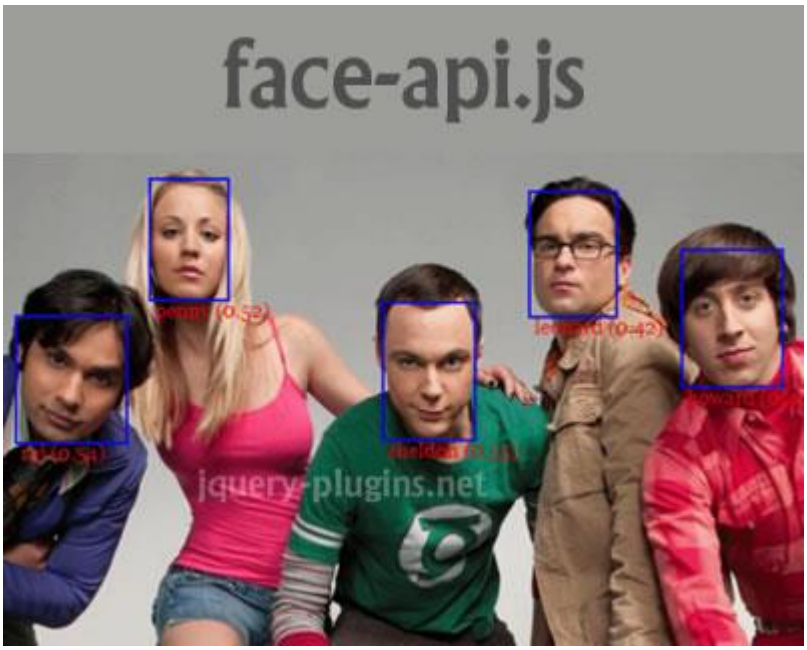


Рисунок 4.4 – Приклад використання бібліотеки Face-api.js

Дана бібліотека реалізовує SSD MobileNet V1, Tiny Face Detector та MTCNN для архітектури виявлення та архітектуру, подібну до ResNet-34 для розпізнавання обличчя. Будучи полегшеною за допомогою Tensorflow.js, бібліотеку можна без проблем використовувати як у мобільних, так і веб-браузерах [35].

Дана бібліотека використовується виключно для розпізнавання облич, тобто спектр її використання обмежений.

R.js - це бібліотека, яка базується на математичній утиліті, яка написана мовою програмування R. Важливі компоненти мови R були іммпортовані у мову TypeScript. На рисунку 4.5 зображений логотип R.js [36].

Рисунок 4.5 – Логотип R.js

Специфікація лінійної алгебри, наприклад як Basic Linear Algebra Subprograms (BLAS - Основні підпрограми з лінійної алгебри), широко використовується утилітами машинного навчання, як Numphy чи R.

Не зважаючи на переваги даної бібліотека, вона має й багато недоліків, наприклад як складність роботи з нею, невелика спільнота, та вичерпна документація.

Stdlib-js – це бібліотека, головна специфіка якої – побудова поширених статичних моделей, базуючись на наборі даних різних типів, включаючи бінарну класифікацію, регресію та дані, потрібні для формування машинного навчання, що допомагає розробникам застосунків машинного навчання швидше переходити на наступні етапи роботи над застосунком. На рисунку 4.6 зображений логотип stdlib-js [37].

Рисунок 4.6 – Логотип stdlib-js

Іншими словами, це бібліотека, яка включає в собі вже готові рішення для створення моделей машинного навчання, які описані в документації. Дана бібліотека може бути чудовою супровідною бібліотекою для будь-якого проекту машинного навчання.

Tensorflow.js - це найпопулярніша бібліотека браузерного машинного навчання, яка побудована на базі бібліотеки Tensorflow, яка використовується мовою програмування Python [38].

Tensorflow.js у 2019 році стала основною бібліотекою для всіх проектів Javascript для машинного навчання завдяки своїй всебічній реалізації лінійної алгебри і реалізації шарів глибокого навчання. Вона швидко наздогнала бібліотеку Tensorflow, з якої була створена, за кількістю підтримуваних API методів, і майже будь-які проблеми в машинному навчанні можуть бути вирішені, використовуючидану бібліотеку в цей момент. На рисунку 4.6 зображений логотип Tensorflow.js.

Рисунок 4.7 – Логотип Tensorflow.js

Окрім забезпечення глибокого навчання та машинного навчання в середовищах Node.js, Tensorflow.js можна використовувати безпосередньо в браузерах, використовуючи WebGL для прискорень. Модель Tensorflow.js

сконструйована так, щоб вона підтримувалась не тільки браузерами, але й середовищами Node.js (тобто можна писати додатки з клієнт-серверною архітектурою). Завдяки цьому вона використовується багатьма іншими бібліотеками з відкритим кодом, наприклад як brain.js та machinelearn.js.

Крім того, tensorflow.js має утиліту tensorboard, яка призначена для роботи з візуалізацією даних. Дана утиліта є також корисною для дебаггінгу та порівняння різних циклів тренування. Tensorboard може збирати різні дані, збір яких може бути налаштований розробником під його потреби, які зберігаються у окремому модулі. Також на даній бібліотеці просто та зручно налаштований паралелізм процесів, який не вимагає втручання розробника у кожен його етап.

4.3 Висновок до четвертого розділу

Для написання дипломної роботи була обрана саме бібліотека Tensorflow.js. Вибір бібліотеки був здійснений за наступнеми критеріями:

- Вичерпність документації;
- Популярність бібліотеки;
- Можливості використання;
- Продуктивність та швидкодія;
- Складність написання самого коду, та наявність потрібних API методів.

Tensorflow.js на даний час має перевагу перед всіма вище перечисленими бібліотеками, тому доцільно використовувати саме її для написання застосунків для браузерного машинного навчання, чи машинного навчання на базі мов JavaScript чи TypeScript.

5 ОБҐРУНТУВАННЯ ЕКОНОМІЧНОЇ ЕФЕКТИВНОСТІ

Метою розділу є обґрунтування економічної ефективності розробки інформаційної системи для прогнозування вартості житла.

Щоб виконати оцінку економічної ефективності необхідно розрахувати трудомісткість реалізації проєкту, витрати на оплату праці найманим працівникам, витрати апаратного і програмного забезпечення, амортизаційні відрахування, витрати енергоресурсів та інші витрати які є основними пунктами виконання обчислень, а також показники економічної ефективності розробки проєкту.

5.1 Розрахунок норм часу на виконання науково-дослідної роботи

Ефективне використання часу має велике значення тому, що коефіцієнт корисної дії залежить від оптимального використання часу.

Кожен із етапів реалізації проєкту характеризується метою та змістом, оцінкою часу виконання, кількістю та спеціалізацією виконавців, а також приблизною оцінкою вартості.

Розробку інформаційної системи для прогнозування вартості житла поділено на декілька етапів, що дозволяє полегшити і структурувати виконання роботи.

Основні етапи при виконанні розробки інформаційної системи наступні:

1. Підготовка опису задачі.
2. Збір необхідної інформації для аналізу програмної системи.
3. Вибір програмного забезпечення.
4. Розробка алгоритму.
5. Розробка структури програми.
6. Тестування.

Для оцінки тривалості виконання окремих робіт використовують нормативи часу.

Виконавцем усіх операцій по розробці додатку виступає інженер - програміст.

Витрати часу по окремих операціях технологічного процесу відображені в таблиці 5.1.

Таблиця 5.1 – Операції технологічного процесу та їх час виконання

№ п/п	Назва операції (стадії)	Виконавець	Середній час виконання операції, год.
1	Підготовка опису задачі	Інженер-програміст	9
2	Збір необхідної інформації для аналізу програмної системи	Інженер-програміст	10
3	Вибір програмного забезпечення	Інженер-програміст	8
4	Розробка алгоритму	Інженер-програміст	85
5	Розробка структури програми	Інженер-програміст	90
6	Тестування	Тестувальник	65
Разом			267

Загальні затрати часу на реалізацію додатку становить 267 години, найбільше часу витрачено на реалізацію алгоритму виділення кісткових структур – 90 годин.

5.2 Визначення витрат на оплату праці та відрахувань на соціальні заходи

Відповідно до Закону України «Про оплату праці» заробітна плата – це «винагорода, обчислена, як правило, у грошовому виразі, яку власник або уповноважений ним орган виплачує працівникові за виконану ним роботу».

Розмір заробітної плати залежить від складності та умов виконуваної роботи, професійно-ділових якостей працівника, результатів його діяльності. Заробітна плата складається з основної та додаткової оплати праці.

Основна заробітна плата нараховується за виконану роботу за тарифними ставками, відрядними розцінками чи посадовими окладами.

Додаткова заробітна плата – це складова заробітної плати працівників, до якої включають витрати на оплату праці, не пов'язані з виплатами за фактично відпрацьований час. Нараховують додаткову заробітну плату залежно від досягнутих і запланованих показників, кваліфікації виконавців. Джерелом додаткової оплати праці є фонд матеріального стимулювання, який створюється за рахунок прибутку.

При розрахунку заробітної плати кількість робочих днів у місяці слід в середньому приймати – 24,5 дні/міс., або ж 196 год./міс. (тривалість робочого дня – 8 год.).

Наймані працівники для розробки додатку працюють згідно контракту, в якому вказано їхню погодинну ставку. Тобто розрахунок заробітної плати працівників відбуватиметься на базі тарифної ставки.

Тарифна ставка розробників додатків:

- Інженер – програміст – 72 грн./год
- Тестувальник – 35 грн./год

Основна заробітна плата розраховується за формулою:

$$Зосн. = Тс \cdot Кг, \quad (5.1)$$

де Тс – тарифна ставка, грн.; Кг – кількість відпрацьованих годин.

Оскільки всі види робіт виконує інженер - програміст, то основна заробітна плата буде розраховуватись тільки за однією формулою.

$$Зосн.= 72 \cdot 202 + 35 \cdot 65= 16819 \text{ грн.}$$

Додаткова заробітна плата становить 10–15 % від суми основної заробітної плати.

$$Здод. = Зосн. \cdot Кдопл. \quad (5.2)$$

де Кдопл – коефіцієнт додаткових виплат працівникам, 0,1–0,15 (візьмемо його рівним 0,15).

$$Здод = 16819 \cdot 0,15 = 2523 \text{ грн.}$$

Звідси загальні витрати на оплату праці (Во.п.) визначаються за формулою:

$$Во.п. = Зосн. + Здод. \quad (5.3)$$

$$Во.п. = 16819 + 2523 = 19342 \text{ грн.}$$

Крім того, слід визначити відрахування на соціальні заходи:

1) єдиний соціальний внесок ЄСВ(прибутковий податок) – 22%;

2) військовий збір – 1,5%.

у сумі зазначені відрахування становлять 23,5 %.

Отже, сума відрахувань на соціальні заходи буде становити:

$$Вс.з. = Фоп \cdot 0,235 \quad (5.4)$$

де Фоп – фонд оплати праці, грн.

$$Вс.з. = 19351 \cdot 0,235 = 4545 \text{ грн.}$$

Проведені розрахунки витрат на оплату праці наведено у таблицю 5.2.

Таблиця 5.2 – Розрахунки витрат на оплату праці

№з/п	Категорія працівників	Основна заробітна плата, грн.			Додаткова заробітна плата, грн.	Нарахув. на ФОП, грн.	Всього витрати на плату праці, грн. (6=3+4+5)
		Тарифна ставка, грн	Кількість відпрацьованих год.	Фактично нарах. з/пл., грн.			
А	Б	1	2	3	4	5	6
1.	Інженер-програміст	72	202	14544	2182	-	-
2.	Тестувальник	35	65	2275	341	-	-
Разом		107	267	16819	2523	4545	23887

З таблиці розрахунки витрат на оплату праці видно що всього витрати на плату праці становить 23887 грн.

5.3 Розрахунок матеріальних витрат

Матеріальні витрати визначаються як добуток кількості витрачених матеріалів та їх ціни:

$$Мві = qi \cdot pi \qquad (5.5)$$

де: qi– кількість витраченого матеріалу і–го виду; pi – ціна матеріалу і –го виду.

Звідси, загальні матеріальні витрати можна визначити:

$$Зм.в. = \sum Мві. \quad (5.6)$$

Розрахунки занесемо у таблицю 5.3.

Таблиця 5.3 – Розрахунки матеріальних витрат

Найменування матеріальних ресурсів	Один. Виміру	Норма витрат	Ціна за один., грн.	Затрати матер., грн.	Транспортно–заготівельні витрати, грн.	Загальна сума витрат на матер., грн.
1. Основні матеріали						
Оплата за користування інтернетом	Мбайти	–	–	75	–	150

Загальні матеріальні витрати на Internet становлять 150 грн.

5.4 Розрахунок витрат на електроенергію

Затрати на електроенергію 1-ці обладнання визначаються за формулою:

$$Зв = W \cdot T \cdot S \quad (5.7)$$

де W – необхідна потужність, кВт; T – кількість годин на реалізацію розробки; S – вартість кіловат-години електроенергії.

Вартість кіловат-години електроенергії слід приймати згідно існуючих на даний час тарифів. Отже, 1 кВт з ПДВ коштує 2,42 грн.

Потужність комп’ютера для розробки додатку – 400 Вт, кількість годин роботи обладнання згідно таблиці 5.1 –267 годин.

Тоді, Зв = 0,4 · 267 · 2,42 = 258,45 грн.

Згідно формули затрати на електроенергію де необхідна потужність множиться на кількість годин на розробку додатку і множиться на вартість кіловат-години електроенергії що в висновку дорівнює 258,45 грн.

5.5 Розрахунок суми амортизаційних відрахувань

Характерною особливістю застосування основних фондів у процесі розробки архітектури додатку є їх відновлення. Для відновлення засобів праці у натуральному виразі необхідне їх відшкодування у вартісній формі, яке здійснюється шляхом амортизації.

Амортизація – це процес перенесення вартості основних фондів на вартість новоствореної продукції з метою їхнього повного відновлення.

Для визначення амортизаційних використовується формула:

$$А = \frac{Б_В \cdot Н_А}{100} \quad (5.8)$$

де А – амортизаційні відрахування за звітний період, грн.; БВ – балансова вартість групи основних фондів на початок звітного періоду, грн.; НА – норма амортизації.

Комп’ютери та оргтехніка належать до четвертої групи основних фондів. Для цієї групи річна норма амортизації дорівнює 60 % (квартальна – 15 %).

Для розробки архітектури на основі об’єктного підходу засобом розробки є комп’ютер. Його сума становить 35000 грн. Отже, амортизаційні відрахування будуть рівні:

$$A = 35000 \cdot 5\% / 100\% = 1750 \text{ грн.}$$

Згідно формули для визначення амортизаційних де БВ множиться на І ділиться на 100% амортизація розробки становить 1750 грн.

5.6 Обчислення накладних витрат

Накладні витрати пов'язані з обслуговуванням апаратури та створенням необхідних умов праці.

В залежності від організаційно-правової форми діяльності господарюючого суб'єкта, накладні витрати можуть становити 20–60 % від суми основної та додаткової заробітної плати працівників.

$$Нв = Во.п. \cdot 0,2…0,6 \quad (5.9)$$

де Нв – накладні витрати.

Отже, накладні витрати:

$$Нв = 19342 \cdot 0,2 = 3868 \text{ грн.}$$

Накладні витрати згідно розрахунку формули на підтримку апаратури для розробки архітектури графчного рушія на основі об'єктног підходу, становить 3868 грн.

5.7 Складання кошторису витрат та визначення собівартості науково-дослідницької роботи

Результати проведених вище розрахунків зведемо у таблицю 5.4.

Таблиця 5.4 – Кошторис витрат на НДР

Зміст витрат	Сума, грн.	В % до загальної суми
Витрати на оплату праці	16819	61,35
Відрахування на соціальні заходи	4545	16,58

Матеріальні витрати	150	0,55
Витрати на електроенергію	258,45	0,94
Амортизаційні відрахування	1750	6,38
Накладні витрати	3868	14.11
Собівартість	27417	100.00

Собівартість (Св) розробки архітектури додатку розрахуємо за формулою:

$$Св = Во.п. + Вс.з. + Зм.в. + Зв + А + Нв \quad (5.10)$$

Отже, собівартість розробки архітектури додатку дорівнює:

Св = 16819 + 4545 + 150 + 258,45 + 1750 + 3868 = 27417 грн.

Загальний кошторис витрат та визначення собівартості науково-дослідницької роботи становить 27417 грн.

5.8 Розрахунок ціни програмного продукту

Ціну розробки архітектури додатку можна визначити за формулою:

$$Ц = \frac{C_B \cdot (1 + Р_{рен}) + К \cdot В_{н.і.}}{К} \cdot (1 + ПДВ) \quad (5.11)$$

де Ррен. – рівень рентабельності, 30 %; К – кількість замовлень, од. (встановлюється лише при розробці програмного продукту та мікропроцесорних систем); Вн.і. – вартість носія інформації, грн. (встановлюється лише при розробці програмного продукту); ПДВ – ставка податку на додану вартість, (20 %).

Оскільки розробка є прикладною, і використовуватиметься тільки для одного підприємства, то для розрахунку ціни не потрібно вказувати коефіцієнти К та Ві.н , оскільки їх в даному випадку не потрібно.

Тоді, формула для обчислення ціни розробки буде мати вигляд:

$$Ц = С_В \cdot (1 + Р_{рен}) \cdot (1 + ПДВ) \quad (5.12)$$

Звідси ціна на роботу складе:

$$Ц = 27417 \cdot (1 + 0,3) \cdot (1 + 0,2) = 42771 \text{ грн.}$$

Загальний розрахунок ціни архітектури додатку становить 42771 грн.

5.9 Визначення економічної ефективності і терміну окупності капітальних вкладень

Ефективність виробництва – це узагальнене і повне відображення кінцевих результатів використання робочої сили, засобів та предметів праці на підприємстві за певний проміжок часу.

Економічна ефективність (Ер) полягає у відношенні результату виробництва до затрачених ресурсів:

$$Е_р = \frac{П}{С_В} \quad (5.13)$$

де П – прибуток; СВ – собівартість.

Плановий прибуток (Ппл) знаходимо за формулою:

$$Ппл = Ц – Св . \quad (5.14)$$

Розраховуємо плановий прибуток:

$$Ппл = 42771 – 27417 = 15354 \text{ грн.}$$

Отже, формула для визначення економічної ефективності набуде вигляду:

$$E_р = \frac{П_{пл}}{Св} \quad (5.15)$$

Тоді, Ер = 15354 / 27417 = 0,56.

Поряд із економічною ефективністю розраховують термін окупності капітальних вкладень (Тр):

$$T_р = \frac{1}{E_р} \quad (5.16)$$

Термін окупності дорівнює:

Тр = 1 / 0,56 = 1,8 р.

Згідно формул плановий прибуток від розробки архітектури додатку становить 15354 грн., економічна ефективність дорівнює 0,56 а термін окупності становить 1,8 роки що вважається доцільним та економічно вигідним.

5.10 Висновок до п’ятого розділу

В обґрунтуванні економічної ефективності магістерської роботи освітнього рівня «магістр» було розраховано основні техніко-економічні показники розробки архітектури на основі об’єктного підходу (див. таблиця 5.5).

Орієнтоване значення економічної ефективності становить 0,56 що є достатньо високим значенням.

Період окупності повинен варіюватися від 1 до 3 років, тоді розвиток вважається доцільним та економічно вигідним. Термін окупності архітектури додатку становить 1,8 років.

Таблиця 5.5 – Техніко-економічні показники додатку

№	Показник	Значення
1	Собівартість, грн.	27417
2	Плановий прибуток, грн.	15354
3	Ціна, грн.	42771
4	Економічна ефективність	0,56
5	Термін окупності, рік	1,8

Отже, розробка архітектури додатку може бути реалізована та розвинена, оскільки вона є економічно вигідною для всіх основних технічних та економічних показників.

6 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

6.1 Охорона праці

6.1.1 Види страхових виплат Фонду соціального страхування від нещасних випадків на виробництві та професійних захворювань, на які може розраховувати працівник у разі його травмування, профзахворювання або смерті

Фонд соціального страхування України [41] — державний цільовий фонд, який здійснює керівництво та управління загальнообов'язковим державним соціальним страхуванням в Україні від нещасного випадку, у зв'язку з тимчасовою втратою працездатності та медичним страхуванням, провадить акумуляцію страхових внесків, контроль за використанням коштів, забезпечує фінансування виплат за цими видами загальнообов'язкового державного соціального страхування та здійснює інші функції згідно із затвердженим статутом.

У разі настання страхового випадку Фонд здійснює наступні виплати [42]:

- Виплата допомоги по тимчасовій непрацездатності. Допомога по тимчасовій непрацездатності, пов’язана з нещасним випадком на виробництві або професійним захворюванням, призначається та виплачується страхувальником за місцем роботи потерпілого, де стався страховий випадок, у розмірі 100 відсотків середньої заробітної плати (оподаткованого доходу). Виплата допомоги по тимчасовій непрацездатності за перші п’ять днів тимчасової непрацездатності потерпілого проводиться за рахунок коштів страхувальника, а починаючи з шостого дня непрацездатності - за рахунок коштів Фонду.
- Призначення потерпілому одноразової допомоги та щомісячної страхової виплати. Право на отримання потерпілим одноразової допомоги та

щомісячної страхової виплати настає з дня встановлення медико-соціальною експертною комісією (далі – МСЕК) стійкої втрати професійної працездатності. Право на отримання потерпілим одноразової допомоги та щомісячної страхової виплати настає з дня встановлення МСЕК стійкої втрати професійної працездатності. Якщо справи про страхові виплати розглядаються вперше по закінченню трьох років з дня встановлення МСЕК стійкої втрати потерпілим професійної працездатності внаслідок ушкодження здоров'я, то страхові виплати призначаються з дня звернення за їх призначенням відповідно до чинних нормативно–правових актів.

- Призначення одноразової допомоги. Щомісячна страхова виплата встановлюється відповідно до ступеня втрати професійної працездатності та середньомісячної заробітної плати потерпілого перед настанням страхового випадку. При цьому максимальний розмір щомісячної страхової виплати не повинен перевищувати 10 розмірів прожиткового мінімуму, встановленого для працездатних осіб на дату настання права на страхову виплату. Мінімальний розмір призначеної щомісячної страхової виплати потерпілому у перерахунку на 100 відсотків втрати професійної працездатності не може бути меншим за прожитковий мінімум, встановлений для працездатних осіб, на дату настання права на страхову виплату.

- Перерахування розміру щомісячної страхової виплати у разі зміни ступеня втрати професійної працездатності. Перерахування розміру щомісячної страхової виплати у разі зміни ступеня втрати професійної працездатності проводиться, виходячи з відкоригованої заробітної плати, визначеної за останнім її коригуванням при проведенні перерахування відповідно до частини другої статті 37 Закону № 1105. Максимальний розмір щомісячної страхової виплати після проведеного перерахування не повинен перевищувати 10 розмірів прожиткового мінімуму, встановленого для працездатних осіб на дату настання права на проведення перерахунку. Перерахований розмір щомісячної страхової виплати встановлюється з дня зміни відсотка втрати професійної працездатності.

- Призначення страхової виплати потерпілому при тимчасовому переведенні на легшу, нижчеоплачувану роботу. За потерпілим, тимчасово переведеним на легшу, нижчеоплачувану роботу, зберігається його середньомісячний заробіток на строк, визначений лікарсько-консультаційною комісією (далі - ЛКК), або до встановлення стійкої втрати професійної працездатності. Починаючи з п’ятнадцятого дня переведення потерпілого на легшу роботу, зазначені виплати (доплата до середнього заробітку, який він мав до ушкодження здоров’я) проводяться страхувальником за рахунок коштів Фонду на строк, установлений ЛКК або МСЕК.
- Право на страхові виплати в разі смерті потерпілого. У разі смерті потерпілого право на одержання щомісячних страхових виплат мають непрацездатні особи, які перебували на утриманні померлого або мали на день його смерті право на одержання від нього утримання, а також дитина померлого, яка народилася протягом не більш як десятимісячного строку після його смерті.

6.1.2 Ефективність охорони праці у Великобританії у сфері ІТ

Питання охорони праці та здоров’я працівників у сфері ІТ, як і у інших сферах, є невід’ємною частиною стратегії Великобританії. Працівники інформаційно-технічних відділів не мають державних преференцій з точки зору охорони праці, тому працівники даної галузі користуються такими ж правами і льготами, як і працівники інших галузей [39].

У Великобританії була прийнята Стратегія безпеки здоров’я на робочих місцях, яка поставила завдання зниження рівня смертності і важких травм на виробництві на 10%, рівня професійної захворюваності – на 20%, втрат робочого часу внаслідок захворюваності – на 30%. У Стратегії розроблені нові шляхи досягнення безпечної праці в мінливій економіці, тобто в економіці, де відбувається зниження кваліфікації працівників з питань охорони праці у зв’язку зі збільшенням числа дрібних підприємств (до 10

чол.), зростанням кількості працівників, зайнятих неповний робочий час; збільшенням числа працюючих жінок; з розширенням сфери послуг і залученням недостатньо навченої іноземної робочої сили [40].

Особливе місце приділено розвитку бюджетного сектора (який повинен стати зразком створення умов безпечної та здорової праці для приватного сектора), розвитку соціального партнерства з профспілками, діловими асоціаціями, з регіонами та місцевою владою. Стратегія ставить за мету створити суспільство, в якому ризики правильно оцінюються, усвідомлюються і управляються, в якому управління здоров'ям і безпекою є колективним завданням, у вирішенні якого значна роль відводиться і самому працівнику. Великобританія виділяє чотири стратегічні області:

- Розвиток тісного партнерства.
- Допомога людям в отриманні переваг від ефективного управління охороною й безпекою праці, розвиток культури безпечної праці й відповідальності за своє здоров'я.
- Фокусування на основній роботі комісії по охороні та безпеці праці , направленій на зниження травмувань та шкоди здоров'ю за робочим місцем.
- Розповсюдження стратегічного бачення (встановлення надійної двохсторонньої комунікації з основними зацікавленими у цьомусторонами).

Виконання Стратегії покладено на комісію з охорони та безпеки праці, підвідомчу Міністерству праці і пенсій Великобританії. Вона випустила велику кількість спеціальних матеріалів, у яких у наочній і зручній формі доводить до роботодавців та їх працівників основні вимоги закону про охорону здоров'я працівників та безпеки праці. Просто і зрозуміло ("п'ять кроків оцінки ризиків") у них розписана процедура оцінки ризиків (повторюється не рідше одного разу на рік), результати якої заносять в спеціальну реєстраційну карту. Системою управління професійними ризиками охоплено кожне робоче місце у Великобританії. Для малого бізнесу розроблені спрощені форми і методики оцінки ризиків, а також інформаційні

матеріали, що полегшують засвоєння правил техніки безпеки, обов’язків роботодавця і працівників з управління ризиками.

Цікавий також багатий досвід Великобританії в цілеспрямованому культивуванні серед бізнес-спільноти політики корпоративної соціальної відповідальності, через яку ефективно вирішуються багато питань охорони здоров'я працівників. Керівникам підприємств, членам рад директорів і управління присвячені спеціальні матеріали урядового офісу Об'єднаного Королівства, у яких в лаконічній формі викладаються їх обов'язки в сфері охорони праці та здоров'я персоналу і переваги конкурентоспроможних здорових умов праці для ведення успішного бізнесу [43].

6.2 Безпека в надзвичайних ситуаціях

6.2.1 Організація цивільного захисту на об’єктах промисловості та виконання заходів щодо запобігання виникненню надзвичайних ситуацій техногенного походження

Об’єкт господарської діяльності (підприємство, установа, організація) – основна ланка в системі ЦЗ держави. На об’єкті, де зосереджено людські і матеріальні ресурси, здійснюють економічні і захисні заходи []. Відповідно до законодавства, керівництво підприємств, установ і організацій незалежно від форм власності і підпорядкування забезпечує своїх працівників засобами індивідуального та колективного захисту, місцем у захисних спорудах, організовує евакозаходи, створює сили для ліквідації наслідків НС та забезпечує їх готовність, виконує інші заходи з ЦЗ і несе пов’язані з цим матеріальні та фінансові витрати. Власники потенційно небезпечних об’єктів відповідають також за оповіщення і захист населення, що проживає в зонах можливого ураження від наслідків аварій на цих об’єктах. Начальником ЦЗ об’єкта є керівник об’єкта. він відповідає за організацію і стан ЦЗ об’єкта, керує діями органів і сил цз під час проведення рятувальних робіт на ньому.

Заступники начальника ЦЗ об’єкта допомагають йому з питань евакуації, матеріально-технічного постачання, інженерно-технічного забезпечення тощо (рис. 3.15.1). Органом повсякденного управління ЦЗ є відділ (сектор) з питань НС та ЦЗ, який організовує і забезпечує повсякденне керівництво виконанням завдань ЦЗ на об’єкті. Для підготовки та втілення в життя заходів з окремих напрямів створюють служби зв’язку та оповіщення, сховищ і укриттів, протипожежної охорони, охорони громадського порядку, медичної допомоги, протирадіаційного і протихімічного захисту, аварійно-технічного та матеріально-технічного забезпечення тощо. Начальниками служб призначають начальників установ, відділів, лабораторій, на базі яких вони утворюються. Службу зв’язку та оповіщення створюють на базі вузла зв’язку об’єкта. Головне завдання служби – забезпечити своєчасне оповіщення керівного складу та службовців про загрозу аварії, катастрофи, стихійного лиха, нападу противника; організувати зв’язок і підтримувати його в стані постійної готовності. Протипожежну службу створюють на базі підрозділів відомчої пожежної охорони. Служба розробляє протипожежні профілактичні заходи і контролює їх виконання; організовує локалізацію і гасіння пожежі. На рисунку 6.1 зображено схему організації цивільного захисту на підприємстві.

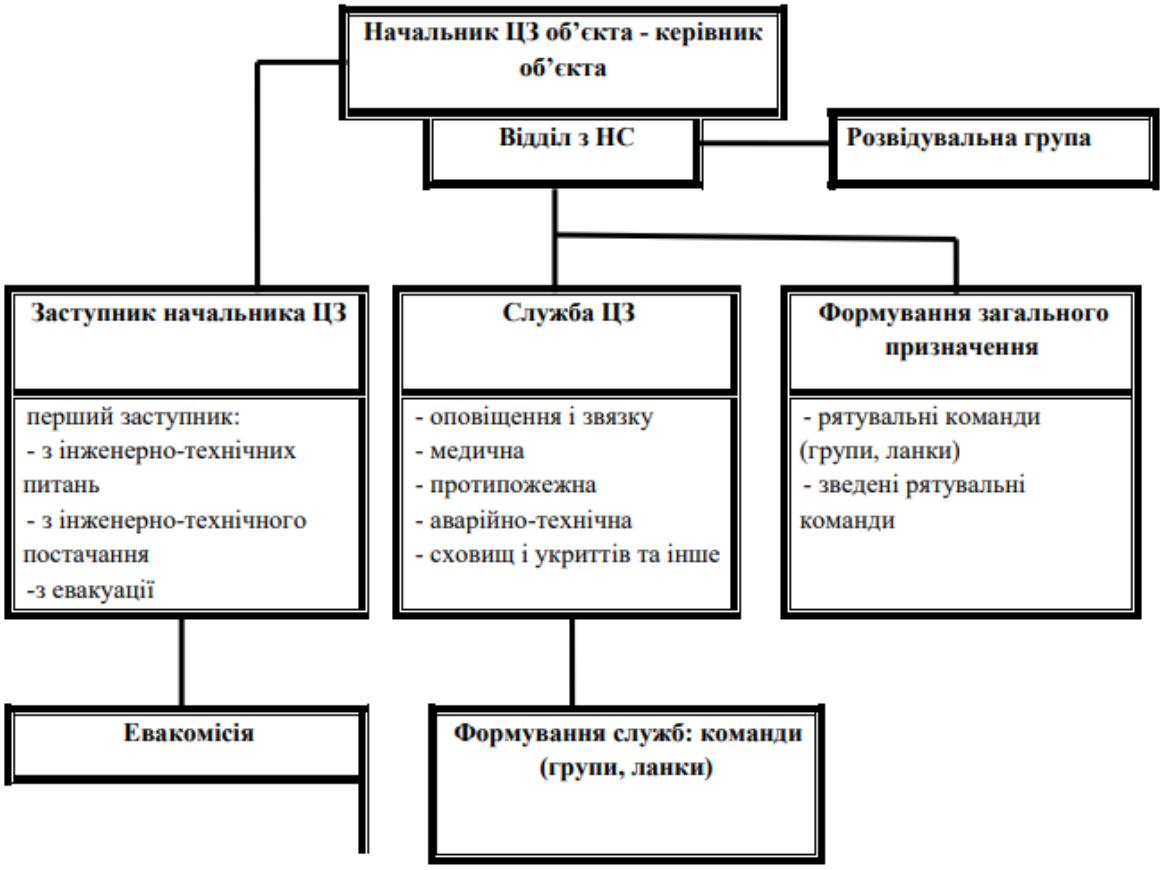


Рисунок 6.1 – Схема організації цивільного захисту на підприємстві

В основу інженерно-технічних заходів щодо запобігання надзвичайним ситуація і зменшення можливих втрат і збитків від них повинні бути покладені конкретні превентивні заходи, які здійснюються за видами природних і техногенних небезпек та загроз. Запобігання більшості небезпечних природних явищ пов'язане із значними фінансовими і матеріально-технічними затратами [45].

У техногенній сфері робота з попередження аварій повинна вестися на конкретних об'єктах і виробництвах. До них належать удосконалення технологічних процесів, підвищення надійності технологічного обладнання та експлуатаційної надійності систем, своєчасне оновлення виробничих фондів, застосування якісної конструкторської документації, високоякісної сировини, матеріалів, комплектуючих виробів, використання кваліфікованого персоналу, створення і використання ефективних систем

контролю та технічної діагностики, безаварійної зупинки виробництва, локалізація і ліквідація аварійних ситуацій.

Місцеві органи виконавчої влади в першу чергу повинні забезпечити створення і підтримання в постійній готовності систем централізованого оповіщення населення.

Одним із напрямків зниження масштабів надзвичайних ситуацій є будівництво і використання захисних споруд різного призначення: гідротехнічні захисні споруди (греблі, шлюзи, дамби, тощо), проведення берегоукріплюючих робіт від зсувів та обвалів.

До превентивних заходів відноситься інженерний захист населення і територій від негативного впливу повеневих вод. Важливим напрямком превентивних заходів, які сприяють зменшенню масштабів надзвичайних ситуацій є створення і використання систем оповіщення населення, персоналу об’єктів і органів управління, що дозволяє вжити своєчасних заходів щодо захисту населення та зменшення матеріальних збитків [46].

Конкретні заходи щодо запобігання надзвичайним ситуаціям місцевими органами виконавчої влади повинні здійснюватися під час підготовки об’єктів економіки і систем життєзабезпечення населення до роботи в екстремальних умовах.

З метою сталого функціонування економіки і виживання населення в умовах надзвичайних ситуацій повинні здійснюватися наступні заходи:

- обмеження (недопущення) нового будівництва об’єктів і розширення існуючих в районах підвищеної небезпеки природних стихійних явищ, у водоохоронних зонах забороняється розміщення полігонів для твердих побутових промислових відходів, складів нафтопродуктів і мінеральних добрив, а також житлових будинків і баз відпочинку;
- поступове виведення з міст, підприємств, баз, складів, які переробляють або використовують значну кількість небезпечних речовин;

- розвиток у позаміській зоні об’єктів матеріальних резервів з урахуванням потреб для забезпечення населення у надзвичайних ситуаціях.

6.2.2 Особливості роботи та розлади здоров’я користувачів комп’ютерів, що формується під впливом роботи за ком’ютером

У професійних операторів частіше зустрічаються порушення органів зору, опорно-рухового апарату, центральної нервової, серцево-судинної, імунної та статевої систем, захворювання шкіри. Зафіксована значна кількість скарг операторсь- кого персоналу на загальне недомагання, передчасне стомлювання, головний біль, порушення функцій органів зору, які здійснювали несприятливий психофізіологічний вплив на самопочуття та працездатність операторів [47]. Сучасна професія користувача ВДТ належить до розумової праці, яка характеризується: високою напруженістю зорових функцій; одноманітною позою; великою кількістю стереотипних висококоординованих рухів, що виконуються лише м'язами кистей рук на фоні малої загальної рухової активності; значним нервовоемоційним компонентом, особливо в умовах дефіциту часу; роботою з великими масивами інформації, що викликає активізацію уваги та інших вищих психічних функцій. Крім того, при роботі з дисплеями на електронно-променевих трубках виникає вплив на користувача цілої низки факторів фізичної природи — електростатичні поля, радіочастотне та рентгенівське випромінювання тощо.

Діяльність професіоналів можна поділити на три групи:

- Діяльність, яка пов'язана з виконанням нескладних багаторазово повторюваних операцій, що не вимагають великого розумового напруження. Наприклад, робота операторів комп'ютерного набору, працівн

- Діяльність, яка пов'язана із здійсненням логічних операцій, що постійно повторюються. Це робота інженера-економіста, інженера-проектувальника, оператора автоматизованого виробництва.
- Діяльність, коли в процесі роботи необхідно приймати рішення за відсутності заздалегідь відомого алгоритму. Наприклад, робота інженера-програміста, диспетчерів руху залізничного транспорту, аеропортів тощо.

У користувачів, які інтенсивно використовують комп'ютер в умовах значних розумових напружень досить часто (40—70%) виникають психологічні та поведінкові порушення (нервозність, роздратування, тривога, нерішучість, замкнутість тощо) [48]. Серед користувачів ВДТ в США і Європі значного поширення набуло специфічне захворювання, яке отримало назву синдром комп'ютерного стресу (СКС). СКС супроводжується головним болем, запаленням очей, алергією, роздратованістю, млявістю і депресією. Інформаційне перевантаження користувачів ВДТ супроводжується низкою специфічних захворювань, які називають інформаційними. Першим симптомом їх є головний біль. Дослідження, проведені в США, Німеччині, Швейцарії та інших країнах, показали, що робота з обслуговування ВДТ супроводжується підвищеним напруженням зору, інтенсивністю і монотонністю праці, збільшенням статичних навантажень, нервово-психічним напруженням, впливом різного виду випромінювань та ін. Внаслідок цього серед операторів ВДТ, як зазначають фахівці Всесвітньої організації охорони здоров'я, частіше, ніж в інших групах працюючих, трапляються такі професійні захворювання, як передчасна стомлюваність, погіршення зору, м'язові і головні болі, психічні й нервові розлади, хвороби серцево-судинної системи, онкологічні захворювання та ін. Вважається, що стан організму операторів ВДТ визначається комплексним впливом факторів трудового процесу і середовища, значення яких є неоднаковим. На операторів з малим стажем роботи на ВДТ домінуючий вплив чинять фактори середовища, а на операторів зі стажем понад 5 років - фактори трудового процесу .

Розлади здоров'я користувачів, що формуються під впливом роботи за комп'ютером [49]:

- Комп’ютерний зоровий синдром (КЗС) – комплекс порушень здоров’я, який може виникати у користувачів персональних комп’ютерів (ПК). Діагноз ставлять, якщо людина, що працює за ПК протягом двох годин, висловлює хоча б дві з десяти скарг: головний біль, сльозотеча, різь, туман, двоїння, свербіж, важкість в очах, фотофобія, миготіння знаків на екрані, нудота. У користувачів ПК дуже поширені кон’юнктивіти і блефарити, патогенетично пов’язані з КЗС. Синдром розвивається при умові, що робоче місце організовано неправильно – у користувача незручне крісло, відсутні пюпітри для паперів, підставки для ніг та кистей рук, не встановлена висота і нахил монітора відносно очей, відстань від очей до екрана. За таких умов тіло людини при роботі займає вимушене положення: спина статично напружена, шия витягнута, плечі жорстко фіксовані. Напружені м’язи погіршують кровотік у сонних артеріях, а недостатнє кровозабезпечення головного мозку веде до очманіння, появи головного болю. На фоні шийного остеохондрозу з’являється відчуття випирання очних яблук, туману в очах, мушок та райдужних кіл у полі зору. Розвитку КЗС сприяє поганий мікроклімат приміщення, значна загальна іонізація та мікробне забруднення, а також куріння.
- Перенапруження скелетно-м'язової системи. Діяльність користувачів комп'ютерів характеризується тривалою багатогодинною (8 год. і більше) працею в одноманітному напруженому сидячому положенні, малою руховою активністю при значних локальних динамічних навантаженнях, що припадають лише на кисті рук. Такий характер роботи може призвести до появи низки хворобливих симптомів, що об'єднані загальною назвою — синдром довготривалих статичних навантажень (СДСН). Узагальнюючи статистичні дані можна зробити висновок про те, що СДСН може проявлятись втомою, скутістю, болем, судомою, онімінням та ін., локалізуватись у різних

частинах тіла (шия, спина, руки, ноги та ін.) і виникати індивідуально з різною частотою (ніколи, рідко, епізодично, щоденно). Робоче положення "сидячи" забезпечується статичною працею значної кількості м'язів, що дуже втомлює. При такому положенні тіла м'язи ніг, плечей, шиї та рук довгий час перебувають у скороченому стані. Оскільки м'язи не розслабляються, в них погіршується кровообіг.

- Ураження шкіри [50]. Частота шкірних уражень корелюється з низькою відносною вологістю на робочих місцях операторів та частим виникненням електростатичних зарядів. Електростатичне поле, яке генерується дисплеєм комп'ютера, посилює електростатичний заряд на тілі оператора, а відтак зростає електростатичне поле біля нього. Підвищення відносної вологості повітря у приміщенні в поєднанні з вилученням килимових покриттів, в яких нагромаджуються статичні заряди, сприяли зниженню шкірних висипань на обличчі. Обладнання заземлення, встановлення сіткового екрана з металевого дроту між дисплеєм і оператором у деяких випадках знижувало частоту захворювань шкіри.

- Розлади центральної нервової системи (ЦНС). Виробнича діяльність операторів ВДТ має свої особливості, під впливом яких можуть формуватись розлади здоров'я. До найважливіших факторів, характерних для роботи операторів ВДТ, що впливають на погіршення стану їх ЦНС належать: інформаційне перевантаження мозку в поєднанні з дефіцитом часу, тривожне очікування інформації, особливо тієї, що викликає необхідність прийняти рішення; велике зорове та нервово-емоційне напруження; гіподинамія; монотомія; висока відповідальність за кінцевий результат. Під впливом цих факторів виникають зміни у співвідношенні процесів збудження та гальмування в корі головного мозку. При цьому функціональна активність ЦНС знижується, а порушення рівноваги основних нервових процесів все більше спрямовано в бік гальмування. В організмі розвивається втома.

6.3 Висновок до шостого розділу

У шостому розділі були розглянуті види страхових виплат Фонду соціального страхування від нещасних випадків на виробництві та професійних захворювань, на які може розраховувати працівник у разі його травмування, профзахворювання або смерті, ефективність охорони праці у Великобританії у сфері ІТ.

Також було проведено огляд організації цивільного захисту на об’єктах промисловості та виконання заходів щодо запобігання виникненню надзвичайних ситуацій техногенного походження та особливості роботи та розлади здоров’я користувачів комп’ютерів, що формується під впливом роботи за ком’ютером.

7 ЕКОЛОГІЯ

7.1 Енергозбереження і його роль у вирішенні екологічних проблем

Енергозбереження - комплекс заходів по реалізації правових, організаційних, наукових, виробничих, технічних і економічних заходів, спрямованих на ефективне (раціональне) використання (і економне витрачання) паливно-енергетичних ресурсів та на залучення в господарський оборот поновлюваних джерел енергії.

В даний час енергозберігаючі технології є одним з ключових напрямків розвитку енергетичної політики України. Так як економіка країни характеризується високою енергоємністю, необхідними заходами щодо забезпечення економії енергії являються:

- Ліквідація технологічної відсталості промисловості;
- Оснащення підприємств новим енергозберігаючим обладнанням;
- Впровадження енергозберігаючих технологій;
- Заохочення інвестицій в енергозберігання;
- Робота з населенням

Ще одним напрямком, покликаним в майбутньому замінити традиційні види палива, є перехід на енергозберігаючі технології в рамках використання поновлюваних джерел енергії, до яких належать: тверда біомаса і тваринні продукти, промислові відходи, гідроенергія, геотермальна енергія, сонячна енергія, енергія вітру, енергія припливів морських хвиль і океану. Це дає не тільки значне зменшення витрат на енергетичні витрати, а й має великі екологічні плюси.

На сучасному етапі можна виділити три основні напрямки енергозбереження:

- Корисне використання (утилізація) енерговитрат;
- Модернізація обладнання з цілю зменшення витрат енергії;
- Інтенсивне енергозбереження

Основною метою енергозбереження є підвищення енергоефективності всіх галузей, у всіх населених путкнах, а також у країні вцілому.

Із розвитком комп'ютерних технологій, методики і заходи щодо розвитку енергозберігання та енергоефективності також розвиваються. Можна уявити, що людина могла б приймати рішення щодо споживання енергії в реальному часі на підставі миттєвого надходження інформації про електричне навантаження мережі. Припустити, що можна не тільки економити, але й збільшувати чистий прибуток компанії за допомогою оптимального споживання енергії та її економії. І все, що потрібно для реалізації цих речей – звичайний смартфон.

Кілька років тому про такі сценарії можна було тільки мріяти. Сьогодні вони стають все більш реалістичними, в міру того як домовласники і землевласники всюди беруть під контроль споживання енергії до такої міри, яка ніколи раніше не була можлива. Електроенергія стає все більш «розумною» на кожному етапі, починаючи з централізованої або локальної видобутку і закінчуючи її кінцевим споживанням електричними пристроями, якими ми користуємося у себе вдома, на роботі і в дорозі, в більш «розумних» будинках і будівлях. Ці успіхи в моніторингу, в мережевій взаємодії і управлінні (так звана «розумна» енергія) стали можливі завдяки прогресивній напівпровідниковій технології.

Вироблення і передача електроенергії піддаються повільній, але безперервній еволюції з переходом до більш розумної і виборчої подачі енергії туди і тоді, де і коли вона потрібна. У промисловому масштабі провайдери послуг вкладаються в такі альтернативні джерела енергії, як сонячні і вітряні ферми, а також працюють над більш ефективним використанням центральних електростанцій. Силове обладнання також стає більш розумним і все більше

об'єднується в мережі передачі даних для мінімізації втрат при передачі, підключення і перетворенні електроенергії.

Використання розумної енергії надає все більш сприятливий вплив на домашню обстановку і офісні будівлі. Датчики віддають команди світильникам знизити яскравість або включитися і вимкнутися, а також команди зменшити або збільшити вентиляцію, в залежності від присутності людей в приміщенні. Енергетичні мережі можуть включати в себе такі елементи, як програмне забезпечення, яке аналізує споживання енергії по окремих зонах, пристроїв і застосункам. Такі програми допомагають визначити, чи потрібно замінити обладнання тому, що його неефективна робота робить його дорогим, або тому, що воно є критично важливим і не повинно виходити зі строю. Енергетична мережа також необхідна, якщо будівля генерує електрику шляхом збору сонячної або вітрової енергії.

Технологія «розумної» енергії постійно вдосконалюється і поповнюється новими і прогресивними методами. Крім відповідності вимогам компактності, надзвичайно низького енергоспоживання і мінливих рівнів продуктивності, пропоновані рішення від виробників мікросхем повинні бути здатні підтримувати гнучкі конфігурації, особливо в тих випадках обміну даними, коли використовуються різноманітні протоколи. Такі функції обробки аналогових сигналів, як перетворювачі даних, також є ключовими елементами, оскільки поставляють дані для функцій управління, регулювання та контролю. Розробникам системи потрібно розглянути переваги від отримання якомога більшої кількості таких компонентів від одного постачальника інформаційної системи, щоб спростити процеси проектування та постачання і забезпечити сумісність різних складних функцій, які виконуються кожним компонентом. Інші можливості, такі як інтеграція, можуть допомогти підтримувати довгострокові технологічні маршрути для наступних поколінь продукції, а наявність різноманітних варіантів корпусів забезпечує гнучкість з точки зору вимог по установці на різні типи плат.

7.2 Методологія моделювання екологічних проблем

Моделювання та прогнозування стану довкілля являє собою систему понять і методів, націлених на відтворення, аналіз та прогноз розвитку різноманітних природних та техногенних екологічних систем на різних рівнях їх ієрархічної організації – від окремої екосистеми до національних і глобальних екосистем планети Земля. Кількісно обґрунтовуючи методи підвищення екологічної безпеки акваторій і територій та мінімізуючи екологічні ризики, моделювання та прогнозування стану довкілля дозволяє розробляти стратегії підвищення якості навколишнього середовища з урахуванням інтересів теперішніх та майбутніх поколінь, підтримуючи біорізноманітність та багатство природних ресурсів.

Методи моделювання та прогнозування екологічної проблеми включають постановку задачі згідно з обраною метою, ідентифікацію досліджуваних структур, вибір оптимального методу моделювання, побудову моделі і доведення її адекватності досліджуваному процесу, варіантну реалізацію моделювання, прогноз розвитку подій та контроль за здобутими результатами.

На сьогоднішній день широко використовуються наступні методи моделювання екологічних проблем:

- Натурно-експериментальне моделювання;
- Математичне моделювання;
- Системне моделювання

При побудові моделей екологічних проблем застосовують наступні основні принципи:

- Принцип системності. Внаслідок пересиченості екосистем зв'язками екологічні об'єкти являють собою єдину систему. З цієї причини в екології виявилося необхідним злиття методів системного аналізу і

математичного моделювання. Це призвело до створення інтегрального методу системного моделювання - вищого етапу в розвитку екологічного моделювання. Принцип системності полягає в усвідомленні цілісності об'єктів світу, їхньої стійкості і взаємозв’язку зі зовнішнім світом тощо; інший аспект цього принципу — динамічна багатогранність, єдність якості її кількості, теорії та практики.

- Принцип єдності структурності та ієрархічності. Фундаментальна риса екосистем — наявність у них складних ієрархічних структур. Звідси випливає вимога єдності структурності й ієрархічності системних екологічних моделей Відповідно виникає проблема структурування моделі, тобто виділення істотних підсистем і елементів із сукупності всіх зв'язків і компоненті. Звичайно систему організують найбільш залежні одне від одного елементи (підсистеми). Інші впливають на поводження системи слабко, а через їхню велику кількість — не узгоджено; отже їх можна розглядати як інтегровані зовнішні чи внутрішні фактори впливу

- Принцип багатомодельного опису. Через динамізм і складність екологічних об'єктів, що виникають у результаті множинності мети антропогенного втручання, на сьогодні немає можливості побудови єдиної теорії сонціоекосистеми в класичному розумінні, тобто як дедуктивної моделі, з якої можна вивести всі можливі наслідки. Тому наука йде по шляху створення множинних взаємодоповнюючих моделей.

- Принцип єдності формалізованою і неформалізованого опису. Досвід перших глобальних моделей розвитку світової соціоекосистеми, побудованих за замовленням Римського клуба, показав: єдиного формалізованого (математичного) опису недостатньо для адекватного моделювання соціоекосистеми. Для цього необхідно враховувати неформальні факторії і доповнювати формалізований опис (з позицій Історичного, психологічного та ні. підходів) неформалізованим описом.

- Принцип визнання фундаментальності екологічних процесів. Екологічні пронеси неможливо звести до простої сукупності біологічних,

фізичних, економічних процесів, оскільки всі вони тісно переплетені між собою. У цьому переплетенні виникають нові, екологічні закономірності. Звідси випливає самостійна значимість екологічних цінностей.

- Принцип єдності теорії та практики. Благополуччя соціоекосистеми, частиною якої є людина, має для неї найважливіше значення. Тому екологія є не тільки фундаментальною, але і прикладною наукою, що поєднує пізнання екологічних закономірностей із практичним їхнім застосуванням у повсякденній діяльності людини. Ця єдність виражається у вигляді принципу: "Не тільки дивися і думай — роби"

За допомогою моделювання одержують можливість оцінювання потенційних наслідків застосування різних стратегій оперативного керування впливу на екосистему, користування природними ресурсами (біотичними й абіотичними), оптимізації екосистем. Моделювання дозволяє глибоко проникнути в сутність явищ, зрозуміти їхню справжню природу.

7.3 Висновок до сьомого розділу

У сьомому розділі було розглянуто питання енергозбереження та його роль у вирішенні екологічних проблем. Також було розглянуто методології моделювання екологічних проболем.

ВИСНОВКИ

Машинне навчання, як і сам штучний інтелект, все частіше зустрічається у повсякденному житті людини. Саме тому ця галузь розробки програмного забезпечення є доволі трендовою, цікавою і являє собою широке поле для майбутніх досліджень. Розвиток штучного інтелекту формуватиме майбутні тенденції у розробці програмного забезпечення, і від нього залежитиме також розвиток суміжних галузей.

В даній роботі було описано тенденції та історія розвитку штучного інтелекту та машинного навчання.

Розглянуто основні популярні алгоритми машинного навчання, основні принципи та проблеми машинного навчання. Дана дипломна робота була націлена на проблематику браузерного машинного навчання – одного з напрямків використання машинного навчання. Було досліджено основні принципи розробки з використанням бібліотеки TensorFlow.js. Було розглянуто основні переваги даної бібліотеки перед її аналогами, було покроково описано етапи створення застосунку для оцінки вартості житла на основі послідовної моделі машинного навчання та проаналізовано вихідні дані.

Результатом даної роботи є сама прогнозована модель, яка базується на великій кількості вхідних даних, що, у свою чергу, доводить ефективність самого розвитку браузерного машинного навчання, та подальшу його популяризацію в майбутніх тенденціях розвитку інформаційних технологій.

Загалом, ідея розробки веб-застосунків з використанням машинного навчання у браузері є доволі переспективною завдяки своїй відносно низькій енергозатратності, вседоступності, та захищенності даних.

Програми браузерного машинного навчання пишуться на популярних мовах програмування JavaScript або Typescript, що дозволяє швидко обробляти інформацію, повністю використовувати потенціал браузера, та розширень, які ним підтримуються.

Прогнозування вартості житла – лише одна з багатьох можливих варіацій завдань, які можуть бути виконані за домогою машинного навчання. Саме за цим напрямком лежить майбутнє, коли багато факторів та різні набори вхідних даних можуть бути проаналізовані без участі людини.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Duchi J. Adaptive subgradient methods for online learning and stochastic optimization / J. Duchi, E. Hazan, Y. Singer // Journal of Machine Learning Research. — 2011. — P. 2121–2159.
2. Kingma D. Adam: A method for stochastic optimization / D. Kingma, J. Ba // arXiv preprint arXiv. — Vol. 1412, N 6980. — 2014. — P. 1–15.
3. Liu P. SVM or deep learning? A comparative study on remote sensing image classification / P. Liu, K.K.R. Choo, L. Wang, F. Huang // Soft Computing. — Vol. 21, N 23. — 2017. — P. 7053–7065.
4. Pirotti F. Benchmark of machine learning methods for classification of a Sentinel-2 image / F. Pirotti, F. Sunar, M. Piragnolo // International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences. — Vol. 41. — 2016. — P. 335–340.
5. Саттон Р.С Обучение с подкреплением / Саттон Р.С, Э. Г. Барто // БИНОМ, Лаборатория знаний, 2014 – С. 42-96.
6. Mnih, V. Playing Atari with deep reinforcement learning. Technical Report / Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller M. – DeepMind Technologies, 2013 – С. 7
7. Schematic illustration of the convolutional neural network. – Режим доступу: http://www.nature.com/nature/journal/v518/n7540/fig_tab/nature14236_F1.ht ml.
8. Моделирование процессов обучения в нейронных сетях. – Режим доступу: http://old.exponenta.ru/soft/others/mvs/stud3/3.asp.
9. Онлайн журнал engadget (Google DeepMind AI wins final Go match for 4-1 series win). – Режим доступу: https://www.engadget.com/2016/03/14/thefinal-lee-sedol-vs-alphago-match-is-about-to-start/.

10. A. M. TURING I.—COMPUTING MACHINERY AND INTELLIGENCE/ A.M. Тюрінг // Mind. – 1950. – № 236. – С. 433-460.

11. Breiman L. Random forests / L. Breiman // Machine learning. — Vol. 45, N 1. — 2001. — P. 5–32.

12. Hinton G.E. A fast learning algorithm for deep belief nets / G.E. Hinton, S. Osindero, Y.W. Teh // Neural computation. — Vol. 18, N 7. — 2006. — 1527– 1554 p

13. Загальна лінійна модель [Електронний ресурс] URL: https:// uk.wikipedia.org/wiki/Загальна лінійна модель.

14. Aggarwal C.C., Charu C. Data Classification Algorithms and Applications. 2015: Chapman & Hall /CRC.

15. Manevitz L. M. Y.M. Document Classification on Neural Networks Using Only Positive Examples // SIGIR. 2000.

16. Hochreiter S. Untersuchungen zu dynamischen neuronalen Netzen / S. Hochreiter // Diss. diploma thesis, institut für informatik, lehrstuhl prof. brauer, technische universität münchen. — 1991.

17. Sutskever I. Sequence to sequence learning with neural networks / I. Sutskever, O. Vinyals, Q.V. Le // In Advances in neural information processing systems. — 2014. — P. 3104–3112.

18. CS234: Reinforcement Learning – Режим доступу: http://web.stanford.edu/class/cs234/index.html.

19. Barnett V., Lewis T. Outliers in Statistical Data. Wiley, 1994

20. Dunnung T., Friedman E. Practical Machine Learning: A New Look at Anomaly Detection. O'Reilly Media, 2004

21. Beckman R., Cook R. Outliers // Technometrics, No. 25(2), 1983. pp. 119– 149

22. Khoshnava,S.M.; Rostami,R.;Valipour,A.; Ismail,M.; Rahmat,A.R. Rankofgreenbuildingmaterialcriteria based on the three pillars of sustainability using the hybrid multi criteria decision making method. J.Clean. Prod. 2018, 173, 82–99.

23. Chen Y. Deep learning-based classification of hyperspectral data / Y. Chen, Z. Lin, X. Zhao et al. // IEEE Journal of Selected topics in applied earth observations and remote sensing. — Vol. 7, N 6. — 2014. — P. 2094–2107.

24. Zhang F. Scene classification via a gradient boosting random convolutional network framework / F. Zhang, B. Du, L. Zhang // IEEE Transactions on Geoscience and Remote Sensing. — Vol. 54, N 3. — 2016. — P. 1793–1802.

25. Chen Y. Spectral–spatial classification of hyperspectral data based on deep belief network / Y. Chen, X. Zhao, X. Jia // IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. — Vol. 8, N 6. — 2015. — P. 2381–2392.

26. Choromanska A. The loss surfaces of multilayer networks / A. Choromanska, M. Henaff, M. Mathieu et al. // In Artificial Intelligence and Statistics. — 2015. — P. 192–204.

27. Lyu H. Learning a transferable change rule from a recurrent neural network for land cover change detection / H. Lyu, H. Lu, L. Mou // Remote Sensing. — Vol. 8, N 6. — 2016. — P. 1–22.

28. Peña-Barragán J.M. Object-based crop identification using multiple vegetation indices, textural features and crop phenology / J.M. Peña-Barragán, M.K. Ngugi, R.E. Plant, J. Six // Remote Sensing of Environment. — Vol. 115, N 6. — 2011. — P. 1301–1316.

29. Zhao W. Learning multiscale and deep representations for classifying remotely sensed imagery / W. Zhao, S. Du // ISPRS Journal of Photogrammetry and Remote Sensing. — Vol. 113. — 2016. — P. 155–165.

30. Zhang F. Saliency-guided unsupervised feature learning for scene classification / F. Zhang, B. Du, L. Zhang // IEEE Transactions on Geoscience and Remote Sensing. — Vol. 53, N 4. — 2015. — P. 2175–2184.

31. Hwang B., Cho S. Characteristics of auto-associative MLP as a novelty detector. // In Proceedings of the IEEE International Joint Conference on Neural Networks. Washington, DC. 10–16 July,1999. Vol. 5, pages 3086– 3091.

32. Офіційний сайт бібліотеки brain.js. – [Електронний ресурс] - Режим доступу https://brain.js.org/.

33. Офіційний сайт бібліотеки machinelearn.js. – [Електронний ресурс] - Режим доступу https://www.machinelearnjs.com/.

34. Офіційний сайт бібліотеки math.js. – [Електронний ресурс] - Режим доступу https://mathjs.org/.

35. Репозиторій бібліотеки face-api.js. – [Електронний ресурс] - Режим доступу https://github.com/justadudewhohacks/face-api.js/.

36. Офіційний сайт бібліотеки r.js. – [Електронний ресурс] - Режим доступу https://requirejs.org/.

37. Офіційний сайт бібліотеки stdlib-js. – [Електронний ресурс] - Режим доступу https://stdlib.io/.

38. Офіційний сайт бібліотеки tensorflow.js. – [Електронний ресурс] - Режим доступу https://www.tensorflow.org/js.

39. Новиков, Ю. В. Охрана окружающей среды [Текст] : учеб. пособие для техникумов / Ю.В. Новиков. - М. : Высшая школа, 1987. - 287 с.

40. Основи екології та охорона навколишнього природного середовища [Текст] : навч. посіб. для студ. вищих навч. закладів / [Бедрій Я. І.; Джигирей В. С.; Кидисюк, А. І. та ін.]; за ред. В. С. Джигирей ; Український держ. лісотехнічний ун-т, Львівський електротехнікум зв'язку. - Л. : [б.в.], 1999. - 239 с. Альтернативна назва : Екологія та охорона природи. - ISBN 5-7763-2641-9.

41. Фонд соціального страхування України : [Електронний ресурс] // Вікіпедія – вільна енциклопедія. – Режим доступу: https://uk.wikipedia.org/wiki/Фонд_соціального_страхування_України

42. Офіційний сайт фонду соціального страхування України : [Електронний ресурс]. – Режим доступу: http://www.fssu.gov.ua/fse/control/main/uk/publish/article/954974

43. Сайт Клінського інституту охорони і умов праці : [Електронний ресурс]. – Режим доступу : http://www.kiout.ru

44. Офіційний сайт комісії з охорони та безпеки праці Великобританії : [Електронний ресурс]. – Режим доступу : https://www.hse.gov.uk/index.htm

45. Тарасова В.В. Екологічна статистика.[Текст]/В.В.Тарасова.- Київ: «Центр учбово\ї літератури», 2008 ро.-391с.

46. Запольський А.К. Основи екології [Текст]: підр. для студ. техн. технол. спец. вищ. навч. закл. / А. К. Запольський, А.І. Салюк; за ред. К.М. Ситника. К.: Вища школа, 2001.- 358с. ISBN 966-642-059-7.

47. Лукашев К.И. и др. Человек и природа (геохим. и эколог. аспекты рацион. природопользования) / АН БССР, Ин-т геохимиии и геофизики. - Минск: Наука и техника, 1984 - 295с.: ил.

48. Маликов, У. М. Организация гражданской защиты на промышленных объектах [Текст] : учеб. пособие для техникумов / У.М.Маликов. - М. : Высшая школа, 1989. - 27 с.

49. Основи запобігання виникненню надзвичайних ситуацій [Текст] : навч. посіб. для студ. вищих навч. закладів / [Бедрій Я. І.; Джигирей В. С.; Кидисюк, А. І. та ін.]; за ред. В. С. Джигирей ; Український держ. лісотехнічний ун-т, Львівський електротехнікум зв'язку. - Л. : [б.в.], 1999. - 239 с. Альтернативна назва :Надзвичайні ситуації техногенного походження. - ISBN 5-7865-2682-6.

50. Лукашев К.И. и др. Человек и природа (геохим. и эколог. аспекты рацион. природопользования) / АН БССР, Ин-т геохимиии и геофизики. - Минск: Наука и техника, 1984 - 193с.: ил.

ДОДАТКИ

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

VII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ »

11–12 грудня 2019 року

ТЕРНОПІЛЬ
2019

М. Садівник
МАШИННЕ НАВЧАННЯ У БРАУЗЕРІ З ВИКОРИСТАННЯМ TENSORFLOW.JS 89
Р. Самець
ПІДВИЩЕННЯ ПРОДУКТИВНОСТІ ОЗОНОГЕНЕРАТОРІВ ДЛЯ МЕДИЧНИХ ОЗОНОТЕРАПЕВТИЧНИХ СИСТЕМ 90
Я. Самиця, М. Горалечко, Ю. Дзига
ІЄРАРХІЧНА СТРУКТУРА МОДЕЛЕЙ ЯКОСТІ СИСТЕМ ЕЛЕКТРОННОЇ КОМЕРЦІЇ 91
Я. Самиця, С. Магула
ПРИНЦИПИ ІНТЕГРАЛЬНОЇ ОЦІНКИ РІВНЯ ЯКОСТІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ АВТОМАТИЗОВАНИХ СИСТЕМ КЕРУВАННЯ 93
Т. Сачик, Н. Загородна
ЗАХИСТ ПЕРСОНАЛЬНОЇ ІНФОРМАЦІЇ В ЗАДАЧАХ АНАЛІЗУ ТА ОБРОБКИ ВЕЛИКИХ ДАНИХ 95
Д. Северин
ПРОГРАМНИЙ ЗАСІБ ДЛЯ УПРАВЛІННЯ ПРОЦЕСОМ МІГРАЦІЇ ВІРТУАЛЬНИХ МАШИН В ОБЧИСЛЮВАЛЬНІЙ ХМАРІ 96
О. Ситник, А. Лазорко
МЕТОД РЕПЛІКАЦІЇ ДАНИХ З ВИКОРИСТАННЯМ NFC-ТЕХНОЛОГІЇ 97
Т. Склярова, О. Палка
ІСТОРІЯ РОЗВИТКУ ГЕОІНФОРМАЦІЙНИХ СИСТЕМ 98
В. Соборук, Л. Матійчук
ЗАДАЧІ ТЕСТУВАННЯ СИСТЕМ МОБІЛЬНОГО ЗВ'ЯЗКУ 99
А. Тарапата, М. Іваник
ВИКОРИСТАННЯ МЕТОДУ АНАЛІЗУ ІЄРАРХІЙ ДЛЯ ОЦІНЮВАННЯ ЯКОСТІ ПРОЕКТУ КОМП'ЮТЕРНИХ МЕРЕЖ 100
А. Тарапата, А. Гулик
ВИКОРИСТАННЯ МОДЕЛЕЙ ЯКОСТІ ДЛЯ РОЗРОБКИ ВИМОГ 101
П. Телевяк, Л. Матійчук
АНАЛІЗ СУЧАСНИХ МЕТОДІВ ЗАХИСТУ ІНФОРМАЦІЇ ТА ЇХ КЛАСИФІКАЦІЯ 102
О. Топчак, Н. Куцанець
РЕКОМЕНДАЦІЙНА СИСТЕМА РЕАБІЛІТАЦІЇ ХВОРИХ З ПРОБЛЕМАМИ ОПОРНО-РУХОВОГО АПАРАТУ 103
Б. Тригубець
РОЗРОБКА CMS ТА МЕТОДІВ ЗАХИСТУ WEB-САЙТІВ НА ЇЇ ОСНОВІ 104
Л. Тучапський, М. Поліщук
ЦИФРОВА ФІЛЬТРАЦІЯ РАДІОСИГНАЛІВ 105
М. Шмигельський, В. Ліщинський
ОСНОВНІ МЕТОДИ І ПРИЙОМИ ПОРУШЕННЯ БЕЗПЕКИ СУЧАСНИХ БЕЗДРОТОВИХ МЕРЕЖ 106
А. Шум'як, О. Палка, І. Пятківський
АНАЛІЗ ІНТЕЛЕКТУАЛЬНИХ ТРАНСПОРТНИХ СИСТЕМ 107
Р. Яворський, В. Амбок, В. Леньо
ПРОБЛЕМИ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ ПРИ РОЗГОРТАННІ СИСТЕМ ВИЯВЛЕННЯ ВТОРГНЕНЬ 108

УДК 004.85
М. Садівник
(Тернопільський національний технічний університет імені Івана Пулюя)

МАШИННЕ НАВЧАННЯ У БРАУЗЕРІ З ВИКОРИСТАННЯМ TENSORFLOW.JS

UDC 004.85
M. Sadivnuk
(Ternopil Ivan Puluj National Technical University, Ukraine)

MACHINE TRAINING IN USE BROWSER TENSORFLOW.JS

Останнім часом все більше уваги приділяється штучному інтелекту та машинному навчанню. На перший погляд ці концепції абсолютно не пов'язані з веб-розробкою та технологіями JavaScript. Зазвичай вони асоціюються з середовищем Python / R або навіть бібліотеками C ++. Одним з найпопулярніших фреймворків, яким користується все більша кількість розробників, є TensorFlow. Він був розроблений в Google в 2011 році. Він створений на мові C ++ і може використовуватись різними мовами, такими як Python, R або Java.

Тривалий час, використовуючи JavaScript, машинне навчання здійснювалося на стороні сервера. Навчена модель була розгорнута на сервері і, наприклад, доступна через протокол HTTP. Веб-додаток надсилав запит з необхідними даними за допомогою JS для отримання результату від сервера. У березні 2018 року з'явився TensorFlow.js і з його допомогою можна писати застосунки для машинного навчання / глибокого навчання за допомогою JavaScript, не використовуючи програм, які обробляються на стороні сервера. TensorFlow.js можна використовувати для визначення, навчання та запуску моделей машинного навчання повністю у браузері. З точки зору користувача, це дуже просто і зручно; не потрібно встановлювати бібліотеки чи драйвери. Слід просто відкрити веб-сторінку і програма готова до запуску.

Як відомо, JavaScript є однопоточним і виконується лише на центральному процесорі, який призначений для переключення між програмами та задач з великою затримкою, а не для високої пропускної здатності. Графічний процесор розроблений для високих навантажень та для високої пропускної здатності. З точки зору розробки, різниця між цими процесорами у тому, що на графічному процесорі робота буде виконуватись паралельно та багатопоточно і завдяки цьому обчислення, які обробляються графічним процесором є у багато раз швидшими, ніж якби вони оброблялись центральним процесором. Tensorflow.js автоматично підтримує WebGL, який є браузерним інтерфейсом для OpenGl, і дозволяє виконання JavaScript коду за допомогою графічного процесора.

Tensorflow.js може бути використаний для імпорту вже існуючої навченої моделі у веб-інтерфейс; для перенавчання існуючої моделі; для моделі, яка може бути розроблена, навчена та запущена у браузері.

Також машинне навчання у браузері має низку переваг. Оскільки, програма не відсилає ніяких запитів до сервера для обробки, користувач може не перейматись за коонфіденційність своїх даних. Tensorflow.js застосунки можуть бути відкриті у будь-якому сучасному браузері та на будь-якій операційній системі. Також завдяки відсутності обробки даних на сервері, різні затримки по типу «запит – відповідь» будуть відсутні, що збільшує швидкість обробки даних.

Література

1. Герон А. Hands–On Machine Learning with Scikit–Learn and TensorFlow 2019. – С. 247–275
2. Tensorflow.js Essential documentation [Електронний ресурс]. – Режим доступу: https://www.tensorflow.org/guide