**Міністерство освіти і науки України Ніжинський державний університет імені Миколи Гоголя**

**Навчально-науковий інститут точних наук і економіки**

**Кафедра інформаційних технологій та аналізу даних**

*Середня освіта (Математика)*

*014.04 Середня освіта (Математика)*

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня **магістр**

Розробка та застосування машиного навчання

студентки **Бузиль Вікторії Вікторівни**

Науковий керівник:

Казачков Іван Васильович,

доктор фізико-математичних наук, професор

Рецензензенти:

Допущено до захисту

Зав. кафедри\_\_\_\_\_\_\_\_\_\_\_\_ Казачков І.В.

Ніжин - 2019 рік

ЗМІСТ

[ВСТУП 2](#_30j0zll)

[1 АНАЛІЗ ЗАДАЧ ВИКОРИСТАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ 4](#_1fob9te)

[1.1 Фундаментальні аспекти теорії машинного навчання 4](#_3znysh7)

[1.2 Математичне забезпечення методів штучного інтелекту 11](#_2et92p0)

[1.3 Постановка задачі 16](#_tyjcwt)

[2 ОГЛЯД ІСНУЮЧИХ МЕТОДІВ ТА ЗАСОБІВ МАШИННОГО НАВЧАННЯ 19](#_3dy6vkm)

[2.1 Класифікація задач в області машинного навчання 19](#_1t3h5sf)

[2.2 Програмні пакети для реалізації алгоритмів машинного навчання та аналізу даних 32](#_4d34og8)

[2.3 Характеристика сфери застосування та перспектив розвитку алгоритмів машинного навчання 36](#_2s8eyo1)

[3 ВИКОНАННЯ АЛГОРИТМУ МАШИННОГО НАВЧАННЯ ЗАСОБАМИ MATLAB 41](#_17dp8vu)

[3.1 Моделювання нейронної мережі на основі Neural Network Toolbox 41](#_3rdcrjn)

[3.2 Аналіз отриманих результатів 46](#_26in1rg)

[ЗАКЛЮЧЕННЯ 50](#_lnxbz9)

[СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 52](#_35nkun2)

# ВСТУП

Теорія машинного навчання зародилася практично одночасно з появою перших комп'ютерів і протягом останніх років є напрямом, що активно розвивається.   
 **Актуальність** роботи полягає в тому, що все більшої уваги потребують питання методологій та принципів використання, комбінування і вибору конкретних моделей і методів машинного навчання. Багаторічний прогрес в розробці методів машинного навчання породив не тільки різноманітні математичні, програмні та навіть апаратні рішення, призначені для задач інтелектуального і генеративного аналізу даних в самих різних областях, але зустрів на своєму шляху чимало труднощів і перешкод.

Основна складність використання машинного навчання - величезна кількість розрізнених методів, кожен з яких має свої особливості, область використання і переваги. В індустрії машинного навчання давно назріла необхідність створення більш простих у використанні підходів, які можна застосовувати в широкому колі користувачів. Часто процес використання систем машинного навчання передбачає виконання повного циклу прикладних досліджень з обробки даних, виділення ознак, вибору виду моделі, навчання параметрів, гіперпараметрів тощо.

**Метою** роботи є огляд актуальних прийомів, технологій і методик, що застосовуються при вирішенні прикладних задач машинного навчання. **Об’єктом** дослідження є процеси розробки та застосування машинного навчання.   
 **Предметом** дослідження є модель навчання нейронної мережі засобами пакету Matlab. Для досягнення поставленої мети необхідно виконати наступні **задачі**:

* розкрити фундаментальні аспекти теорії машинного навчання;
* виконати аналіз математичного забезпечення методів штучного інтелекту;
* розглянути основні задачі машинного навчання;
* розглянути програмні пакети для реалізації алгоритмів машинного навчання та аналізу даних;
* надати характеристику сфері застосування та перспективам розвитку методів машинного навчання
* продемонструвати на прикладі машинне навчання.

# 1 АНАЛІЗ ЗАДАЧ ВИКОРИСТАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ

## 1.1 Фундаментальні аспекти теорії машинного навчання

Методи машинного навчання описують розділ штучного інтелекту, який вивчає методи побудови моделей, здатних до навчання. Машинне навчання тісно пов'язане з інтелектуальним аналізом даних (Data Mining); іноді його навіть розглядають як складову частину інтелектуального аналізу даних.

У загальному вигляді можна визначити головну функцію машинного навчання як автоматизацію (повну або часткову) рішення складних професійних завдань в різних областях діяльності [5].

В реальності вихідні дані про об'єкти у вигляді даних навчальної вибірки можуть бути неповними, неточними і навіть нечисловими. У кожному разі використовуються різні методи машинного навчання, тому вони досить різноманітні та численні.

Машинне навчання працює з об'єктами - елементарними одиницями даних, що виникають в конкретних завданнях, які характеризуються змінними, які спостерігаються, - і прихованими змінними - , які приймають значення з деяких заздалегідь відомих множин. Головним завданням машинного навчання є автоматичне визначення взаємозалежностей між змінними, що спостерігаються, та прихованими змінними об'єкта, з тим, щоб для довільного об'єкта за його відомими компонентам можна було оцінити можливі значення прихованих компонент. Як правило, можливі взаємозалежності попередньо задаються за допомогою параметричних вирішальних правил, визначених значенням параметрів – ваг . Конкретні значення вагових коефіцієнтів визначаються в результаті навчання з використанням навчальної вибірки, яка представляє собою множину об'єктів з відомими прихованими змінними та змінними, які спостерігаються, що відповідає навчанню «з учителем» або тільки змінними, які спостерігаються, у разі навчання «без учителя» [1]. При цьому задача визначення вагових коефіцієнтів вирішального правила за навчальною вибіркою називається задачею навчання або налаштування (training), а задача визначення допустимих значень прихованої змінної по заданим спостережуваним компонентам об'єкта і заданим вагам вирішального правила – задачею виведення (inference). Зазвичай передбачається, що кожен об'єкт описується одним і тим же набором змінних, а номенклатура спостережуваних і прихованих змінних для всіх об'єктів однакова. Прикладом такої стандартної задачі є задача класифікації, в якій прихована змінна для кожного об'єкта одна і приймає значення з кінцевої дискретної множини, а кожна спостережувана змінна може приймати дійсні, або дискретні значення. Якщо прихована змінна об'єкта є не дискретною, а безперервною, задача називається задачею відновлення регресії, що є ще однією зі стандартних і добре вивчених задач машинного навчання.

Загальна задача процесу машинного навчання проілюстрована за наведеною на рис. 1.1 схемою

Рисунок 1.1 – Загальна постановка задачі машинного навчання

У різний час робилися неодноразові спроби ввести деяку універсальну мову опису різних постановок і методів розв'язання задач машинного навчання. Починаючи з 90-х р.р. минулого століття широкого поширення отримав байєсівський формалізм [5, 8]. При його використанні передбачається, що залежність між спостережуваними змінними об'єкта, вагами вирішального правила і прихованими змінними об'єкта моделюється з допомогою спільного розподілу на ці групи змінних p(). Якщо має місце тільки задача визначення прихованих змінних за спостережуваними, розглядаються дискримінативні моделі (discriminative models) p(|). Значення спостережуваних змінних в цьому випадку моделюються через припущення, що вони є відомими на всіх етапах виконання завдання, і спільний розподіл стає простішим. В стандартних постановках задачі машинного навчання передбачалося, що приховані змінні кожного об'єкта залежать тільки від спостережуваних змінних цього об'єкта, причому вид залежності визначається параметрами . Це відповідає представленню:

(1.1)

При використанні такого формалізму задача налаштування параметрів вирішується, наприклад, знаходженням найбільш вірогідного значення:

(1.2)

В той час задача виведення вирішується наступним шляхом:

(1.3)

Таким чином, для формулювання і рішення задачі машинного навчання достатньо знати дві функції: p(| , ) і p (). Перша функція, звана функцією правдоподібності, характеризує ступінь «істинності» отриманого прогнозу прихованої змінної, другий компонент, є апріорним розподілом, при зміні якого існує можливість впливу на результат процедури налаштування, тобто на . При цьому спосіб адекватного визначення апріорного розподілу неочевидний. Апріорний розподіл є ефективним способом контролю складності вирішального правила і дозволяє здійснювати регуляризацію процедури налаштування. Замість знаходження ваг, що забезпечують найменшу помилку прогнозу на навчальній вибірці та загрожує ефектом перенавчання, доцільно дещо зменшити точність, але забезпечити здатність отримати ту ж помилку прогнозу на інших об'єктах генеральної сукупності. В будь-якій моделі машинного навчання можна виділити найпростіше вирішальне правило (наприклад, що відповідає нульовим значенням ваг), в якому міститься мода унімодального апріорного розподілу. Чим більше відстань поточних значень ваг від моди, тим менше значення p(). Ширина апріорного розподілу задається параметром регуляризації, який може бути досить ефективно знайдений процедурою змінного контролю або байєсівською процедурою вибору моделі [6].

Перевагою байєсівського формалізму виявилася можливість враховувати численні апріорні знання про можливі залежності між спостережуваними та прихованими змінними об'єктів, які є в багатьох прикладних задачах. Наприклад, відомо, що надійність позичальника (прогнозована змінна) повинна позитивно корелювати з його доходом і освітою (спостережувані змінні). Такі попередні значення у алгоритмах загального призначення, виражені у вигляді апріорного розподілу на , дозволили домогтися значного збільшення точності і знизити ефект перенавчання, завдяки адаптації їх під специфіку конкретного завдання.

Зведення практично будь-якої задачі машинного навчання до подібного формалізму відкриває уніфікований спосіб аналізу різних моделей машинного навчання, наприклад, з метою дослідження їх узагальнюючої здатності або вироблення ефективних наближених методів настройки і виведення загального призначення.

У рамках байєсівського формалізму розроблений ряд спільних методів для оцінки апостеріорних розподілів, байєсівського виведення, автоматичного вибору моделі та ін. Не менш важливим успіхом байєсівського формалізму стала можливість успішного узагальнення результатів і методів класичного машинного навчання на абсолютно нові задачі [6].

Нейронна мережа [1-2] являє собою потужний механізм машинного навчання, який в основному імітує те, як вчиться людський мозок. Штучна нейронна мережа – набір нейронів, з'єднаних між собою. Штучний нейрон за своїми властивостями нагадує біологічний нейрон. На вхід штучного нейрона надходить деяка множина сигналів x1, x2, ..., xn, кожен з яких є виходом іншого нейрона. Кожен вхід множиться на відповідну вагу w1, w2, ..., wn, і надходить на підсумовуючий блок, позначений Σ (рис. 1.2), який складає зважені входи алгебраїчно, створюючи вихід – NET [1].

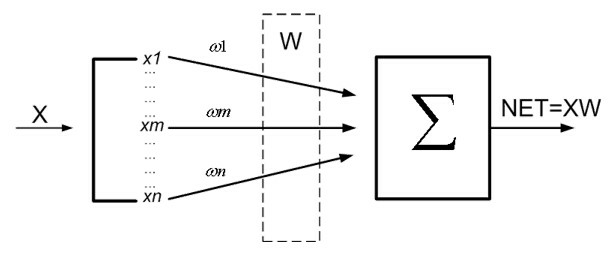


Рисунок 1.2 – Загальний вигляд нейронної мережі

Найпростішою нейронною мережею вважається персептрон (рис. 1.2), який є однією з найпопулярніших реалізацій нейронних мереж. Причиною його популярності є відносна простота реалізації на тлі універсальності і широкого кола завдань, які можуть вирішувати персептрони.

Для вирішення більш складних завдань використовують багатошарові нейронні мережі [1-2, 8]. Але для навчання багатошарових нейронних мереж потрібні більш складні алгоритми навчання нейронних мереж. Штучна нейронна мережа навчається за допомогою деякого процесу, що модифікує її ваги. Якщо навчання успішне, то передача мережі множини вхідних сигналів призводить до появи бажаної множини вихідних сигналів.

Навчання нейронної мережі полягає в рішенні задачі багатомірної оптимізації, яка являє собою здачу зменшення суми квадратів похибок між експериментальним та модельним значенням [1]:

Σ *(Ye-Ym)2 →* min (1.4)

Є два класи навчальних методів: детерміністський і стохастичний. Детерміністський метод навчання крок за кроком здійснює процедуру корекції ваги мережі, засновану на використанні їх поточних значень, а також величин входів, фактичних виходів і бажаних виходів. Стохастичні методи навчання виконують псевдовипадкові зміни величин ваг, зберігаючи ті зміни, які ведуть до покращення результату.

Методи глибинного навчання (deep learning) використовують механізми нейронних мереж [8]. Недоліками традиційних нейронних мереж стали:

* критична залежність якості настройки ваг мережі від вибору початкового наближення;
* велика схильність до перенавчання укупі зі слабкими можливостями контролю узагальнюючої здатності мережі;
* велика кількість локальних мінімумів функціонала якості, більшість з яких є несприйнятими.

З іншого боку, незаперечною сильною стороною нейронних мереж стало відкриття методу зворотного поширення помилки, який дозволив відстежувати вплив внутрішніх шарів мережі на якість прогнозу прихованих змінних об'єктів навчальної вибірки.

Останнім часом став активно розвиватися напрямок, який одержав назву глибинного навчання [5]. В його основі лежать нейронні мережі, які зазнали значних змін:

* Глибинне навчання будує не дискримінативність, а породжує моделі, в яких моделюється загальний розподіл p(), на відміну від дискримінативних моделей, що дозволяє, наприклад, генерувати нові об'єкти.
* У найбільш поширеній постановці всі змінні об'єктів передбачаються бінарними. Це полегшує моделювання залежностей між змінними об'єкта.
* Кожен шар мережі спочатку навчається незалежно, проходячи процедуру преднавчання. Це дозволяє знайти хороше початкове наближення для подальшого запуску алгоритму зворотного поширення помилки. Кожен шар, в залежності від обраної моделі, являє собою обмежену машину Больцмана або згорткову нейронну мережу.
* Для навчання використовуються сотні тисяч і мільйони об'єктів. Такі гігантські вибірки дозволяють налаштовувати мережі з десятками тисяч параметрів, без ризику перенавчання. Навчені таким чином мережі, не просто дозволяють моделювати складні об'єкти (наприклад, тексти або зображення), але і генерують в процесі навчання інформативні ознакові описи, які можуть бути використані іншими, більш простими алгоритмами машинного навчання як спостережувані змінні об'єкта.

Методологія глибинного навчання дозволила домогтися нових результатів при навчанні на великих і надвеликих обсягах даних. В даний час вона є одним з найбільш перспективних шляхів розвитку машинного навчання.

Традиційно, методи непараметричної статистики визначалися як розділ статистики, в якій число параметрів, що описують дані (наприклад, параметри щільності розподілу об'єктів) не фіксоване, а зростає зі збільшенням числа об'єктів [6]. Широкого застосування набула задача визначення числа кластерів в зростаючій вибірці об'єктів. Дана задача тим більш актуальна, що загальноприйнятих методів визначення на сьогоднішній день не існує. Чим більше об'єктів надходить в розпорядження, тим з більшим дозволом ми можемо знаходити в них структуру, виділяючи кластери схожих між собою об'єктів. У разі досить неоднорідної вибірки число кластерів має поступово збільшуватися в міру надходження нових об'єктів. Тому постає задача визначення швидкості зростання числа кластерів з ростом даних вибірки. Формально дана залежність може бути описана формулою Байєса, яка об'єднує апріорні уявлення з поточними спостереженнями:

(1.5)

У непараметричному випадку розподіл задається за допомогою випадкових процесів. В даному випадку, це процес Діріхле [6]. З його допомогою, вдається не тільки розрахувати для будь-якого розбиття довільного числа об'єктів на кластери його апріорну ймовірність, але і врахувати характеристики об'єктів (їх спостережувані змінні), щоб перейти до апостеріорного розподілу на можливе розбиття.

Як це часто буває при застосуванні байєсівських методів, апостеріорний розподіл має гострий пік, який відповідає сталому розбиттю вибірки об'єктів на деяке число кластерів. Фактично, процес Діріхле дозволяє задавати розподіл над можливими дискретними розподілами. При виведенні використовуються наближені методи Монте-Карло з марківськими ланцюгами і методи варіаційного виведення. Описана схема допускає численні узагальнення на випадок ієрархій кластерів та множинних вибірок.

Ще однією областю машинного навчання, яка активно розвивається, є навчання з підкріпленням, призначене для навчання агентів (автономних модулів, самостійно приймаючих рішення в реальному часі на підставі наявних даних) в умовах невизначеності, що породжується як неповнотою інформації про навколишнє оточення, так і можливими діями інших агентів. Залежно від поточного стану середовища і дій агентів розраховується функція вигоди, яку отримає агент в наступний момент часу. В ролі спостережуваних змінних об'єкта виступає інформація, що надається агентом, а прихованими змінними є довгострокові оцінки отриманої вигоди. Важливою перевагою алгоритмів навчання з підкріпленням є можливість навчання агента «з нуля» за рахунок балансованого поєднання режимів «дослідження-використання» і вивчання стратегій, які дозволяють жертвувати малим зараз заради отримання більшої вигоди в подальшому [6].

Алгоритми навчання з підкріпленням знайшли широке застосування не тільки в таких традиційних областях як робототехніка, але і, наприклад, на фондових ринках.

## 1.2 Математичне забезпечення методів штучного інтелекту

При реалізації алгоритмів машинного навчання виникає задача визначення найкращих значень параметрів або структури об'єктів. Така задача є оптимізаційною. В сучасних умовах оптимізаційні задачі і задачі прийняття рішень моделюються і вирішуються в самих різних областях техніки [3]. До навичок математичного обґрунтування прийняття рішень відносяться навички математичного моделювання оптимізаційних задач, вибору адекватного математичного забезпечення (методу, алгоритму, програмної системи) з необхідним обґрунтуванням, аналізу отриманих результатів та їх інтерпретації в термінах предметної області.

Завдання оптимізації в цілому зводиться до задачі пошуку екстремуму (мінімуму або максимуму) цільової функції з заданими обмеженнями. Її математична постановка виглядає наступним чином: необхідно визначити значення вектору змінних x = (x1, x2,..., xm), які задовольняють обмеженням виду [3-4]:

*gi(x1, x2, . . . , xm) ≤ bi* (1.6)

Для всіх i = 1,. . . , K і при яких досягається максимум або мінімум цільової функції f (x1, x2,..., xm):

*f (x1, x2,..., xm)* → (max, min). (1.7)

Допустимим рішенням задачі називається таке рішення, яке задовольняє її обмеженням (1). Сукупність допустимих рішень задачі називають областю допустимих рішень (ОДР). Остаточним рішенням задачі є пара (x\*, f\*(x\*)), що складається з оптимального рішення і оптимального значення цільової функції.

Методи математичного програмування дають велику різноманітність алгоритмів рішення даного завдання. В цілому алгоритми пошуку реалізують методи спуску до екстремуму, при яких значення цільової функції послідовно покращується до досягнення екстремуму. Залежно від можливості знаходження алгоритмом локального або глобального екстремуму, вони діляться на алгоритми локального і глобального пошуку [4].

Алгоритми пошуку локального екстремуму призначені для визначення одного з локальних екстремумів на множині допустимих рішень, в якій цільова функція приймає максимальне або мінімальне значення. При їх побудові можуть використовуватися як детермінований спуск в область екстремуму, так і випадковий пошук. Серед детермінованих методів розрізняють методи нульового порядку і градієнтні (1-го і 2-го порядку). Перші засновані на обчисленнях тільки значень оптимізованої функції. Другі використовують часткові похідні відповідного порядку. Для пошуку екстремуму у випадках, коли вид оптимізованої функції відомий не повністю, або її структура занадто складна, застосовуються методи стохастичного програмування або нейронних мереж. Ефективність процедури пошуку оптимуму - можливість відшукання рішення і збіжність до вирішення по швидкості залежать від виду функції і застосовуваного для неї методу. Виникає необхідність визначення стратегії кожного методу більш детально, досліджуючи для визначеності мінімізацію цільової функції.

З прямих методів нульового порядку найбільш відомі методи [3-4]:

- координатного спуску - почергова оптимізація параметрів уздовж осей одним з відомих одновимірних методів;

- спірального координатного спуску;

- обертових координат (метод Розенброка);

- симплексного пошуку;

- Хука-Дживса з пошуком за зразком і ін.

Метод координатного спуску [3-4] полягає в тому, що в якості напрямків траєкторії спуску від попередньої точки пошуку X(k-1) до подальшої X(k) приймаються по черзі напрямки координатних осей Xi (i = 1, 2,..., N). Після спуску на один крок по координаті X1 відбувається перехід до спуску на один крок по координаті X2, а потім рух уздовж координати X3 і т.д., поки не буде знайдена наступна точка пошуку X(k) з координатами X1(k), X2(k),. . . , Xn(k). Рух за траєкторією спуску від попередньої точки X(k-1) до подальшої X(k) триває до тих пір, поки не буде досягнуто точку мінімуму X\* цільової функції, що визначаються точністю обчислень. Для пошуку координат точки X(k) на кожному кроці ітерації можна використовувати будь-який з методів одновимірної мінімізації: метод золотого перетину, метод поділу відрізка навпіл, метод інтерполяції-екстраполяції та ін.

Метод спірального координатного спуску [3] відрізняється від розглянутого вище лише тим, що крок h змінюється кожен раз при переході від пошуку мінімуму за однією змінною до пошуку мінімуму за іншою змінною. У тривимірному просторі це нагадує спуск по спіралі. Зазвичай цей метод дає деяке скорочення часу пошуку. Методи координатного спуску недостатньо ефективні для поверхонь з "оврагами", так як в цьому випадку отримання рішення з необхідною точністю не гарантоване. Це викликано тим, що в разі "оврагу", повернутого щодо осей, спроба просування в будь-якому напрямку може викликати "погіршення" цільової функції. У той же час просування уздовж "оврагу" може давати "покращення" цільової функції.

Метод Розенброка [3] спрямований на усунення одного з недоліків метода покоординатного спуску - високу чутливість до вибору системи координат. В процесі пошуку методом Розенброка проводиться поворот координатних осей так, щоб одна з осей була направлена вздовж напрямку "оврагу". На кожній ітерації процедура здійснює ітеративний пошук вздовж n лінійно незалежних і ортогональних напрямків. Коли отримано нову точку в кінці ітерації, будується нова множина ортогональних векторів. Побудова напрямків пошуку полягає в наступному: нехай d1,. . . , dn - лінійно незалежні вектори, за нормою рівні одиниці. Приймемо, що ці вектори взаємно ортогональні (di \* dj = 0) для i=j. Починаючи з поточної точки x(k), цільова функція послідовно мінімізується уздовж кожного з напрямків, в результаті чого виходить точка x(k + 1). Зокрема, x(k + 1)-x (k) = Σ (λj \* dj), де λj - довжина кроку у напрямку dj. Новий набір напрямків q1,. . . , qn будується за допомогою процедури Грамма-Шмідта [3]. Нові напрямки, побудовані описаним чином, є лінійно незалежними і ортогональними.

Таким чином, метод Розенброка дозволяє уникнути проблем, пов'язаних з отриманням рішення заданої точності для поверхонь c "оврагами". Однак такий підхід порівняно збільшує час пошуку, що є відносним недоліком описаного методу.

Ітераційні процеси оптимізації, напрямок пошуку яких на кожному кроці співпадає з антиградієнтом функції, називаються градієнтними методами [1, 3-4].

Алгоритм найшвидшого спуску реалізує ітераційну процедуру руху до мінімуму з довільно вибраної початкової точки в напрямку найбільш сильного зменшення функції, визначеному в околиці поточного значення аргументу мінімізованої функції. Такий напрям протилежний напрямку, що задається вектором градієнта ∇f (x) функції, що мінімізується f(x). Загальна формула для знаходження значення аргументу:

(1.8)

|  |  |
| --- | --- |
|  | (1.9) |

Алгоритми найшвидшого спуску розрізняються за способом визначення кроку λ(k). Якщо крок λ(k) не залежить від k (є постійним), то в околиці екстремуму спостерігатимуться незгасаючі коливання, амплітуда яких залежить від величини λ і від форми функції, що мінімізується. Використання постійного кроку має наступні особливості:

- дозволяє побудувати найбільш простий варіант алгоритму;

- при великих значеннях λ забезпечує швидкий рух до екстремуму, але призводить до помітних змін на околицях;

- при малих значеннях λ призводить до низької швидкості збіжності до екстремуму;

- інформація про прийнятну величину кроку λ може бути отримана тільки в ході налагодження алгоритму, тому що ніяка інформація про властивості функції, що мінімізується, не використовується.

Покращенням методу з постійним кроком, що дозволяє усунути коливання в околиці екстремуму без істотного ускладнення алгоритму, є використання кроку λ, величина якого зменшується по ходу ітераційного процесу і залежить від номера ітерації k. Поряд з перевагами такий підхід має і недолік, а саме незв'язаність значень λ(k) з формою функції. Якщо далеко від екстремуму функція f(x) має малий градієнт, швидкість збіжності може виявитися неприпустимо повільної. Ця проблема вирішується шляхом модифікації алгоритму, яка може бути здійснена різними способами.

Існує алгоритм найшвидшого спуску з кроком, довжина якого залежить від властивостей функції, що мінімізується (використання других похідних). Метод Ньютона [1, 3-4] заснований на квадратичній апроксимації функції, що мінімізується в околиці точки x(k). Мінімум квадратичної функції легко знайти, прирівнюючи її градієнт нулю. Можна відразу ж обчислити положення екстремуму і вибрати його в якості наступного наближення до точки мінімуму. Обчислюючи точку нового наближення за формулою: *xk+1 = xk + Δxk* і розкладуючи *f(x(k+1))* в ряд Тейлора, отримаємо формулу квадратичної апроксимації функції в наступній точці.

Переваги методу Ньютона:

- якщо мінімізована функція є квадратичною, то метод дозволить знайти мінімум за один крок;

- якщо мінімізована функція належить до класу поверхонь обертання (тобто володіє симетрією), то метод також забезпечує збіжність за один крок;

- якщо функція несиметрична, то метод не забезпечує збіжність за кінцеве число кроків. Але для багатьох функцій досягається набагато більш висока швидкість збіжності, ніж при використанні інших модифікацій методу найшвидшого спуску.

Недоліки методу Ньютона пов'язані з необхідністю обчислень і обернення матриць других похідних. При цьому не стільки витрачається машинний час, скільки можуть з'явитися значні обчислювальні похибки, якщо матриця виявиться погано обумовленою.

## 1.3 Постановка задачі

Центральною задачею систем, що функціонують на принципах машинного навчання, є реалізація алгоритму налаштування параметрів системи. З поняттям навчання тісно пов'язане поняття узагальнюючої здібності. Узагальнююча здатність - це властивість моделі відображати вихідні дані в необхідні результати (*X → Y*) на всіх множинах вихідних даних (у всіх сценаріях, а не тільки на тренувальних прикладах). Величину узагальнення оцінюють через зворотну величину - відхилення або помилку. Помилка – це чисельно виражена різниця між відповіддю моделі і необхідним реальним значенням. У більш загальному сенсі узагальнююча здатність - здатність моделі знайти певний закон, який буде описувати невідомий прихований взаємозв'язок вхідних і вихідних даних. Таким чином, спираючись на поняття узагальнюючої здатності, можна виділити основні проблемні питання машинного навчання:

1. Кількість даних для знаходження в них корисних знань.

2. Спроможність моделі навчитися на наявних даних.

3. Поширення отриманого закон на всю можливу множину X.

4. Оцінка якості роботи навченої моделі.

З математичної точки зору задача наближення знаходиться в області оптимізації або мінімізації помилки [1]:

*Е = | y - y '|* → min (1.10)

Таким чином, загальний алгоритм вирішення завдань в сфері машинного навчання складається з наступних кроків:

1. Математична постановка задачі;
2. Обробка даних і виділення ключових ознак;
3. Побудова моделі (або моделей);
4. Навчання моделі (моделей) і оцінка якості;

Експлуатація моделі при досягненні необхідної якості, або повернення до одного з попередніх кроків (переналаштування моделі, видобуток нових даних і т. п.).

Метою роботи є огляд актуальних прийомів, технологій і методик, що застосовуються при вирішенні прикладних задач машинного навчання. Для досягнення поставленої мети необхідно виконати наступні задачі:

* розкрити фундаментальні аспекти теорії машинного навчання;
* виконати аналіз математичного забезпечення методів штучного інтелекту;
* розглянути основні задачі машинного навчання;
* розглянути програмні пакети для реалізації алгоритмів машинного навчання та аналізу даних;
* надати характеристику сфері застосування та перспективам розвитку методів машинного навчання
* продемонструвати на прикладі машинне навчання.

# 2 ОГЛЯД ІСНУЮЧИХ МЕТОДІВ ТА ЗАСОБІВ МАШИННОГО НАВЧАННЯ

## 2.1 Класифікація задач в області машинного навчання

Машинне навчання, як правило, має справу з обробкою дуже великих обсягів даних. Серед завдань, що вирішуються за допомогою методів машинного навчання, можна виділити наступні основні класи [6]:

1. Відновлення щільності розподілу ймовірності за вибірковими даними - спостережуваними значеннями випадкової величини. Ця задача вирішується також в рамках математичної статистики - по вибірці будується гістограма, висувається гіпотеза про закон розподілу, знаходяться вибіркові оцінки параметрів передбачуваного розподілу, після чого гіпотеза перевіряється за тим або іншим критерієм;
2. Знаходження коефіцієнтів регресії. Це досить широкий клас задач, рішення яких знаходить в площині різних дисциплін: аналіз даних, економетрика тощо. Задачі регресійного аналізу, в свою чергу, допускають різні класифікації. Так, наприклад, в залежності, від кількості факторів (кількості спостережуваних властивостей об'єктів) розрізняють завдання однофакторної та множинної регресії; в залежності від використовуваної формули розрізняють лінійну або нелінійну регресію. Знайдені коефіцієнти використовуються для цілей прогнозу. Крім того, за допомогою побудованої регресійної моделі можна оцінити ступінь залежності між спостережуваною ознакою (чинником, предиктором) і змінною відгуку, а також вирішити ряд інших задач);
3. Класифікація - за наявною множиною об'єктів, для яких відомо, до якого класу кожен з них належить, потрібно запропонувати правило, яке дозволить віднести новий об'єкт до одного з можливих класів.

За принципами організації розрізняють два типи машинного навчання:

* індуктивне навчання, або навчання по прецедентах, засноване на виявленні закономірностей в емпіричних даних (поняття «навчання по прецедентах») [6];
* дедуктивне навчання, що припускає формалізацію знань експертів і створення на їх основі комп'ютерних баз знань (поняття «дедуктивне навчання» часто відносять до області експертних систем).

Загальну задачу машинного навчання по прецедентах можна охарактеризувати наступним чином. Є деяка множина об'єктів (або ситуацій) і множина можливих відгуків (реакцій). Між відгуками і об'єктами існує деяка залежність, яка є невідомою. Відома лише деяка кінцева множина прецедентів, тобто пар «об'єкт - відгук», звана навчальною вибіркою. На основі цих даних потрібно відновити залежність відгуку від об'єктів та їх властивостей. Зважаючи на зазначене, завдання машинного навчання полягає в тому, щоб знайти модель у вигляді формули або алгоритму, за допомогою яких для кожного нового об'єкта з набором його властивостей можна отримати відповідний відгук. Оскільки відгук, знайдений за допомогою побудованої моделі, має помилку, виникає також питання вимірювання точності відповідей – оцінки якості моделі. Для цього вводиться деякий функціонал якості [1].

Термін «великі дані» (Big data) увійшов у вживання, коли став можливим збір і зберігання величезних обсягів даних. Традиційні методи машинного навчання не завжди можуть бути застосовані для аналізу вибірок такого розміру, оскільки в них часто неявно передбачається, що вся вибірка поміщається в пам'яті комп'ютера, або ж вони мають недостатньо високі показники масштабованості (швидкості зростання обчислювальної складності залежно від розміру вибірки). Для подолання цих обмежень часто використовуються прийоми з наступних категорій:

1. Розпаралелювання. Незалежні частини алгоритму можуть виконуватися паралельними обробниками (в тому числі на різних комп'ютерах) і в довільному порядку. В деяких випадках паралельної реалізації класичного алгоритму може бути досить для конкретного завдання. В тій чи іншій формі паралельність лежить в основі практично всіх обчислювальних систем, орієнтованих на великі дані. Слід зазначити, що паралельність накладає суттєві обмеження на взаємодію між обробниками, так як накладні витрати на «спілкування» між ними можуть перевищувати виграш від використання великого обчислювального кластера.
2. Апроксимація. Відомо, що багато складних завдань можуть бути вирішені наближено з досить великою (а іноді і контрольованою) точністю, достатньою для даного експерименту. Прикладами можуть служити фільтр Блума або наближений алгоритм пошуку найближчого сусіда, які допускають помилки першого роду, але мають істотно нижчу обчислювальну складність ніж їх «точні» аналоги.
3. Стохастичність (рандомізація). При наявності великого числа незалежних об'єктів у вибірці, багато необхідних статистик можуть бути оцінені за випадковою підвибіркою, при цьому зберігаються теоретичні гарантії оптимальності та збіжності алгоритму. У разі, якщо вибирається підвибірка деякого фіксованого розміру, це дозволяє отримувати алгоритми з сублінійною масштабністю. Найбільш відомим алгоритмом, де застосовується даний підхід, є метод стохастичного градієнтного спуску.

Останнім часом стали набирати популярність потокові алгоритми (streaming algorithms, online learning), здатні навчатися інкрементально в режимі реального часу на даних, що постійно надходять, без необхідності зберігати їх в пам'яті. Попит на них виникає коли дані надходять в таких кількостях і з такою швидкістю, що немає можливості зберігати їх надовго.

Даний тип є найбільш поширеною архітектурою штучних нейронних мереж і в певному сенсі класичної. Дуже часто, коли мова йде про нейронні мережі взагалі, то мається на увазі саме багатошарова мережа прямого поширення. Багато задач класифікації, апроксимації та управління, які вирішуються за допомогою нейронних мереж, вирішуються саме цим типом мережі. Крім того, принципи навчання, розроблені для цього типу мережі, згодом стали застосовуватися для інших типів. Так що в якомусь сенсі це базова архітектура для інших типів мереж.

Якщо підходити до визначення строго, то не зовсім коректно називати такий тип мережі персептроном, адже вона відрізняється від нього не тільки кількістю шарів, а й тим, що:

* Нейрони мають не ступеневу функцію активації Хевисайда, а сигмоїдальну функцію.
* Навчання проводиться не за правилом Хебба, а за допомогою зворотного поширення помилки.

Тому таку архітектуру називають або просто багатошарової мережею, або мережею прямого поширення або багатошарової мережею прямого поширення. Але в той же час можна зустріти і більш зручне назву MLP, що означає багатошаровий персептрон. Структура MLP представлена на рис. 2.1.

Перший шар містить вхідні нейрони. Існують приховані шари, які складаються з певної кількості нейронів та поєднані з вхідним шаром та шаром, представленим вихідними нейронами. Активація нейрону прихованого шару в мережі з n вхідними нейронами обчислюється наступним чином:

де – ваговий коефіцієнт з'єднання вхідного нейрона хi з нейроном прихованого шару R.

Активація нейрону вихідного шару, пов'язаного з m нейронами прихованого шару обчислюється наступним чином:

Функція активації - це спосіб нормалізації вхідних даних. Тобто, якщо на вході доволі велике за модулем число, пропустивши його через функцію активації, отримаємо вихід в потрібному діапазоні.



Рисунок 2.1 – Структура багатошарового персептрону

Функцій активації досить багато, але найбільш широко досліджені наступні [1-2]:

* Лінійна,
* Сигмоїд (Логістична)
* Гіперболічний тангенс.

Головні їх відмінності - це діапазон значень. Лінійна функція описується рівнянням f(x) = x (рис. 2.2)

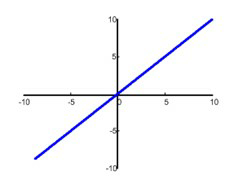


Рисунок 2.2 – Графік лінійної функції активації

Лінійна функція майже ніколи не використовується, за винятком випадків, коли потрібно протестувати нейронну мережу або передати значення без перетворень.

Сигмоїдна функція описується рівнянням [1-2]:

(2.3)

Це найпоширеніша функція активації, її діапазон значень [0,1]. Також її іноді називають логістичною функцією. Відповідно, якщо присутні від’ємні значення, то знадобитися функція яка охоплює і від’ємні значення параметрів.

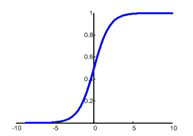


Рисунок 2.3 – Графік функції сигмоїду

Гіперболічний тангенс описується рівнянням [1]:

(2.4)

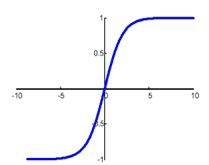


Рисунок 2.4 – Графік функції гіперболічного тангенса

Має сенс використовувати гіперболічний тангенс, тільки тоді, коли значення параметрів можуть бути і від’ємними, і додатними, так як діапазон функції [-1,1]. Використовувати цю функцію тільки з додатними значеннями недоцільно так як це значно погіршить результати роботи нейронної мережі.

Сигмоїдальна функція активації має наступні переваги в порівнянні з пороговою функцією Хевісайда [1-2]:

1. Нелінійність. Забезпечує моделі можливість обробляти (сприймати) нелінійні закономірності в даних.
2. Фіксовані обмеження виходу. Це особливість дозволяє масштабувати дані від шару до шару.
3. Безперервність і диференційованість. Дані властивості дозволяють обійти головний недолік порогової функції - різкий перехід. Оскільки навчання мережі відбувається за допомогою градієнтного спуску, то рух за пороговою функцією занадто різкий, на відміну від сигмоїдальної функції - рух за градієнтом плавний і, як наслідок, перебування мінімуму помилки стає більш стабільним і ймовірним.

Крім того, що нелінійна функція активації дозволяє моделі обробляти нелінійні закономірності в даних, вона виконує і іншу важливу задачу. Нелінійні виходи одного шару нейронів дозволяють підключити до нього другий, третій і т. д. Якщо зробити багато шарів в персептроні з лінійною функцією активації або з пороговою, то можна звести всі ці шари всього лише до одного шару.

Подібно регресійний моделям навчання моделі ШНМ мають на увазі правку тільки вагових коефіцієнтів. Ніякі інші процеси або зміни в моделі не відбуваються.

Одна з базових здібностей людей і тварин полягає в розпізнаванні візуальних образів під будь-яким кутом і в будь-якому положенні. Звичайно, за умови того, що витриманий певний поріг зашумлености, освітленості і т.д. Системи з розпізнавання тексту, на жаль, не володіють такими можливостями. Або вони навчені розпізнавати певний тип шрифту, або вони дуже чутливі до зрушень та нахилів, або до масштабу, або взагалі до всього. І незважаючи на те, що нейронні мережі в якості прототипу використовували принципи роботи мозку, довгий час не було вирішення цієї проблеми. Даний недолік називається проблемою чутливості до просторових спотворень. А домогтися потрібно так званої інваріативності до таких спотворень трьох основних типів: зсувам, поворотам, масштабуванню.

Нейрони тут також упаковані не тільки в шари, але і двовимірні карти. Інформація циркулює зліва направо по нейронам такого ж типу (як і в MLP). Але головна особливість полягає в тому, що мережа не повнозв’язна, тобто кожен нейрон має свою невелику область видимості. Таке локальне сприйняття і узагальнення від шару до шару і дає вирішення проблеми чутливості до просторових спотворень, про які говорилося вище. Іншими словами, Згорткова Нейронна Мережа (ЗНМ) здатна обробляти просторову топологію.

Саме ця архітектура ШНМ лягла в основу так званих Глибоких мереж, що породило поняття глибокого навчання (Deep Learning) [5,8]. Безумовно, даний метод не позбавлений недоліків, але у багатьох задачах саме візуального розпізнавання ЗНМ є провідним рішенням.

Але крім цього важливу роль відіграє різноманітне комбінування певних блоків карт. У глибоких мережах дуже багато різних модулів і відгалужень (може бути і таке, що мережа має кілька входів на різних рівнях).

Однак нарощування шарів призводить до цілого ряду труднощів, як з об'ємом обчислень, так і з навчанням мережі, з чого складаються особливості, притаманні саме глибокому навчанню.

Тип нейронних мереж, названий картами, розроблявся як рішення проблеми «пластичності-стабільності». Як відомо, доросла людина може зрозуміти будь-яку інформацію з одного чи кількох пояснень. Мова, звичайно, не йде про складні речі на зразок курсу математичного аналізу, але, тим не менш, мозок людини здатний зрозуміти якісь концепції навіть за одне пояснення. Якщо переводити це на мову машинного навчання, то це означає, що нам не потрібно багаторазове пред'явлення подібних прикладів, нам вистачить одного-двох об'єктів навчальної вибірки. Звичайно, мозок здатний на такі речі не тільки завдяки швидкості навчання нейронів, а тому, що сприймаючи чиєсь пояснення мозок вже має уявлення про багато речей. Іншими словами, мозок вже навчений, у нього є попередній досвід. Крім того, розум здатний тримати контекст. Мозок донавчається, а не перенавчається в таких випадках.

Перше завдання розробників мереж ART полягало в тому, щоб зробити процес навчання швидким, щоб не проходити ітеративний градієнтний спуск для редагування ваг. Ця проблема і є проблема пластичності ШНМ.

Але якщо людина дізнається, що по якомусь питанню у неї протягом деякого часу була неправильна точка зору, то вона досить просто скоректує окремі уявлення про світ. Якщо не розглядати такі ситуації, то можна прийняти той факт, що мозок дуже пластичний, тобто може навчитися або довчитися малої кількості раз і в той же час стабільний - одні знання можуть бути замінені на інші без руйнування іншої інформації.

Водночас інформація в класичних ШНМ (в MLP) розподіляється за вагами всієї мережі. Іншими словами, немає одного чіткого місця, яке б відповідало за певний блок інформації. Таким чином, якщо донавчання відбувається на об'єктах тих же класів, які використовувалися під час навчання, то багатошарову мережу прямого поширення можна довчити (але знову ж повільно). Але якщо взяти навчену мережу на одних класах і спробувати донавчити на нових, то вся стара інформація почне руйнуватися.

Наприклад, ШНМ типу MLP була навчена для класифікації квадратів і кіл по візуальним образам. Припустимо, що мережа стала показувати хороші показники якості після пред'явлення різних образів квадратів і кіл в кількості 10 тис. штук. Якщо пізніше довчити мережу на нових варіантах квадратів і кіл, то мережа може ще поліпшити показники. Але якщо почати навчати її ще й на трикутниках, то під цей новий клас не виділено певне місце в пам'яті мережі, а розподілена по вагах інформація, навпаки, почне змінюватися, руйнуючи навчений стан.

Це і є «проблемою стабільності ШНМ». Вона була другим завданням розробників мереж типу ARTMAP (творчі карти, або просто Карти).

Перші варіанти Карт (ARTMAP 1, ARTMAP 2) були дуже складними, надлишковими і багато в чому були інженерним творінням у вигляді електричних схем, а не алгоритмів.

Після повернення інтересу до нейронних мереж цей тип мереж був істотно оптимізований і модифікований. Один з найуспішніших і ефективних варіантів є Simplified FUZZY ARTMAP (спрощена нечітка Карта). Під нечіткістю треба розуміти те, що обчислення засновані на нечіткій логіці.

Власне, правильніше буде назвати це системою, що містить нейрони, так як в структурі є безліч сторонніх блоків (вузлів), які не вписуються в біологічну концепцію мозку (хоча і не суперечать їй).

У самому низу рис. 2.5 маємо вектор ознак «a» (як і у всіх інших методах ML). Даний вектор обов'язково повинен бути нормалізований (відмасштабований), тобто мережа не просто дуже чутлива до ненормалізованих даних, а взагалі не працює без нормалізації. Відповідно якщо природа якихось ознак невідома і не ясно, який діапазон може бути в даних, то застосування цієї мережі ускладнюється (потрібно спеціальним чином контролювати дані на вхід) або зовсім скасовується.

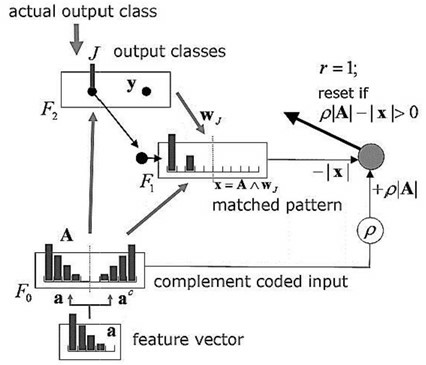


Рисунок 2.5 – Структура класу нейронних мереж ARTMAP

Далі нормалізований вектор a доповнюється комплементарною (зворотною) парою ac: там, де в початковому векторі було значення 0, зворотному буде 1, і навпаки (за умови нормалізації від 0 до 1). Для цього і була необхідна нормалізація. Далі, отриманий вектор порівнюється з усіма шаблонами на базі нейронної мережі. У даному випадку це саме база, а не просто розподілена інформація. Один нейрон в такій архітектурі є ще однією одиницею інформації, один конкретний шаблон є образом. Тобто один нейрон висловлює якийсь образ навчальної вибірки (наприклад, квадрат або коло). При навчанні кожен вектор порівнюється з уже наявною базою образів, які збережені в нейронах мережі.

Якщо близьких образів немає, то образ зберігається як новий, тобто створюється новий нейрон і вектор вхідного образу повністю копіюється в ваги нового нейрона. Так з одного разу відбувається повне коректне запам'ятовування. При цьому за нейроном закріплюється мітка y (тобто це як і раніше навчання з учителем, і для кожного вхідного вектора є відповідь).

З точки зору гіпотези компактності такий підхід дозволяє вирішувати задачу узагальнення. Гіпотеза компактності говорить про те, що об'єкти одного класу мають близькі за значенням ознаки (або будь-яку комбінацію ознак), через що їх взагалі можна відокремити від об'єктів іншого класу. Можна сказати, що об'єкти подібних класів розташовуються в деякому N-вимірному просторі ознак також близько. Об'єкти тому і належать якомусь одному змістовному класу, тому що за якоюсь ознакою чи групою (комбінацією) схожі.

Отже, дана мережа здатна зберігати образ з одного пред'явлення, що вирішує проблему пластичності. У мережі відразу з'являється образ з міткою класу, до якого належить цей образ. Таким чином, замість мінімізації помилки дана мережа навчається шляхом створення схожих прототипів і порівнянням вхідного образу з прототипами. У робочому режимі мережа видає відповідь Y у вигляді мітки того нейрона, образ якого найсильніше схожий на поточний вхідний образ.

При цьому дана мережа також вирішує завдання стабільності, так як тепер є чіткі сховища інформації і система організації нових нейронів. Так що навіть при пред'явленні нових класів мережа буде стабільно працювати. Даний тип мережі добре зарекомендував себе в медицині і при розпізнаванні звукових сигналів. Однак є і ряд мінусів даної мережі:

* при рівній мірі узагальнення мережа ARTMAP потребує істотно більшої кількості пам'яті і обчислень, ніж MLP, якщо мова йде про вхідні вектори великої розмірності і дуже великі навчальні вибірки;
* мережа не здатна враховувати топологію простору і складність, а також абстрактні ознаки (так як шарів по суті немає).

Рекурентні нейронні мережі (Recurrent Neural Network; RNN) - вид нейронних мереж, в яких є зворотний зв'язок. Наявність зворотного зв'язку дозволяє запам'ятовувати і відтворювати цілі послідовності реакцій на один стимул. У мережах такого типу виникає ефект пам'яті і здатності сприймати не тільки статичний образ, але і динаміку образів (так як є можливість враховувати історію через зворотний зв'язок).

Довга короткострокова пам'ять (Longshort-termmemory; LSTM) - різновид архітектури рекурентних нейронних мереж (RNN, запропонована в 1997 році Сеппі Хохрайтером і Юргеном Шмідхубер. Як і більшість рекурентних нейронних мереж, LSTM-мережа є універсальною в тому сенсі, що при достатній кількості елементів мережі вона може виконати будь-яке обчислення, на яке здатний звичайний комп'ютер, для чого необхідна відповідна матриця ваг, яка може розглядатися як програма. На відміну від традиційних рекурентних нейронних мереж, LSTM-мережу добре пристосована до навчання на задачах класифікації, обробки і прогнозування часових рядів у випадках, коли важливі події розділені часовими лагами з невизначеною тривалістю і кордонами.

Відносна несприйнятливість до тривалості тимчасових розривів дає LSTM переваги по відношенню до альтернативних рекурентних нейронних мереж та інших методів навчання для послідовностей в різних сферах застосування. З досягнень LSTM-мереж можна виділити найкращі результати в розпізнаванні несегментованого злитого рукописного тексту. LSTM-мережі також використовуються в задачах розпізнавання мови. Провідні технологічні компанії, включаючи Google, Apple, Microsoft і Baidu, використовують LSTM-мережі в якості фундаментального компонента нових продуктів.

Самоорганізовані мережі являють собою загальну нейронну мережу з навчанням без учителя, що виконує завдання візуалізації і кластеризації. Є методом проектування багатовимірного простору в простір з більш низькою розмірністю (найчастіше, двовимірною), застосовується також для вирішення завдань моделювання, прогнозування та ін. Є однією з версій нейронних мереж Кохонена. Самоорганізовані карти Кохонена служать, в першу чергу, для візуалізації і первинного ( «розвідувального») аналізу даних.

Геометрична суть алгоритму така, що близькі об'єкти в багатовимірному просторі будуть розташовані близько і на площині з двома осями, де об'єкти будуть представлені точками. Власне, навчання мережі це і є процес укладання таких точок (ітеративного відтягування таких точок від початкових позицій).

Сигнал в мережу Кохонена надходить відразу на всі нейрони, ваги відповідних синапсів інтерпретуються як координати положення вузла, і вихідний сигнал формується за принципом «переможець забирає все», тобто ненульовий вихідний сигнал має нейрон, найближчий (у сенсі ваг синапсів) до поданого на вхід об'єкту. В процесі навчання ваги синапсів налаштовуються таким чином, щоб вузли решітки «розташовувалися» в місцях локальних згущень даних, тобто описували кластерну структуру хмари даних, з іншого боку, зв'язки між нейронами відповідають відносинам сусідства між відповідними кластерами в просторі ознак.

Перша наукова модель імпульсної нейронної мережі була запропонована Аланом Ходжкіном і Ендрю Хакслі в 1952 році. Ця модель описувала, як потенціали дії виникають і поширюються. Імпульси, як правило, не передаються безпосередньо між нейронами. Зв'язок вимагає обміну хімічними речовинами, які називаються нейротрансміттерами. З точки зору теорії інформації, проблема полягає у відсутності моделі, яка б пояснювала, як кодується інформація і декодуються серії послідовностей імпульсів, тобто потенціали дії. За допомогою часового кодування один імпульсний нейрон може замінювати сотні прихованих елементів частотної нейронної мережі.

Що ж стосується відмінності згорткових мереж від класичних, то тут необхідно зазначити наступне. Якщо основна відмінність згорткових мереж від класичних полягає в структурі і організації зв'язків, то імпульсні мережі, навпаки, за структурою схожі на MLP. Їх головна відмінність полягає в тому, що нейрони обмінюються короткими (у біологічних нейронів - близько 1-2 мс) імпульсами однакової амплітуди (у біологічних нейронів - близько 100 мВ). Є найбільш реалістичною, з точки зору фізіології, моделлю ШНМ.

В якості основних її переваг можна назвати:

1. Перспективність в обробці динамічної інформації (наприклад, відеопотоку, часових рядів і тощо).
2. Більш щільне і ефективне кодування інформації в порівнянні з MLP.
3. Більш ефективне витрачання електроенергії при реалізації.
4. Високий ступінь ефективності паралельного виконання при фізичній реалізації.

До недоліків можна віднести слабку вивченість, складність математичної моделі і обчислювальну складність при виконанні на звичайному апаратному забезпеченні. А головний недолік полягає в відсутності якісного алгоритму навчання.

## 2.2 Програмні пакети для реалізації алгоритмів машинного навчання та аналізу даних

Протягом останнього часу автоматизація машинного навчання стала однією з найбільш швидкозростаючих областей теоретичних і практичних розробок. Існує три головні процеси, автоматизація яких вивільняє велику кількість часових ресурсів:

* підбір гіперпараметрів моделей;
* випробування великої кількості різних моделей;
* використання різних ознак, виділених з даних.

Таким чином, автоматизоване машинне навчання можна охарактеризувати як набір технологій і методів алгоритмічного вибору, оцінки ефективності моделей машинного навчання та ітеративного моделювання.

Складність традиційного підходу до побудови систем машинного навчання полягає в необхідності знання всіх існуючих алгоритмів штучного інтелекту, вміння їх правильно застосувати і налаштувати.

Історично першим механізмом автоматизації процесу машинного навчання є заснований на байєсівській оптимізації метод AutoWEKA [10].

Пропоновані методи автоматизації машинного навчання пов'язані з існуючими інструментальними засобами моделювання. Більшість з них орієнтується на популярну бібліотеку scikit-learn мови Python. Наприклад, система Auto-Sklearn, розвиваюча ідеї байєсівської оптимізації [11].

Однак байєсівська оптимізація не єдина методологічна основа автоматизації машинного навчання. Існують рішення, засновані на генетичному підході. Такі системи здатні будувати складні процеси машинного навчання без будь-якого втручання людини в процес проектування.

Активним напрямком досліджень в області автоматизації машинного навчання є опрацювання питання використання складних нелінійних конвеєрів обробки даних. Переважним методом знаходження таких конвеєрів є генетичне програмування. В середині минулого року дослідний відділ Google представив архітектуру AutoML, засновану на навчанні з підкріпленням. Ця система будує рекурентні мережі, схожі за своєю архітектурою з побудованими людиною, але більш складні.

TensorFlow - це потужна бібліотека від Google для виконання великомасштабних чисельних обчислень. Одним із завдань, яку вона виконує, є реалізація та навчання глибоких нейронних мереж. TensorFlow надає примітиви для визначення функцій на тензори для автоматичного обчислення їх похідних [9]. TensorFlow він швидко став одним з найпопулярніших проектів машинного навчання з відкритим вихідним кодом на GitHub.

У TensorFlow обчислення не виконуються відразу, замість цього створюються графи, що описують обчислення. Вузли графа представляють собою математичні операції, в той час як ребра графа представляють багатовимірні масиви даних (тензори), з'єднані цими вузлами. TensorFlow можна розглядати як гнучку модель програмування на основі потоку даних для машинного навчання. Потім графи можуть бути розподілені для виконання на кількох пристроях [9].

Дана бібліотека містить наступні переваги:

1. TensorFlow повністю безкоштовний і має відкритий вихідний код. Підтримується компанією Google.

2. Фреймворк досить простий у використанні і підтримує більшість практичних завдань.

3. Гнучка архітектура і портативність дозволяють розгортати обчислення на одному або декількох процесорах, сервері або навіть в складі продукту на смартфоні без переписування коду за допомогою єдиного API.

4. Завдяки можливості роботи на CPU і GPU він може бути впроваджений в широкому спектрі продуктів, наприклад, з розпізнаванням мови або образів.

5. TensorFlow можна використовувати для навчання і обслуговування моделей в режимі реального часу.

6. TensorFlow дозволяє максимально використовувати доступне обладнання, оскільки має розширену підтримку потоків, асинхронних обчислень і черг.

Існують і інші доступні бібліотеки машинного навчання серед яких слід зазначити:

* Theano
* Torch
* Caffe
* MXNet
* Neon
* CNTK

Бібліотеки глибокого навчання розрізняються за якістю, кількістю навчальних посібників і матеріалів для початку роботи. Theano, TensorFlow, Torch і MXNet мають хороші документації, які легко зрозуміти і реалізувати.

Згорткові нейронні мережі (CNN) використовуються для розпізнавання зображень, рекомендаційних механізмів і обробки природної мови. CNN складаються з набору різних верств, які перетворять вихідний обсяг даних в підсумкові оцінки визначених класів. Слід розглянути можливість моделювання технологій CNN для деяких функцій. Ці функції включають в себе простір можливостей для визначення моделей, наявності готових шарів і інструментів, доступних для підключення цих шарів. Існують можливості моделювання CNN є у Torch і MXNet. Проте, легка здатність TensorFlow спиратися на модель Inception v3 робить цю технологію кращої для можливостей моделювання CNN [11].

Щоб створювати і навчати нові моделі в певній бібліотеці, важливо мати простий у використанні модульний інтерфейс. TensorFlow, Torch і MXNet мають просту і модульну архітектуру, яка значно спрощує розробку. Для порівняння, такі фреймворки, як Caffe, вимагають значного обсягу роботи для створення нових шарів. TensorFlow, зокрема, легко налагоджувати і відстежувати під час і після навчання, оскільки в TensorFlow для візуалізації процесу включений графічний додаток TensorBoard.

Torch і Neon мають кращу документально підтверджену продуктивність для згортальних нейронних мереж [10]. Для більшості тестів швидкість TensorFlow [11, 12] також була кращою, в той час як Caffe і Theano були показали дещо гірший результат.

## 2.3 Характеристика сфери застосування та перспектив розвитку алгоритмів машинного навчання

Серед задач, вирішуваних на основі методів машинного навчання, мають місце такі області:

* розпізнавання образів;
* розпізнавання мови;
* розпізнавання рукописного введення;
* розпізнавання жестів;
* діагностика (медична, технічна);
* біоінформатика;
* хемоінформатика;
* прогнозування часових рядів;
* виявлення спаму;
* біржовий технічний аналіз;
* ранжирування результатів пошуку;
* створення рекомендаційних систем.

Одним з найбільш вагомих результатів використання методів машинного навчання для опису завдань обробки даних є апарат імовірнісних графічних моделей [5]. Графічні моделі дозволили радикально переглянути області застосування методів машинного навчання і аналізу даних за рахунок відмови від вимоги незалежності прихованих змінних для різних об'єктів. Дискримінативна модель вибірки об'єктів задається спільним розподілом, який, на відміну від класичного випадку, не факторизується за окремими об'єктами:

p(, | ) = p( | , ) \* p() (2.1)

Прикладами областей, в яких використовується даний підхід, заснований на відмові від припущення про незалежність прихованих змінних, є:

* Соціальні мережі. Користувачі соціальних мереж характеризуються, як спостережувані змінні (наприклад, анкетна інформація, яку користувач повідомив про себе в мережі), так і прихованими змінними (наприклад, його реальні інтереси, схильність до позитивної реакції на адресну рекламу тощо). Хоча є можливість формального аналізу кожного користувача незалежно, видається досить очевидним, що інформація про значення прихованих змінних його друзів, може значно розширити уявлення про даного користувача.
* Комп'ютерний зір. У задачі семантичної сегментації зображень, що є першим етапом будь системи комп'ютерного зору, потрібно зіставити кожному пікселю деяку мітку класу, яка відповідає предметові, в зображення якого входить даний піксель. Очевидно, що крім інформації про даний піксель (колір, значення дескрипторів, інтенсивність та ін.) або інші пікселі, важливу роль відіграють мітки сусідніх пікселів, тому що неявно передбачається, що сусідні пікселі частіше всього мають однакові мітки.
* Імітаційне моделювання. При моделюванні середовищ взаємодіючих агентів (наприклад, транспортних потоків в містах) стан кожного агента залежить від станів інших агентів, що знаходяться в межах зони взаємодії. Стан кожного агента можна розглядати як приховану змінну об’єкта, що залежить від прихованих змінних інших об'єктів. Дослідження таких взаємодій грає важливу роль, тому що дозволяє встановити умови стрибкоподібних переходів від локальних взаємодій до глобальних (фазові переходи), наприклад, коли через різке короткочасне гальмування однієї машини в потоці виникає багатокілометрова пробка.
* Коллаборативна фільтрація (collaborative filtering). З розвитком інтернеткомерції все більшу актуальність отримують рекомендаційні сервіси. В ситуації, коли відвідувач фізично не може переглянути весь асортимент інтернет-магазину, що включає в себе десятки тисяч найменувань, виникає задача формування обмеженого списку товарів, які його потенційно можуть зацікавити. Крім спостережуваних змінних об'єкта, яким виступає клієнт, що характеризують його соціально-демографічний профіль і історію покупок, необхідно аналізувати покупки інших клієнтів і близькість їх переваг до переваг розглянутого клієнта.

Характерне число об'єктів у вибірці, з яким доводиться стикатися в сучасних задачах, становить величину порядку десятків тисяч - мільйонів. Основна складність, яка виникає при спробі побудувати ймовірнісну модель, яка містить взаємозалежності між прихованими змінними об'єктів, полягає в неможливості задати такий розподіл в загальному вигляді. Нехай є тисяча об'єктів, у кожного з яких є одна прихована змінна, що приймає два значення. Для того, щоб задати p( | , ) знадобилося б задати 21000 ≈ 10300 значень ймовірностей. Така кількість на багато порядків перевершує обсяги доступної пам'яті будь-якого сховища даних. При використанні графічних моделей передбачається, що спільний розподіл може бути представлено у вигляді добутку факторів, кожен з яких залежить від невеликої підмножини об'єктів, причому підмножини перетинаються. Завдяки цьому вдається змоделювати ситуації, коли прихована компонента довільного об'єкта залежить від прихованої компоненти кожного з решти об'єктів вибірки. З іншого боку, за рахунок факторизації, можна зменшити вимоги до пам'яті аж до лінійних за кількістю об'єктів, що дозволяє зберігати спільні розподіли на сотні тисяч об'єктів.

Ключовим поняттям, необхідним для розуміння логіки роботи апарату графічних моделей, є поняття умовної незалежності випадкових величин. Випадкові величини *a* і *b* називаються незалежними за умови *c*, якщо виконується рівняння:

*p(a, b | c) = p(a | c) \* p(b | c).* (2.2)

Найпростішим прикладом умовно незалежних величин є: зріст людини (величина *a*), довжина його волосся (величина *b*) і його стать (величина *c*). Добре відомо, що зріст зворотно корелює з довжиною волосся, однак, після додавання в ймовірнісну модель фактора статі людини, зріст і довжина волосся стають незалежними величинами.

Існує два основних правила роботи з випадковими величинами. Розглянемо спільну щільність *n* випадкових величин *p(a1,..., an)*. Правило добутку говорить про те, що будь-яку багатовимірну щільність можна представити у вигляді добутку одновимірних умовних щільностей:

*p(a1,..., an) = p(an | a1,..., an-1) × p(an-1 | a1,..., an-2). . . p(a2 | a1) p(a1).* (2.3)

Аналогічні рівняння можна виписати для довільного перевпорядкованих змінних. Правило підсумовування дозволяє отримувати безумовні розподіли меншої розмірності шляхом виключення частини змінних:

(2.4)

Всі операції, які здійснюються з ймовірнісними моделями при використанні байєсівського формалізму, спираються на застосування цих двох правил. Байєсівські мережі дозволяють моделювати причинно-наслідкові зв'язки між величинами. Для цього на множині змінних Y = (, , ) ймовірнісної моделі задається орієнтований граф, в якому ребра відображають відносини причинності. За змістом побудови в такому графі заборонені орієнтовані цикли. Граф причинності задає систему факторизації спільного розподілу

(2.5)

де *pa*i - множина батьків *i*-ї вершини.

Слід зазначити, що розмір кожного фактора (а саме розмірність чинників служить мірою складності розподілу як на етапі його завдання, так і на етапі роботи з ним) визначається числом батьків вершини. Така система факторизації значно спрощує розрахунки довільних умовних і маргінальних розподілів, а саме до цього, зводяться завдання настройки і виведення в байєсівських моделях. Часто виникає необхідність моделювати системи випадкових величин між якими є залежності, але некоректно говорити про причинно-наслідкові зв'язки. Прикладом таких величин можуть бути мітки сусідніх пікселів в завданні сегментації зображень або профілі друзів у соціальній мережі. Для моделювання таких залежностей на множині величин задається неорієнтований граф, який визначає факторизацію спільного розподілу таким чином:

(2.6)

де *ψc*() - невід'ємні функції.

Слід зазначити, що на відміну від байєсівських мереж, фактори не мають ймовірнісного сенсу, тому необхідна додаткове нормування добутку чинників. Якщо величини *y’* та *у’’* ніколи не входять в один фактор (тобто не з'єднані ребром у графі), то вони є незалежними за умови, що всі інші величини відомі. Таким чином, ребра графа визначають відносини умовної незалежності. Однією з переваг систем факторизації, які задаються графічними моделями, нарівні з зручністю представлення багатовимірних розподілів, є можливість паралельної і розподіленої обробки інформації при підрахунку умовних розподілів, наприклад, за допомогою інтерфейсу передачі повідомлень (message-passing interface). Апарат графічних моделей використовується для точного або наближеного вирішення наступних основних завдань:

- Навчання з учителем

- Навчання без учителя:

= (2.7)

- Підрахунок нормованої константи Z;

- Підрахунок найбільш ймовірної конфігурації прихованих змінних:

(2.8)

- Підрахунок маргінального розподілу фіксованої змінної *p(ti | X, )*.

Слід зазначити, що всі ці завдання зводяться до підрахунку тих чи інших умовних розподілів на невідомі змінні за умови спостережуваних змінних і, можливо, маргіналізації за нерелевантними змінними. Можна помітити, що ті ж завдання виникають в класичному машинному навчанні. Перенесення класичних результатів на більш складні графічні моделі є одним з найважливіших напрямків робіт в сучасному машинному навчанні.

# 3 ВИКОНАННЯ АЛГОРИТМУ МАШИННОГО НАВЧАННЯ ЗАСОБАМИ MATLAB

## 3.1 Моделювання нейронної мережі на основі Neural Network Toolbox

Набір даних для навчання нейронної мережі містить 9568 точок даних, зібраних від комбінованої електростанції за 6 років (2006-2011), коли електростанція була налаштована на роботу з повним навантаженням. Характеристики складаються із середньочасових змінних навколишнього середовища Температура (T), Тиск навколишнього середовища (AP), Відносна вологість (RH) та Витяжний вакуум (V) для прогнозування чистого погодинного виходу електричної енергії (ЕП) установки.

Електростанція комбінованого циклу (CCPP) складається з газових турбін (GT), парових турбін (ST) та парогенераторів, що відновлюють тепло. У ЦАЕС електроенергія виробляється газовими та паровими турбінами, які об'єднуються в один цикл і передаються від однієї турбіни до іншої. Хоча вакуум збирається і впливає на парову турбіну, інші три змінні середовища впливають на продуктивність GT.

Дані складаються із середньочасових змінних навколишнього середовища

Вхідні параметри:

* температура (Т) в межах 1,81 ° С і 37,11 ° С,
* тиск навколишнього середовища (AP) в діапазоні 992,89-1033,30 мілібар,
* відносна вологість (RH) в межах від 25,56% до 100,16%
* витяжний тиск (V) в діапазоні 25,36-81,56 см рт

Вихідний параметр:

* чиста погодинна потужність електричної енергії (ЕП) 420,26-495,76 МВт

Середні показники зазначених параметрів взяті від різних датчиків, розташованих навколо установки, та щосекунди записують змінні навколишнього середовища. Змінні представлені без нормалізації (рис. 3.1).

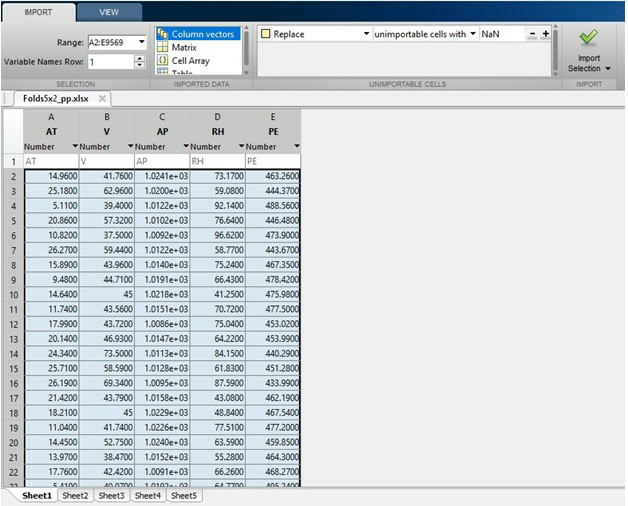


Рисунок 3.1 – Навчальні дані

Виклик вікна для початку роботи з пакетом для навчання нейронних мереж здійснено за допомогою команди nnstart (рис. 3.2) у середовищі Matlab 2014.

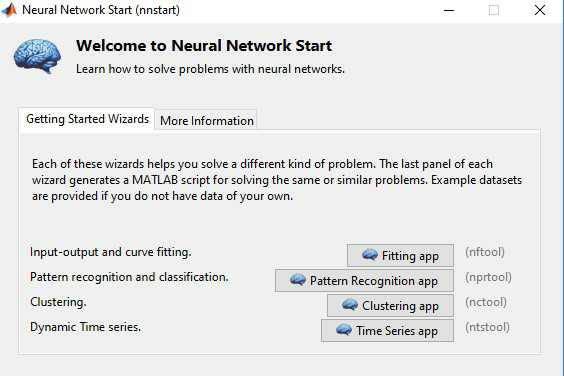


Рисунок 3.2 – Вікно початку роботи з пакетом для навчання нейронних мереж

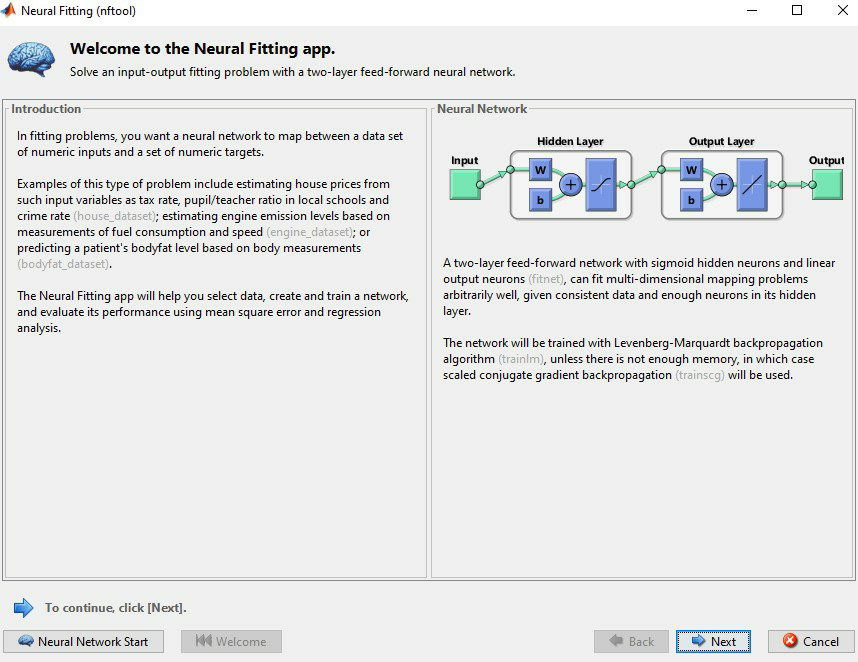


Рисунок 3.3 – Опис налаштувань пакету машинного навчання

Задано вибірки вхідних та вихідних даних, як показано на рис. 3.4.

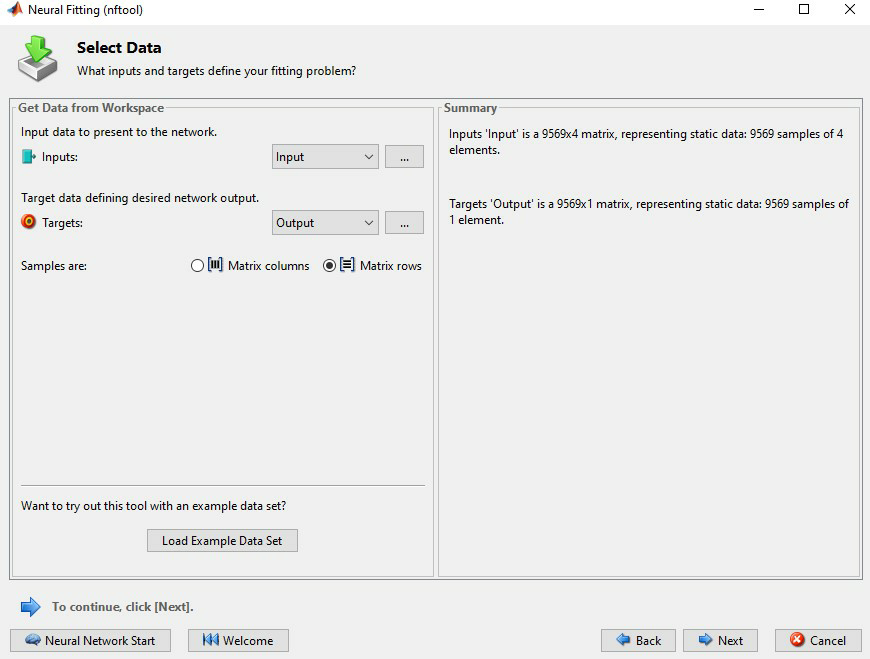


Рисунок 3.4 – Завдання вхідних та вихідних даних

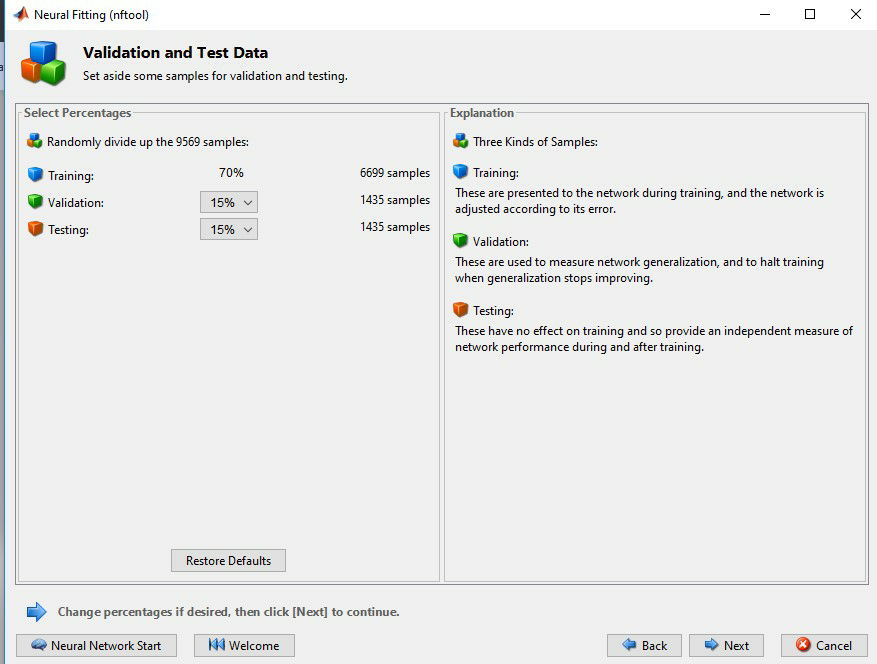


Рисунок 3.5 – Завдання розміру навчальної, тестової та перевірочної вибірки

Як показано на рис. 3.5, навчальна вибірка задана у розмірі 70% даних, тестова та перевірочна у обсязі 15% кожна. Наступним кроком задано кількість прихованих нейронів – 10 (рис. 3.6).

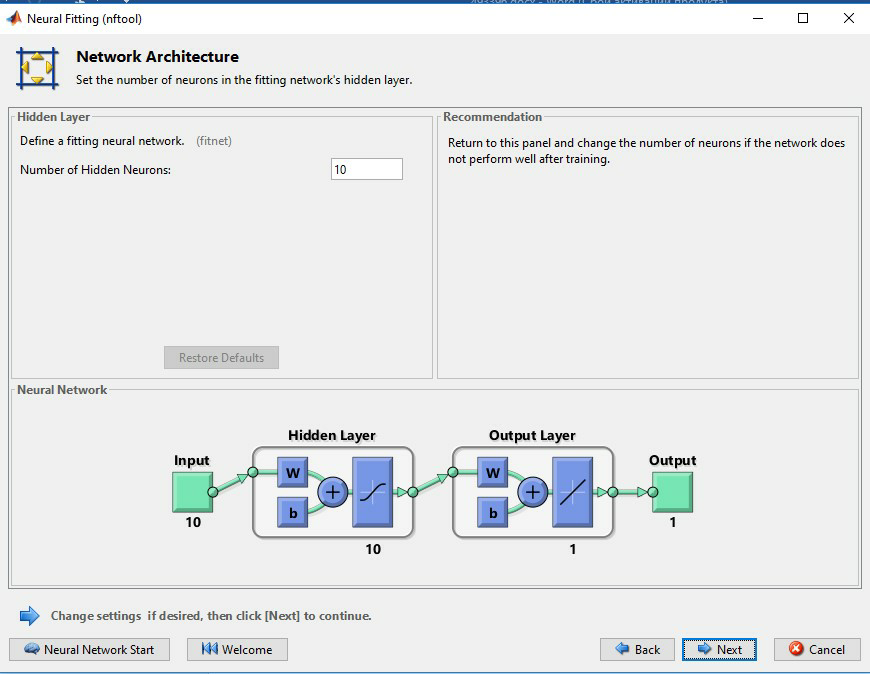


Рисунок 3.6 – Завдання кількості прихованих нейронів

Наступним кроком обрано алгоритм оптимізації для навчання нейронної мережі – алгоритм Левенберга-Марквардта (рис. 3.7).

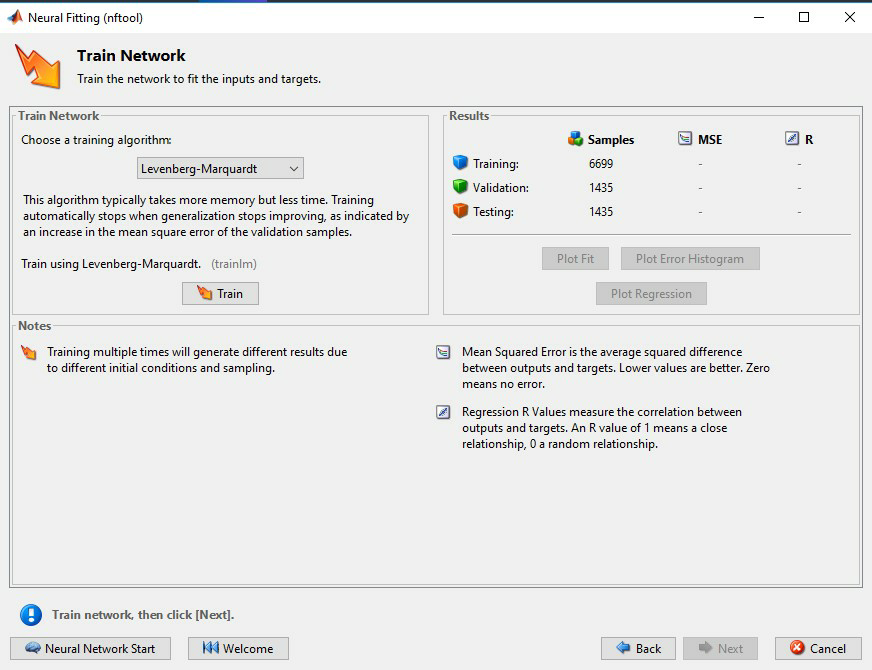


Рисунок 3.7 – Завдання алгоритму алгоритм Левенберга-Марквардта

## 3.2 Аналіз отриманих результатів

Результати навчання мережі представлені на рис. 3.8 – 3.13 для навчальної, тестової та перевірочної вибірки розраховано регресійну помилку між вихідним параметром мережі та цільовим значенням, яка складає 0,96-0,97 та середньоквадратичну помилку, яка в абсолютному своєму значенні складає 15,78 як найменше значення для навчальної вибірки та 18,23 – як найбільше значення для перевірочної вибірки. У відносних значеннях помилка складає 3-3,6%.

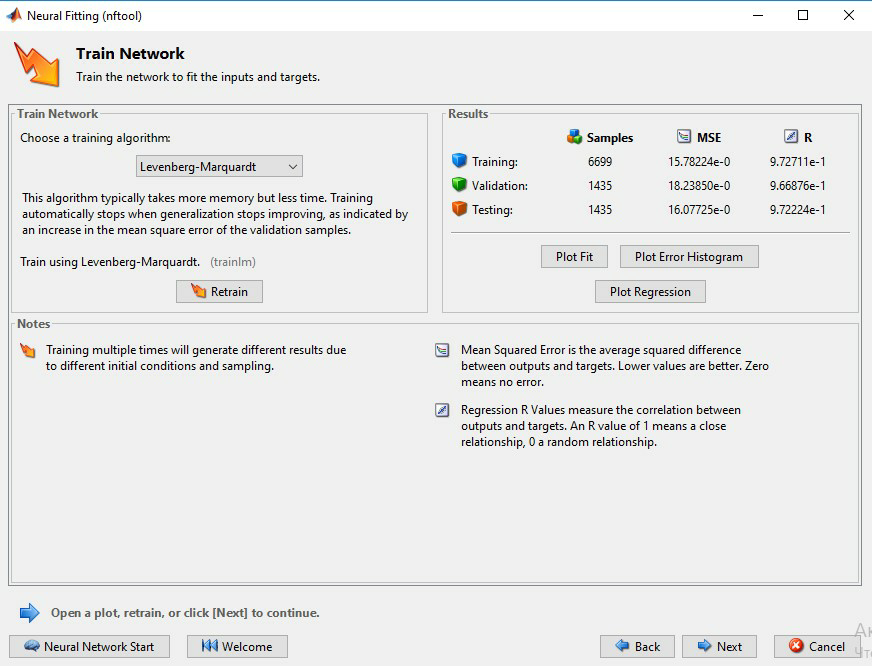


Рисунок 3.8 – Результати навчання нейронної мережі

На рис. 3.9 зображено параметри навчання нейронної мережі: кількість епох навчання, час навчання тощо.

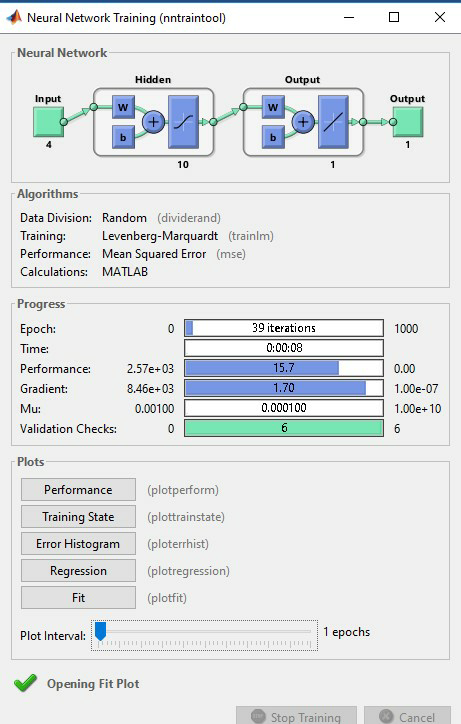


Рисунок 3.9 – Ітогові параметри навчання нейронної мережі

Як видно з рис. 3.10, що демонструє середньоквадратичну помилку на всіх вибірках для кожної епохи навчання, найкращий результат отримано для 33 епохи.

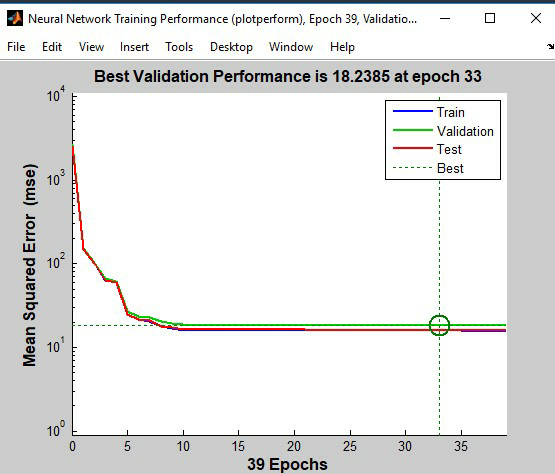


Рисунок 3.10 – Графік середньоквадратичної помилки за епохами навчання

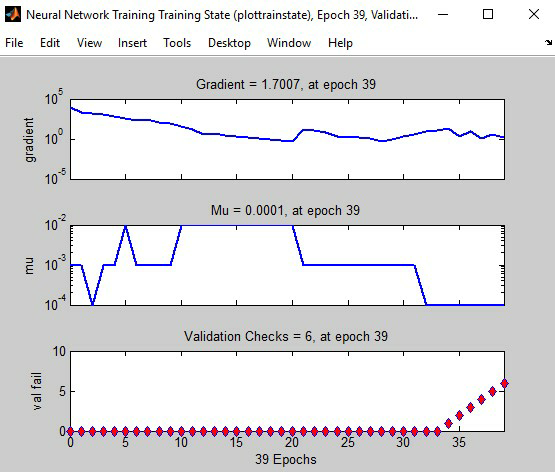


Рисунок 3.11 – Графіки параметрів навчання за епохами

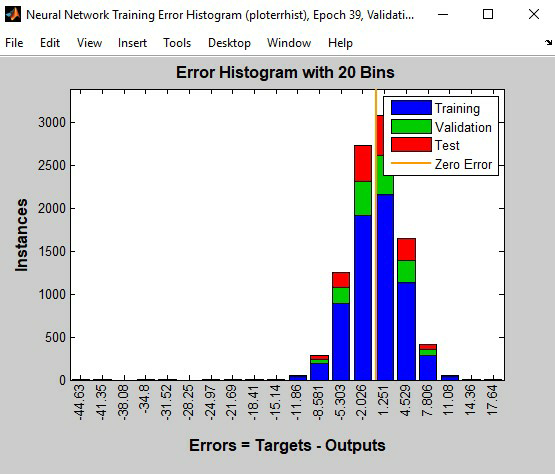


Рисунок 3.12 – Гістограма помилки навчання

Як видно з рис. 3.12, графік помилки навчання, що представлена різницею між цільовим значення та результатом роботи нейронної мережі, близький до нормального розподілу. Залежність між цільовим та отриманим результатом наведено на рис. 3.13, з якого видно, що для всіх вибірок результати схожі, але кращий результат отримано для навчальної вибірки даних.

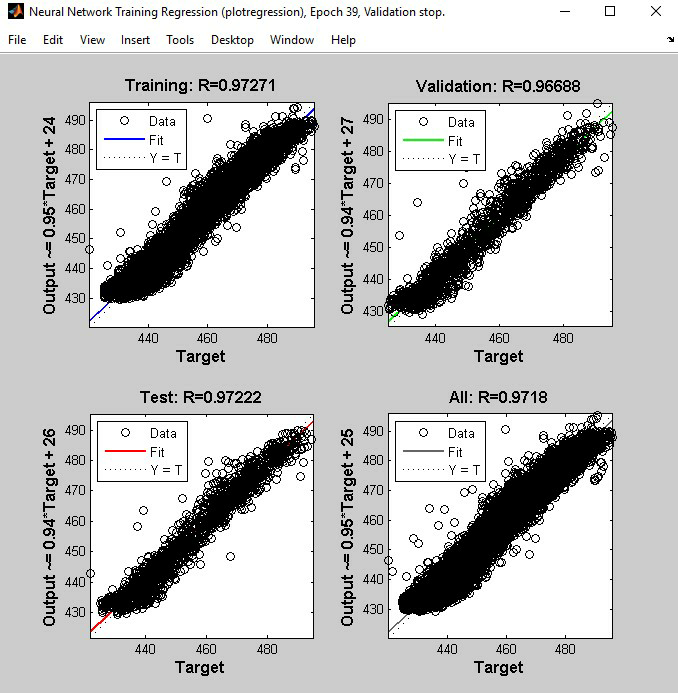


Рисунок 3.13 – Графіки залежності між цільовим та отриманим значенням

**Висновок**

В даний час спостерігається прогрес у розвитку методик машинного навчання шляхом побудови ефективних навчальних моделей аналізу даних, які можна застосувати до багатьох практичних завдань інтелектуального аналізу даних.

В роботі виконано огляд актуальних прийомів, технологій і методик, що застосовуються при вирішенні прикладних задач машинного навчання. Було взято вибірку даних з офіційного джерела, який містить різні вибірки для навчання нейромереж, після чого проведено навчання мережі в пакеті Matlab та виконано наступні задачі:

* розкрито фундаментальні аспекти теорії машинного навчання;
* проаналізовано математичне забезпечення методів штучного інтелекту;
* розглянуто основні задачі машинного навчання;
* виконано огляд програмних пакетів для реалізації алгоритмів машинного навчання та аналізу даних;
* надано характеристику сфери застосування та перспектив розвитку методів машинного навчання;
* продемонстровано на прикладі машинне навчання засобами Neural Network Toolbox в середовищі Matlab.

В ході огляду сучасних тенденцій в машинному навчанні виділено такі перспективні напрямки фундаментальних і прикладних досліджень в даній області:

1. Теоретичні дослідження в області моделей штучного інтелекту в поєднанні з аналізом автоматично побудованих моделей.
2. Практичні дослідження мультизадачних, генеративних (породжуючих) моделей.
3. Поширення автоматизованих засобів машинного навчання.
4. Розвиток і уніфікація інструментальних засобів, в тому числі хмарних засобів і сервісів інтелектуального аналізу даних.
5. Розробка нових інтелектуальних продуктів користувальницького рівня, заснованих на досягненнях методології машинного навчання.

Постійний розвиток методів машинного навчання зумовлений зростанням можливостей сучасних обчислювальних систем та обсягів даних, доступних для аналізу, а також постійним розширенням області застосування методів машинного навчання на все більш широкий клас задач обробки даних.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Хайкин С. Нейронные сети: полный курс. 2-е изд. М., "Вильямс", 2006.
2. Городецкий А.Е., И.Л.Тарасова. Управление и нейронные сети.– СП : Изд-во Политехн. ун-та, 2005. – 312 с.
3. Захарова Е.М., Минашина И.К. Обзор методов многомерной оптимизации. Математические модели. Вычислительные методы. 2014. Т. 14. № -3. С. 256-274.
4. Граничин О.Н., Поляк Б.Т. Рандомизированные алгоритмы оценивания и оптимизации при почти произвольных помехах. М.: Наука, 2003. - 291 с.
5. Бенджио И., Гудфеллоу Я., Курвилль А. Глубокое обучение. М.: ДМКПресс, 2018
6. Воронцов К. Математические методы обучения по прецедентам, [Електронний ресурс] // Режим доступу: http://www.machinelearning.ru/wiki/index.php?title=Машинное\_обучение\_% 28курс\_лекций%2C\_К.В.Воронцов%29
7. Deng L., Yu, D. Deep Learning: Methods and Applications // Foundations and Trends in Signal Processing, 2014. Vol. 7, No. 3–4, P. 197-387.
8. Николенко C., Кадурин А., Архангельская Е. Глубокое обучение. Погружение в мир нейронных сетей. СПб.: Питер, 2018
9. TensorFlow [Електронний ресурс] // Режим доступу: https://www.tensorflow.org/
10. convnet-benchmarks [Електронний ресурс] // Режим доступу: <https://github.com/soumith/convnet-benchmarks/blob/master/README.md>
11. tf benchmarks [Електронний ресурс] // Режим доступу: <https://github.com/tobigithub/tensorflow-deep-learning/wiki/tf-benchmarks>
12. TensorFlow GitHub [Електронний ресурс] // Режим доступу: https://github.com/tensorflow/tensorflow