Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Розробка матричного методу опорних векторів з адаптивним комбінованим навчанням активаційної функції в рамках задачі розпізнавання образів
(тема)

Виконав:
студент 2 курсу, групи СШІм-21-1
Ревека К.І.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник проф. Чалий С.Ф.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Ревеці Катерині Ігорівні
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка матричного методу опорних векторів з адаптивним комбінованим навчанням активаційної функції в рамках задачі розпізнавання образів

затверджена наказом університету від 31 березня 20 23 р. № 306Ст

2. Термін подання студентом роботи до екзаменаційної комісії 16 травня 20 23 р.

3. Вихідні дані до роботи Операційна система – Windows 10, Частота процессору 2.5 ГГц, Середовище розробки – PyCharm, Мова програмування – Python, Фреймворк машинного навчання – Tensorflow, Бібліотеки і модулі, що використовувались: numPy, scikit-learn, opencv, sys, os, keras

4. Перелік питань, що потрібно опрацювати в роботі

1) Аналіз предметної області та постановка завдання

2) Розробка матричного методу опорних векторів з адаптивним комбінованим навчанням активаційної функції

3) Опис системи і експериментальна перевірка отриманих наукових результатів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної області	05.04.2023	виконано
2	Аналіз методів та підходів до задачі класифікації образів	10.04.2023	виконано
3	Визначення основних структурних елементів системи	20.04.2023	виконано
4	Проектування структури системи	21.04.2023	виконано
5	Виділення основних параметрів навчання	25.04.2023	виконано
6	Реалізація алгоритму LSSVM	27.04.2023	виконано
7	Проведення експериментів	30.04.2023	виконано
8	Аналіз отриманих результатів	03.05.2023	виконано
9	Написання пояснювальної записки	07.05.2023	виконано
10	Попередній захист	10.05.2023	виконано
11	Захист перед ЕК	16.05.2023	виконано

Дата видачі завдання 3 квітня 2023 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Чалий С.Ф.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 80 с., 6 табл., 44 рис., 67 форм, 1 дод., 17 джерел.

АДАПТИВНА СИСТЕМА, АДАПТИВНЕ КОМБІНОВАНЕ НАВЧАННЯ, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, МАТРИЧНИЙ МЕТОД, МЕТОД ОПОРНИХ МАШИН, ФУНКЦІЇ АКТИВАЦІЇ, ШТУЧНИЙ ІНТЕЛЕКТ

Постановка задачі – матричний метод опорних векторів з адаптивним комбінованим навчанням активаційної функції для дослідження опису характеристик даних зображень з метою покращення стабільності і точності передбачень.

Об'єктом дослідження є обробка і аналіз даних, що мають суміжні описові характеристики, вивчення функцій методу опорних векторів за допомогою адаптивних методів комбінованого навчання активаційної функції.

Предметом дослідження є методи опорних векторів з адаптивним комбінованим навчанням активаційної функції.

Метою роботи є побудова моделі опорних векторів з адаптивним комбінованим навчанням активаційної функції.

Support Vector Machines — це потужна методологія для вирішення проблем нелінійної класифікації, оцінки функцій і оцінки щільності, яка також призвела до багатьох інших останніх розробок у методах на основі ядра загалом. Спочатку це було введено в контексті теорії статистичного навчання та мінімізації структурного ризику. У методах розв’язуються задачі опуклої оптимізації, як правило, квадратичні програми. Машини опорних векторів найменших квадратів (LS-SVM) — це модифікація стандартних SVM, що веде до вирішення лінійних

систем KKT. LS-SVM тісно пов’язані з мережами регуляризації та процесами Гаусса, але додатково підкреслюють і використовують первинно-дуальні інтерпретації. Доступні зв’язки між версіями ядра класичних алгоритмів розпізнавання образів, такими як дискримінантний аналіз ядра Фішера та розширення для неконтрольованого навчання, рекурентних мереж і керування.

Методи дослідження: проведення аналізу предметної області, аналіз використання методу опорних векторів для класифікації з різними функціями активацій і параметрами з подальшим проведення оцінки ефективності запропонованих підходів, порівняння результатів за різними параметрами налаштування системи.

ABSTRACT

Explanatory note: 80 pp., 6 tables, 44 figures, 67 formulas, 1 appendice, 17 sources.

ADAPTIVE SYSTEM, DATA ANALYSIS, SUPPORT MACHINES METHOD, INTELLIGENT MATRIX METHOD, ADAPTIVE COMBINED LEARNING, ACTIVATION FUNCTIONS, ARTIFICIAL INTELLIGENCE,

The problem statement is the matrix method of support vectors with adaptive combined learning of the activation function for the study of the description of the characteristics of image data in order to improve the stability and accuracy of predictions.

The object of the study is the processing and analysis of data with related descriptive characteristics, the study of functions of the method of support vectors using adaptive methods of combined learning of the activation function.

The subject of research is support vector methods with adaptive combined learning of the activation function.

The purpose of the work is to build a model of support vectors with adaptive combined learning of the activation function.

Support Vector Machines is a powerful methodology for solving nonlinear classification, feature estimation, and density estimation problems that has also led to many other recent developments in kernel-based methods in general. It was originally introduced in the context of statistical learning theory and structural risk minimization. The methods solve convex optimization problems, usually quadratic programs. Least Squares Support Vector Machines (LS-SVMs) is a modification of standard SVMs that leads to the solution of linear KKT systems. LS-SVMs are closely related to regularization networks and Gaussian processes, but additionally emphasize

and use primal-dual interpretations. Links are available between kernel versions of classic pattern recognition algorithms such as Fisher Kernel Discriminant Analysis and extensions for unsupervised learning, recurrent networks, and control.

Research methods: analysis of the subject area, analysis of the use of the support vector method for classification with various activation functions and parameters, followed by an evaluation of the effectiveness of the proposed approaches, comparison of results based on various parameters of the system setting.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

SVM – support vector machine - метод опорних векторів;

LSSVM – least squares support vector machines - це версії за методом найменших квадратів машин опорних векторів;

LSSVMa – least squares support vector machines adaptive - матричний метод опорних векторів з адаптивним комбінованим навчанням активаційної функції;

CNN – convolutional neural network – спеціальна архітектура штучних нейронних мереж;

DL – deep learning – глибоке навчання;

ВСТУП

Проблема класифікації – розпізнавання образів, в тому числі зображень, є однією з найважливіших проблем в інтелектуальному аналізі даних, інтелектуальний аналіз потоку даних і інтелектуальний аналіз великих даних. В даний час існує багато підходів для її вирішення. Як на сьогодні найбільш ефективними є глибокі нейронні мережі, демонструючи дійсно вражаючі результати саме в завдані обробки зображень. В той же самий час, ці мережі мають кілька істотних недоліків, які обмежують їх використання, особливо в завданнях, коли дані отримані для обробки послідовні, спостереження за спостереженням, і можливі в режимі реального часу. Це тому, що глибокі мережі є досить повільними системами, які навчаються з використанням помилок зворотного поширення протягом багатьох епох у пакетному режимі, який займає багато часу і потребує великих обсягів навчальних даних, які не завжди придатні в реальних програмах.

Тому актуальним і доцільним є створення системи мережевого апроксиматора, що здатна відновлювати гіперплощини, які розділяє класи зображень без перетворення їх у векторну форму за умов обмеженості навчальних наборів або нестаціонарності входу інформаційних характеристик.

Запропоновано метод опорних векторів із квадратичною функцією втрат (LS-SVM), що є модифікацією алгоритму опорних векторів (SVM) з функцією нечутливості Вапника як з лінійними, так і з нелінійними ядерними функціями. Він є одним із найперспективніших алгоритмів побудови класифікації і регресії. Одним із важливих етапів побудови класифікації з використанням методу опорних векторів є налаштування його низки внутрішніх параметрів. При використанні довільних значень параметрів алгоритму опорних векторів якість роботи алгоритму може суттєво змінюватись.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ І ПОСТАНОВКА ЗАВДАННЯ

Метод опорних векторів із квадратичною функцією втрат (LS–SVM) є модифікацією алгоритму опорних векторів (SVM). Одним із важливих етапів побудови моделі з використанням методу опорних векторів є налаштування ряду його внутрішніх параметрів. При використанні довільних значень параметрів алгоритму опорних векторів якість роботи алгоритму може істотно варіюватися.

1.1 Задача розпізнавання зображень

Розпізнавання зображень – це завдання ідентифікації цікавих об’єктів на зображенні та розпізнавання, до якої категорії належить зображення. Розпізнавання зображень, розпізнавання фотографій і розпізнавання зображень – це терміни, які використовуються як синоніми. Коли візуально людина бачить об’єкт або сцену, то автоматично ідентифікує об’єкти як різні екземпляри та пов’язуємо їх з окремими визначеннями. Однак візуальне розпізнавання є дуже складним завданням для машин, яке потребує значної потужності обробки. Розпізнавання зображень за допомогою штучного інтелекту є давньою проблемою досліджень у сфері комп’ютерного зору. Хоча різні методи імітації людського зору розвивалися з часом, спільною метою розпізнавання зображень є класифікація виявлених об’єктів у різні категорії (визначення категорії, до якої належить зображення). Тому його ще називають розпізнаванням об'єктів (Рис 1.1). За останні роки машинне навчання, зокрема технологія глибокого навчання, досягло великих успіхів у багатьох задачах комп’ютерного зору та розуміння зображень .

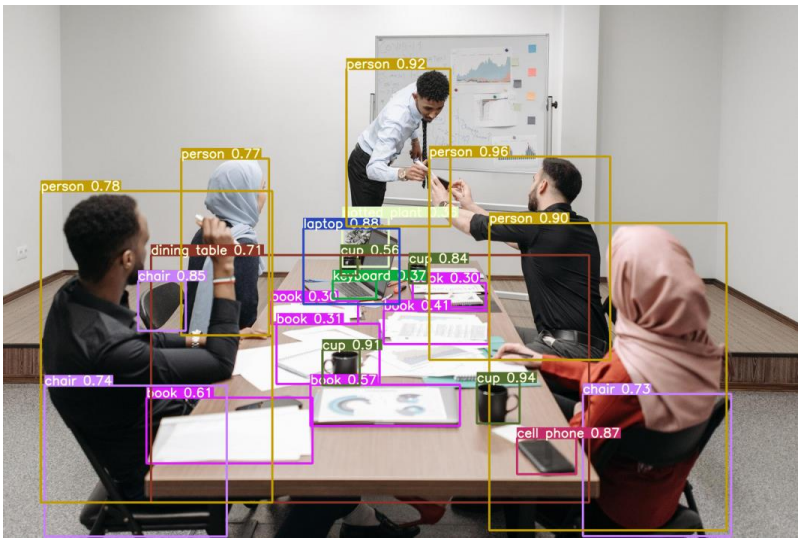


Рисунок 1.1 – Приклад розпізнавання об’єктів

У сфері комп’ютерного зору такі терміни, як сегментація, класифікація, розпізнавання та виявлення, часто використовуються як синоніми, а різні завдання збігаються. Хоча це здебільшого не проблема, все стає заплутаним, якщо ваш робочий процес вимагає від вас конкретного виконання певного завдання. Розпізнавання зображень проти комп’ютерного зору.

Задача 1: Image Detection

- Виявлення зображення або об’єкта — це комп’ютерна технологія, яка обробляє зображення та виявляє об’єкти в ньому. Дозволяэ визначити місцезнаходження об'єктів, наприклад, дізнатися кількість об'єктів на зображенн.

Задача 2: Image Classification

- Це завдання присвоєння мітки або класу всьому зображенню. Зображення повинні мати лише один клас для кожного зображення. Моделі класифікації зображень приймають зображення як вхідні дані та повертають прогноз про те, до якого класу належить зображення.

Задача 3: Image Recognition

- Це поєднання виявлення та класифікації об’єктів. Це здатність ШІ для виявлення об’єкта, класифікації та розпізнавання. Тип програмування штучного інтелекту (ШІ), який може призначати єдину мітку високого рівня зображенню шляхом аналізу та інтерпретації піксельних шаблонів зображення.

Рисунок 1.2 – Задачі обробки зображень

Терміни розпізнавання зображень і комп’ютерний зір часто використовуються як синоніми, але насправді вони різні. Насправді розпізнавання зображень – це застосування комп’ютерного зору, яке часто вимагає виконання кількох завдань комп’ютерного зору, таких як виявлення об’єктів, ідентифікація зображень і класифікація зображень. Локалізація об’єктів – це ще одна підмножина комп’ютерного зору, яку часто плутають із розпізнаванням зображень. Локалізація об’єкта означає визначення розташування одного або кількох об’єктів на зображенні та малювання обмежувальної рамки навколо їхнього периметра. Однак локалізація об'єктів не включає класифікацію виявлених об'єктів.

Терміни розпізнавання зображень і виявлення зображень часто використовуються замість один одного. Однак є важливі технічні відмінності. Виявлення зображень – це завдання взяти зображення як вхідні дані та знайти в ньому різні об’єкти. Мета виявлення зображення полягає лише в тому, щоб відрізнити один об’єкт від іншого, щоб визначити, скільки різних об’єктів присутні на зображенні.

Таким чином, навколо кожного окремого об'єкта малюються обмежувальні рамки. З іншого боку, розпізнавання зображень — це завдання ідентифікації об’єктів, що цікавлять зображення, і розпізнавання, до якої категорії чи класу вони належать. Звичайний підхід комп’ютерного зору до розпізнавання зображень — це послідовність фільтрації зображень, сегментації, виділення ознак і класифікації на основі правил. Однак традиційний підхід комп’ютерного зору вимагає високого рівня досвіду, багато часу на розробку та містить багато параметрів, які потрібно визначати вручну, тоді як переносимість на інші завдання досить обмежена. Розпізнавання зображень за допомогою машинного навчання, з іншого боку, використовує алгоритми для вивчення прихованих знань із набору даних хороших і поганих зразків. Найпопулярнішим методом машинного навчання є глибоке навчання, де в моделі використовуються кілька прихованих шарів нейронної мережі (Рис 1.2).

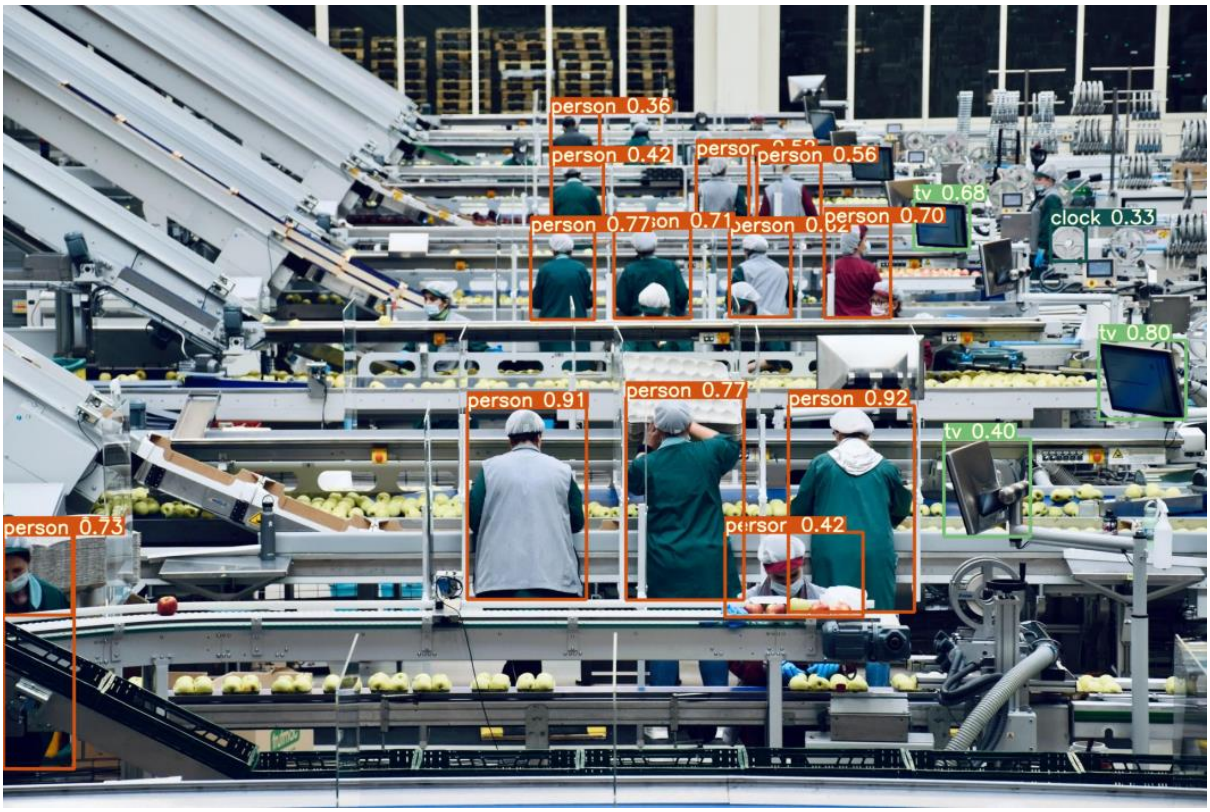


Рисунок 1.3 – Приклад розпізнавання об’єктів

Впровадження глибокого навчання в поєднанні з потужним апаратним забезпеченням штучного інтелекту та графічним процесором стало можливим для великих проривів у сфері розпізнавання зображень. Завдяки глибокому навчанню алгоритми класифікації зображень і розпізнавання облич досягають продуктивності, що перевищує людський рівень, і виявляють об’єкти в реальному часі. У 2017 році алгоритм Mask RCNN був найшвидшим детектором об’єктів у реальному часі в еталонному тесті MS COCO з часом висновку 330 мс на кадр. В якості порівняння, алгоритм YOLOR, який був випущений у 2021 році, досягає часу висновку 12 мс на тому ж тесті, навіть перевершуючи популярні алгоритми глибокого навчання YOLOv4 і YOLOv3. А в липні 2022 року алгоритм YOLOv7 навіть значно перевершив YOLOR як за швидкістю, так і за точністю (Рис 1.3).

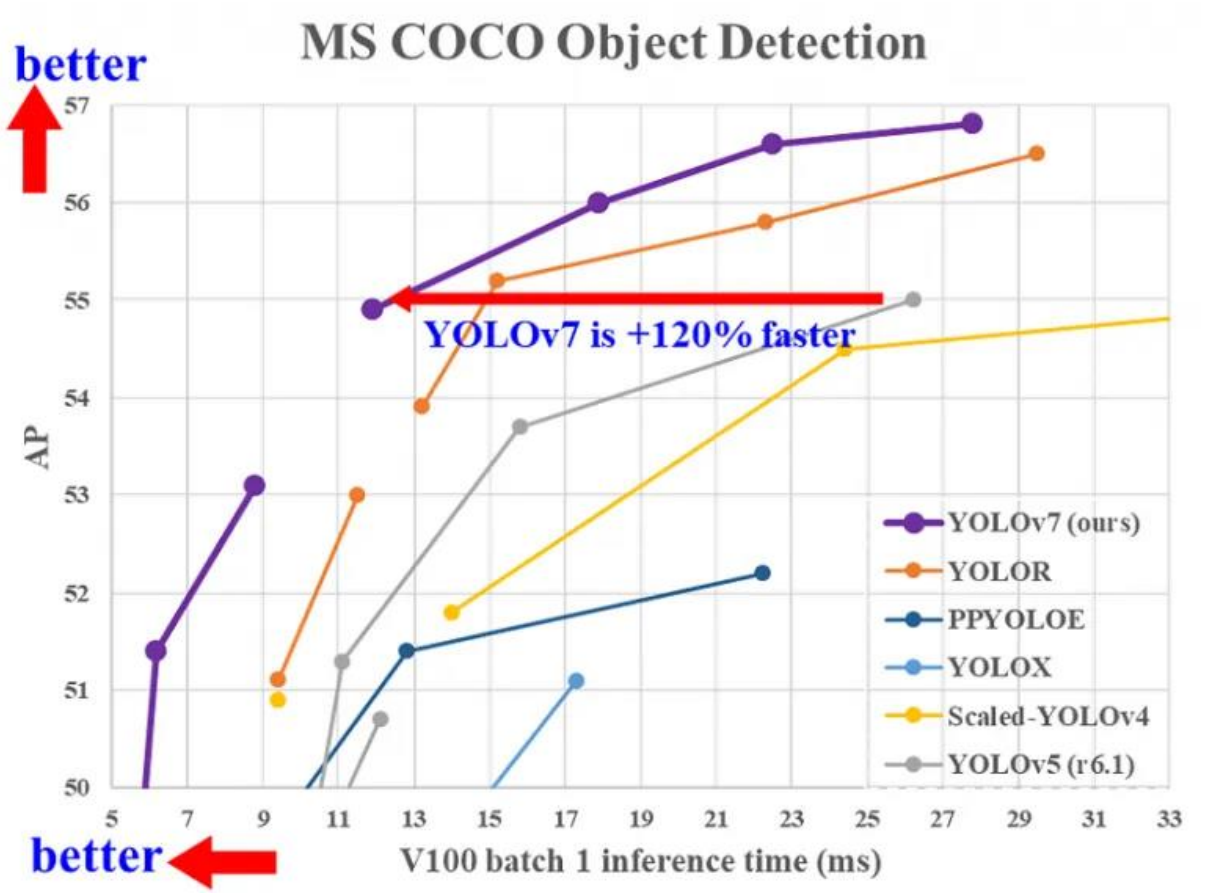


Рисунок 1.4 – Швидкість нейромережевих алгоритмів

Порівняно з традиційним підходом комп'ютерного зору до ранньої обробки зображень 20 років тому, глибоке навчання вимагає лише інженерних знань інструменту машинного навчання, а не досвіду в конкретних областях машинного зору для створення ручних функцій. Однак глибоке навчання вимагає ручного маркування даних для анотування хороших і поганих зразків, процес називається анотацією зображення.

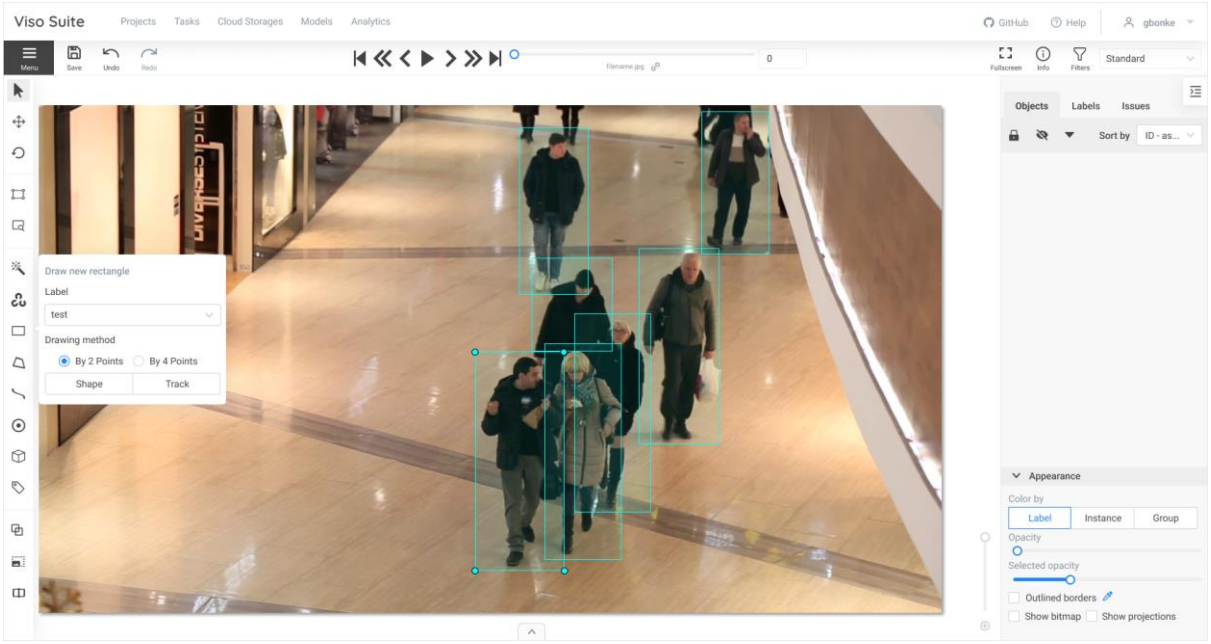


Рисунок 1.5 – Швидкість навчання під наглядом

Процес навчання на основі даних, позначених людьми, називається Supervised Learning. Процес створення таких мічених даних для навчання моделей штучного інтелекту вимагає трудомісткої людської роботи, наприклад, щоб анотувати стандартні дорожні ситуації під час автономного водіння.

1.2 Підходи до вирішення задачі розпізнавання зображень

Контрольована класифікація базується на ідеї, що користувач може вибрати зразки пікселів на зображенні, які є репрезентативними для певних класів, а потім наказати програмному забезпеченню обробки зображень використовувати ці навчальні вибірки як посилання для класифікації всіх інших пікселів на зображенні. Існує багато різних підходів до вирішення цієї задачі, проте в цій роботі будуть розглянуті: KNN, Decision Tree, SVM, CNN, LSSVM і запропонований підхід.

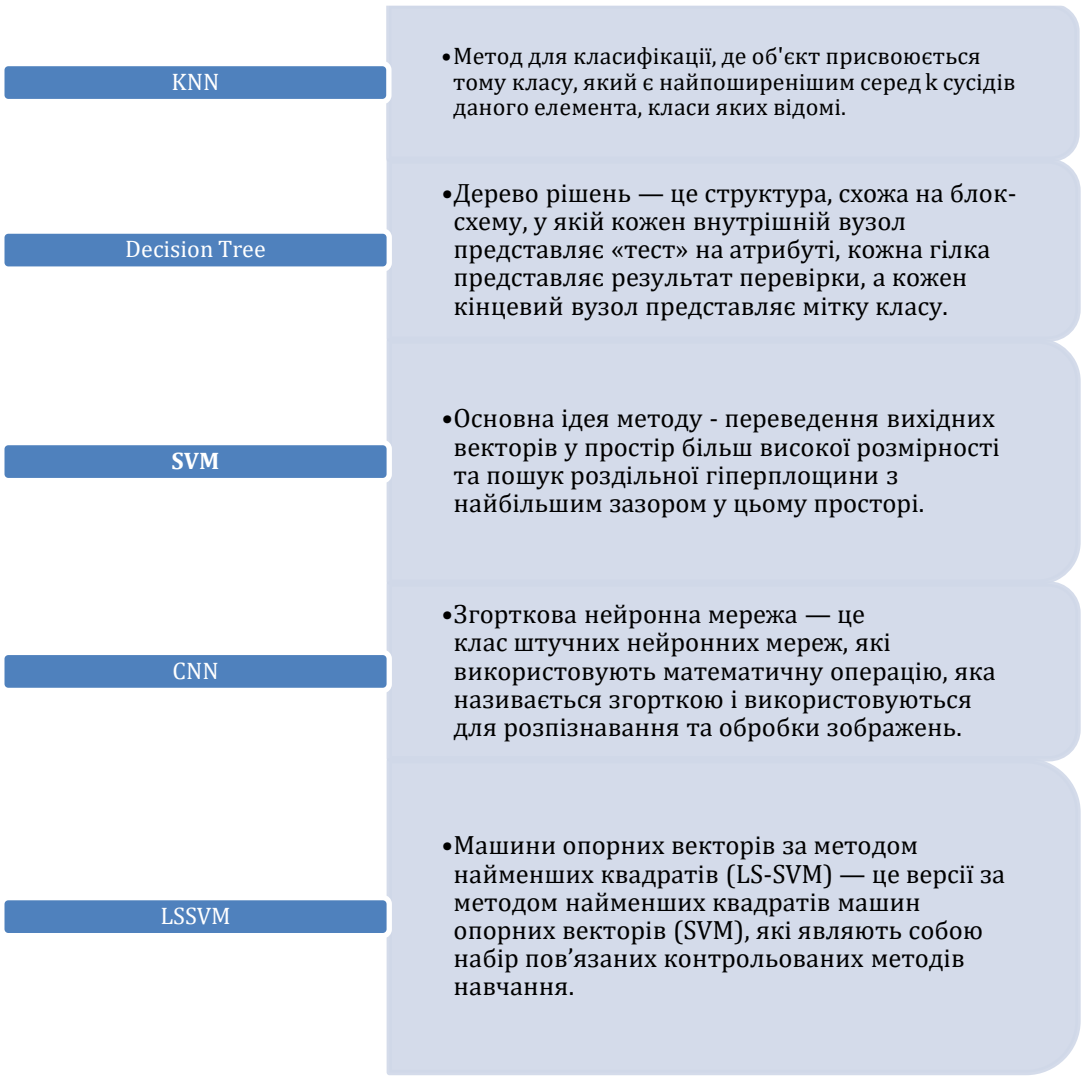


Рисунок 1.6 – Опис підходів для вирішення задачі класифікації

Машина опорних векторів (SVM) — це контрольований метод класифікації, який після етапу навчання може визначити, чи належить нова точка до класу чи іншого класу з найвищою математичною точністю. Це двійковий метод класифікації, але за допомогою підходу під назвою One vs All можна використовувати SVM для багатокласової класифікації.

Якщо набір даних є лінійно роздільним, це означає, що є можливість використати підхід жорсткого поля, або, точніше, знайти дві паралельні гіперплощини, які розділяють два класи даних, щоб відстань між ними була якомога більшою, щоб можна було визначити, до якого класу належить кожна точка набору даних. Але здебільшого набори даних не

розділяються лінійно, тому існує два шляхи: один — продовжити використання лінійного підходу з використанням м’якої маржі або спростити, допустити деяку неправильну класифікацію. У той час як другий спосіб полягає у використанні нелінійного ядра (яке має задовольняти умову Мерсера) або, скоріше, відображення даних у просторі вищої розмірності, де дані є лінійно роздільними, а завдання класифікації можна вирішити легко, навіть без потреби для розрахунку проекцій точок.

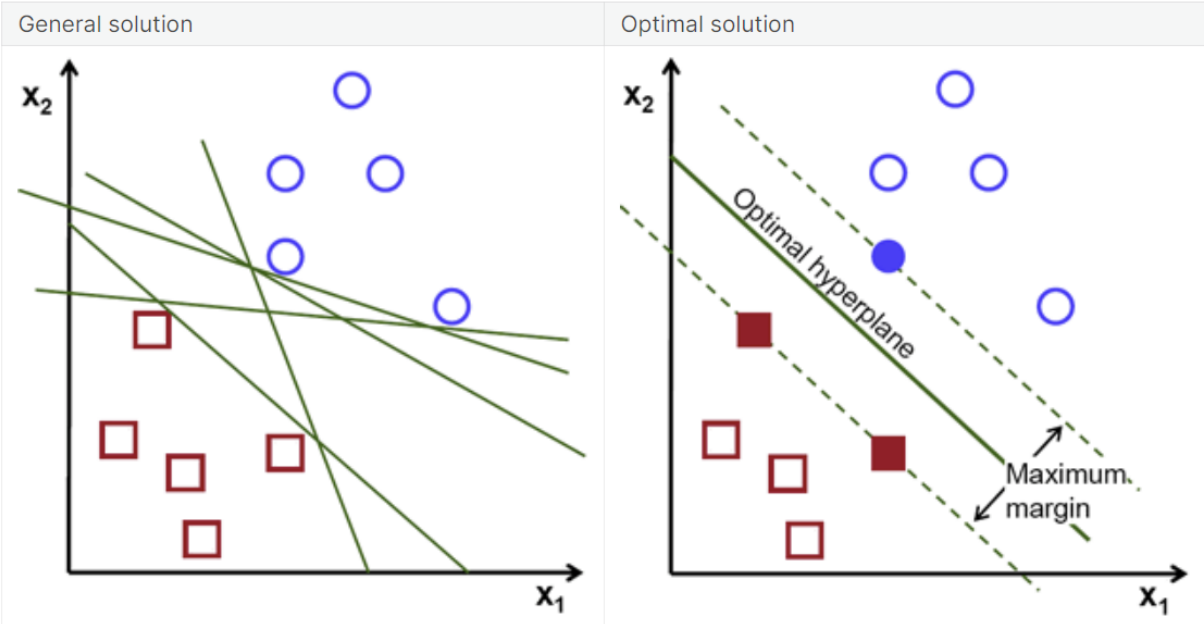


Рисунок 1.7 – Приклад побудови алгоритму методу опорних машин

Якщо використовується підхід лінійної м’якої маржі, завдання оптимізації полягає в тому, щоб знайти якомога більший запас, а потім потрібно додати штраф, якщо точка неправильно класифікована. Це досягається шляхом додавання до задачі оптимізації ще одного члена втрати, регуляризованого змінною C.

Функція втрати, яка використовується для цієї проблеми, є Hinge Loss.

$$L\big(y, f(x)\big) = \max\left(0{,}1 - y * f(x)\right)$$

(1.1)

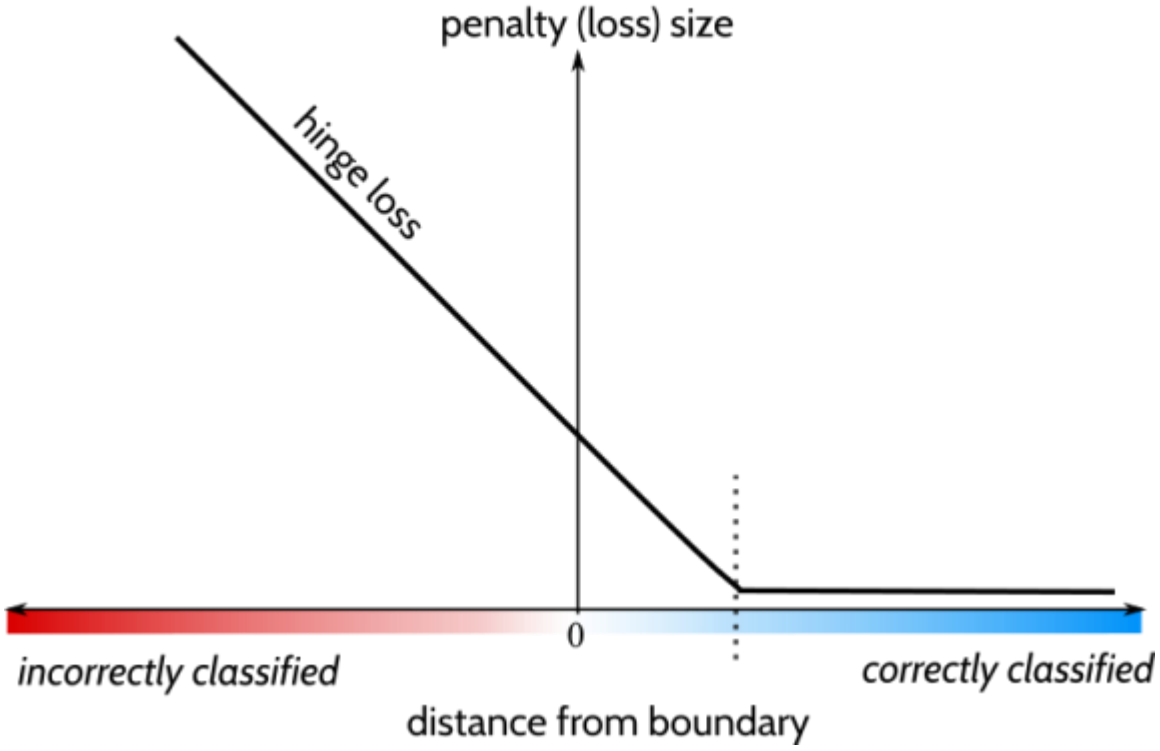


Рисунок 1.8 – Оптимізація алгоритму методу опорних машин

Проблема оптимізації, яку потрібно вирішити, буде:

$$minimize\frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\xi_i,\ \ y_i[x_i * w + b] \geq 1 - \xi_i \qquad (1.2)$$

де $\frac{1}{||w||}$ – розмір поля;

C –гіперпараметр;

ξ_i відстань від точки до межі.

Зрештою, потрібно знайти правильний компроміс між маржею та штрафом. Отже, єдиним параметром, який можна вибрати, є C, тому що задача оптимізації та обчислення пов’язані з калькуляторами.

Тип ядра можна встановити як лінійний, якщо проблема є лінійно роздільною, інакше ядро Гауса (RBF) добре підходить більшу частину

часу. Гама — це гіперпараметр ядра, який намагається точно відповідати навчальним даним.

KNN — це контрольований метод навчання, який розглядає K найближчих навчальних прикладів до точки інтересу для прогнозування свого класу. Очко присвоюється найближчому класу. Можуть бути застосовані різні метрики відстані, такі як: евклідова, зважена, гаусова тощо. Кроки досить прості:

1) отримати несекретні дані;

2) виміряти відстань за допомогою вибраних показників від нових даних до всіх інших даних, які вже засекречені;

3) отримати K менших відстаней;

4) перевірити список класів, які мали найкоротшу відстань, і підрахувати кількість кожного класу, який з’являється;

5) взяти за правильний клас клас, який з’явився найбільше разів;

6) класифікує нові дані за класом, який взяті на попередньому кроці.

У дереві рішень кожен проміжний вузол дерева містить атрибути розбиття, які використовуються для побудови різних шляхів, тоді як листи містять мітки класів.

Існують різні алгоритми для побудови дерева рішень, усі створені з жадібним підходом, оптимальні локально. Найвідомішим є алгоритм Ханта.

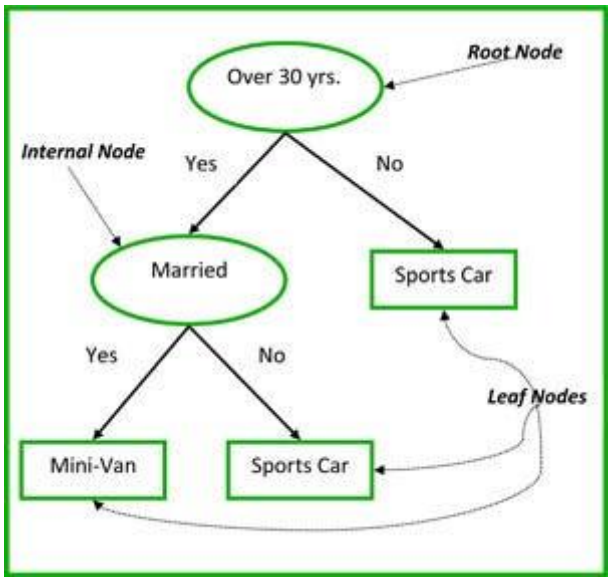


Рисунок 1.9 – Приклад дерева рішень

Починаючи з порожнього дерева, потрібно знайти ітераційно найкращий атрибут, за яким локально розбивати дані на кожному кроці. Якщо підмножина містить записи, які належать до того самого класу, тоді створюється аркуш, що містить таку мітку класу, інакше, якщо підмножина порожня, за замовчуванням призначається головному класу.

Критичними точками дерев рішень є умова перевірки, вибір найкращого атрибута та умова розщеплення. Для вибору найкращого атрибута зазвичай вибирають атрибут, який генерує однорідні вузли. Існують різні показники, щоб знайти найкращу однорідність розщеплення, найпоширенішими є:

Gini Impurity Index: Дано n класів і p_i частка предметів і класу у підмножині p, для i $\in\{1,2,...,n\}$. Тоді індекс GINI визначається як:

$$GINI = 1 - \sum_{i=1}^{n} p_i^2 \tag{1.3}$$

Коефіцієнт Інформаційного Прибутку: Інформаційний приріст базується на зменшенні ентропії після того, як набір даних розділено на атрибут. Побудова дерева рішень полягає в тому, щоб знайти атрибут,

який повертає найбільший приріст інформації (тобто, найбільш однорідні гілки).

Ентропія визначається як:

$$H(i) = -\sum_{i=1}^{n} p_i log_2 p_i \tag{1.4}$$

Отже, виграш інформації визначається як:

$$IG = H(p) - H(p,i) = H(p) - \sum_{i=1}^{n} \frac{ni}{n} H(i) \tag{1.5}$$

де p — батьківський вузол.

Перевагами дерев рішень є швидкість, легкість інтерпретації та хороша точність, але на них можуть вплинути відсутні значення.

Згорткові нейронні мережі (CNN) – це тип алгоритму глибокого навчання, який було розроблено спеціально для роботи із зображеннями та іншими сітковими даними, такими як аудіосигнали та дані часових рядів. Архітектура CNN для класифікації зображень включає згорткові шари, шари максимального об’єднання та повністю зв’язані шари.
Базова архітектура CNN для класифікації зображень (Рис 1.10)

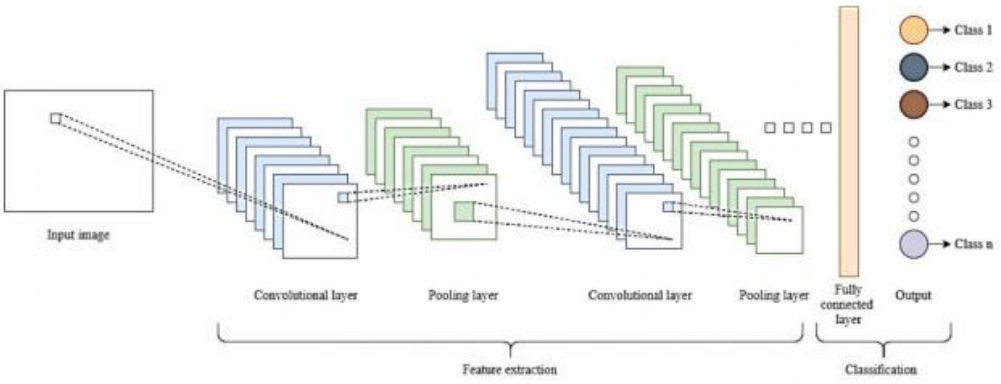


Рисунок 1.10 – Схема конволюційної нейромережі

У наведеній вище архітектурі наступні різні рівні:

- вхідний рівень: цей шар приймає дані вхідного зображення;
- згортковий рівень 1 : цей шар застосовує набір фільтрів до вхідного зображення, щоб виділити низькорівневі елементи, такі як краї та кути. Вихід згорткового шару передається, скажімо, функції активації випрямленої лінійної одиниці (ReLU). Можна використовувати інші функції активації. Це вносить нелінійність у модель;
- рівень об’єднання 1: цей шар зменшує вибірку карт функцій, створених першим згортковим шаром, зменшуючи їх розмірність і роблячи їх обчислювально ефективнішими для обробки;
- згортковий рівень 2 : цей рівень застосовує набір фільтрів до результату першого максимального шару об’єднання, щоб отримати складніші функції, такі як візерунки та текстури. Вихід згорткового шару передається до функції активації випрямленої лінійної одиниці (ReLU). Це вносить нелінійність у модель;
- максимальний рівень об’єднання 2 : цей шар зменшує вибірку карт функцій, створених другим згортковим шаром, ще більше зменшуючи їх розмірність.

Прикладами класифікаційного навчального завдання, де використовується CNN, є класифікація зображень, виявлення об’єктів і розпізнавання обличчя.

1.3. Адаптивне навчання

Для впровадження моделей машинного навчання у виробництво використовуються різні підходи. Досить часто моделі запускаються у виробництво після одноразового навчання (стаціонарні моделі). Щоб така модель продовжувала точно прогнозувати протягом тривалого часу, дані, на основі яких вона робить прогнози, повинні мати такий же розподіл, як і

дані, на яких була навчена модель. Однак з часом розподіли можуть змінюватися.

Такі відхилення називають дрейфом концепції (Рис 1.11). (Поняття у «зміні поняття» стосується невідомого та прихованого зв'язку між вхідними та вихідними змінними. Поняття також відомі як коваріати.)

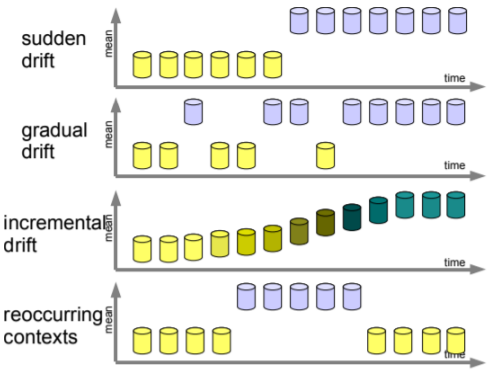


Рисунок 1.11 – Типи дрейфу концепції

Одна з технік, яка вирішує зміну концепції з часом є онлайн методом. Це тип машинного навчання, який використовує динамічні вхідні дані (вхідні дані в реальному часі, наприклад, дату датчика) після припущення початкової статичної моделі. Вхідні дані обробляються один за одним.

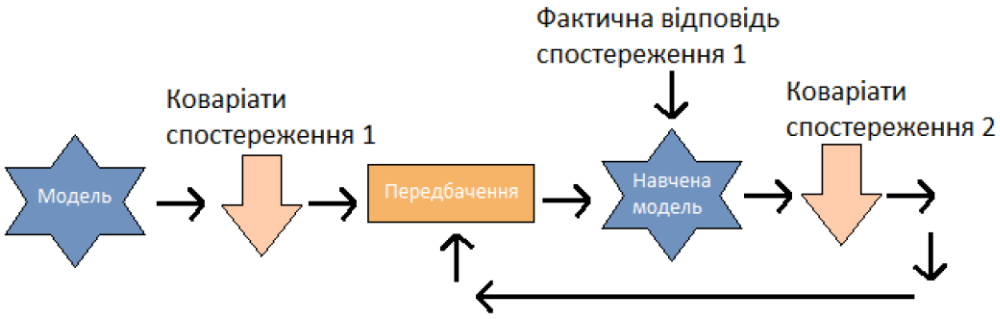


Рисунок 1.12 – Блок-схема онлайн-моделей

Штучні нейронні мережі вимагають великої обчислювальної потужності. Нейронні мережі моделюються за мозком і складаються з багатьох взаємопов'язаних вузлів обробки. Кожен вузол виконує обчислення на основі своїх вагових параметрів і коригує їх за допомогою зворотного поширення . Через багато параметрів ШНМ також потребує більших наборів даних для навчання. З цих причин ANN вимагає високої обчислювальної потужності.

Більший пакет тренував би мережу швидше за епоху, але потребував би більше пам'яті. Менші розміри пакетів займуть більше епох для навчання мережі, але вимагатимуть менше пам'яті.

Навчання нейронної мережі вимагає багато даних. Нейронні мережі дуже гнучкі і можуть навчитися розпізнавати шаблони вхідних даних. Ця гнучкість має свою ціну. Нейронні мережі потребують багато даних для навчання . Вони не в змозі зробити узагальнення на основі обмежених навчальних даних. На невеликих наборах даних моделі нейронних мереж мають тенденцію до надмірного пристосування. Вони запам'ятовують навчальні дані та погано узагальнюють нові приклади.

Простіша модель машинного навчання працюватиме краще, ніж нейронна мережа на невеликих наборах даних. Проте розглянуті в даній роботі підходи також мають певні недоліки.

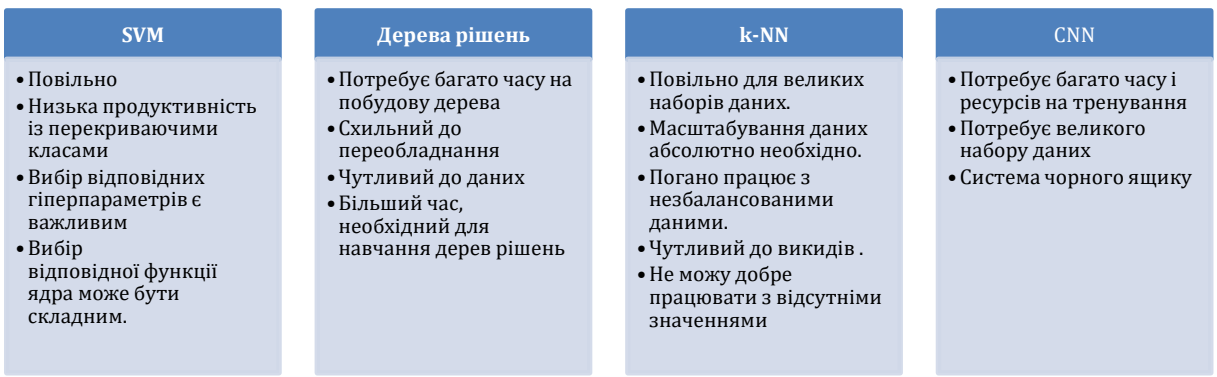


Рисунок 1.13 – Недоліки різних підходів для класифікації в онлайн режимі

Ми можемо спробувати вирішити ці проблеми за допомогою запропонованого в даній роботі методу, що є різновидом LSSVM – популярного методу машин5ного навчання та статистичного аналізу, який можна використовувати для онлайн-навчальних завдань. Є кілька причин, чому LSSVMa добре підходить для онлайн-навчання:

1) LSSVMa — це алгоритм навчання на основі ядра, який може ефективно обробляти дані великої розмірності. Це робить його придатним для онлайн-навчальних завдань, де кількість вхідних змінних може бути великою або невідомою;

2) LSSVMa базується на задачі опуклої оптимізації, яку можна ефективно розв’язати за допомогою методів чисельної оптимізації. Це робить його добре придатним для завдань онлайн-навчання, коли дані генеруються безперервно, а модель потрібно оновлювати в режимі реального часу;

3) LSSVMa має вбудований параметр регуляризації, який допомагає запобігти переобладнанню та покращити продуктивність узагальнення моделі. Це важливо для завдань онлайн-навчання, коли модель має адаптуватися до нових даних, уникаючи при цьому надмірного підбору даних навчання;

4) LSSVMa можна навчати поступово, що означає, що нові дані можна додавати до навчального набору та оновлювати модель без необхідності повторного навчання всієї моделі з нуля. Це робить його придатним для завдань онлайн-навчання, де постійно генеруються нові дані, а модель потрібно оновлювати в режимі реального часу;

5) LSSVMa — це надійний метод, який може ефективно обробляти зашумлені дані. Це важливо для завдань онлайн-навчання, де якість даних може бути змінною, і модель повинна мати можливість адаптуватися до шумних даних без шкоди для її продуктивності.

Загалом, LSSVMa — це гнучкий і надійний метод, який можна використовувати для широкого спектру онлайн-навчальних завдань. Його

здатність ефективно обробляти багатовимірні дані, вбудований параметр регулярізації та можливість поступового навчання роблять його привабливим варіантом для онлайн-додатків для навчання.

1.4 Постановка задачі дослідження

Метод матричних опорних векторних машин — популярний підхід до розпізнавання образів, який використовує алгоритм навчання на основі ядра для класифікації даних у різні категорії. Останніми роками зростає інтерес до поєднання методу LSSVMa з адаптивними методами навчання для покращення його продуктивності та ефективності при обробці великих і складних наборів даних.

Метою дослідження цієї магістерської роботи є дослідження ефективності методу матричних опорних векторних машин з адаптивним комбінованим підходом до навчання функції активації в задачі розпізнавання образів. Зокрема, дослідження спрямоване на дослідження того, як поєднання методу LS-SVMa і адаптивного навчання може підвищити точність і ефективність розпізнавання образів порівняно з традиційними підходами. Дослідження включатиме аналіз і порівняння ефективності методу LS-SVMa з підходом адаптивного навчання та без нього з використанням набору контрольних наборів даних. Результати цього дослідження сприятимуть розумінню методу LS-SVMa і дадуть зрозуміти його потенціал для практичного застосування в задачах розпізнавання образів.

Якщо є можливість передбачення і тренування моделі в офлайн немає обмежень часу і існує можливість обробки невеликих даних. Тобто можна використати актуальну інформацію останню по часу, такі вибірки мають невеликий об’єм. Однак існуючі методи обробки зображень розраховані на навчальні вибірки середнього і великого розміру, що не дає можливості використати ці методи для оперативного аналізу в режимі

онлайн. Тому розробка методів розпізнавання зображень на основі коротких вибірок потребує проведення подальших досліджень.

Об’єктом дослідження є процес розпізнавання образів і обробки зображень.

Предметом дослідження є методи розпізнавання зображень з використанням коротких вибірок, покращений метод опорних векторів , що залежить від коротких вибірок.

Постановка задачі – матричний метод опорних векторів з адаптивним комбінованим навчанням активаційної функції для дослідження опису характеристик даних зображень з метою покращення стабільності і точності передбачень.

Метою роботи є удосконалення методу опорних векторів з використанням адаптивного комбінованого навчання активаційної функції.

2 РОЗРОБКА МАТРИЧНОГО МЕТОДУ ОПОРНИХ ВЕКТОРІВ З АДАПТИВНИМ КОМБІНОВАНИМ НАВЧАННЯМ АКТИВАЦІЙНОЇ ФУНКЦІЇ

SVM є одними з найкращих (і багато хто вважає, що вони дійсно найкращі) «готовий» алгоритм навчання під наглядом. Алгоритм SMO, який забезпечує ефективну реалізацію SVM.

2.1 Геометричні поля

Розглянемо логістичну регресію, де ймовірність $\mathrm{p}(\mathrm{y} = 1|\mathrm{x};\ \theta)$ моделюється за допомогою

$$h_\theta(\mathrm{x}) = \mathrm{g}(\theta^T \mathrm{x}) \tag{2.1}$$

Тоді була б змога передбачити «1» на вхідних даних x, якщо і тільки якщо $h_\theta(\mathrm{x}) \geq 0{,}5$, або, еквівалентно, тоді і тільки тоді, коли $\theta^T \mathrm{x} \geq 0$. Розглянемо наприклад позитивного навчання (y = 1). Чим більше $\theta^T \mathrm{x}$, чим більше також є$h_\theta(\mathrm{x}) = p(y = 1|\mathrm{x};\ \mathrm{w}, \mathrm{b})$ і, отже, вищий ступінь «впевненості», що мітка дорівнює 1. Таким чином, неофіційно можна вважати прогноз таким дуже впевнений, що y = 1, якщо $\theta^T \mathrm{x} \geq 0$.

Для іншого прикладу, візьмемо два класи (Рис 2.1), на якому x представляють позитивні приклади навчання, о позначає негативні приклади навчання, межа рішення (це лінія, задана рівнянням $\theta^T \mathrm{x} = 0$, і (також називається роздільною гіперплощиною) також показано та три точки також позначені A, B і C.

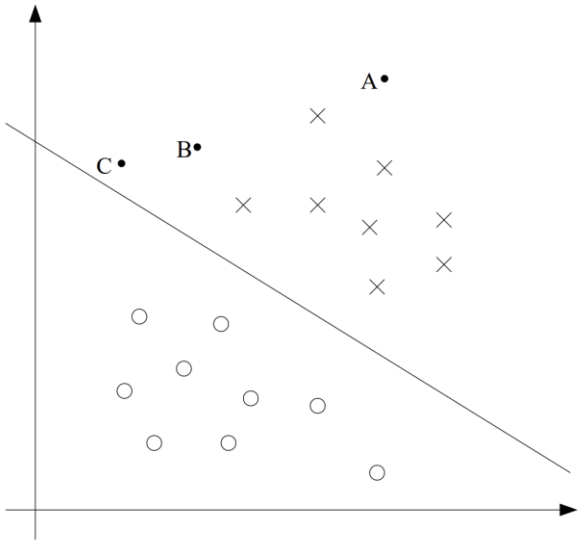


Рисунок 2.1 – Точки розподілені гіперплощиною

Точка A знаходиться дуже далеко від межі рішення. Якщо знаходити прогноз для значення y в точці A, здається, що там y = 1. І навпаки, точка C знаходиться дуже близько до межі прийняття рішення, і поки він знаходиться збоку від межі прийняття рішення прогноз був би y = 1, що здається ймовірним, що лише невелика зміна границя прийняття рішення могла легко спричинити те, що прогноз став y = 0. Отже, більша впевненість щодо прогнозу на A, ніж на C. Точка B лежить між цими двома випадками, і в більш широкому сенсі можна помітити, що якщо точка знаходиться далеко від розділової гіперплощини, то можна бути значно більш впевненим в своїх прогнозах.

Щоб спростити обговорення SVM, спершу потрібно представити нові позначення для розмови про класифікацію на прикладі лінійного класифікатору для задачі бінарної класифікації з мітками y і ознаками x. Відтепер введемо $y \in \{-1, 1\}$ (замість $\{0, 1\}$) для позначення міток класу. Крім того, замість того, щоб параметризувати лінійний класифікатор вектором θ, введемо параметри w, b і запис класифікатору набуде наступного вигляду:

$$h_{w,b}(x) = g(w^T x + b) \quad (2.2)$$

де g(z) = 1, якщо z ≥ 0, і g(z) = −1 в іншому випадку.

Це позначення «w, b» дозволяє явно розглядати перехоплений член b окремо від іншого параметру. Таким чином, b виконує роль, що раніше було θ_0, а w виконує роль $[\theta 1 \dots \theta n]^T$.

Зауважимо також, що з визначення g вище, класифікатор буде безпосередньо передбачувати або 1, або −1, без попереднього переходу через проміжний крок оцінки ймовірності того, що y дорівнює 1 (що й зробила логістична регресія).

2.2 Функціональний запас

Давайте формалізуємо поняття функціональних і геометричних полів. Враховуючи навчальний приклад $(x^{(i)}, y^{(i)})$, визначаємо функціональний запас (w, b) з увагою до навчального прикладу

$$\gamma^{(i)} = y^{(i)}(w^T x + b) \quad (2.3)$$

Варто зауважити, що якщо $y^{(i)} = 1$, то для великого функціонального запасу (тобто прогноз має бути впевненим і правильним), то потрібно $w^T x + b$ було великим позитивним числом. І навпаки, якщо$y^{(i)} = -1$, то щоб функціональний запас був великим, знадобиться щоб $w^T x + b$ був великим від'ємним числом. Крім того, якщо $y^{(i)}(w^T x + b) > 0$, то прогноз у цьому прикладі правильний. Отже, великий функціональний запас означає впевненість і правильний прогноз.

Для лінійного класифікатора з вибором g, наведеним вище (беручи значення в$\{-1, 1\}$), є одна властивість функціонального поля, яка робить його не дуже хорошою мірою впевненості. Враховуючи вибір g,

зауважимо, що якщо замінити w на 2w і b на 2b, то оскільки $g(w^T x + b) = g(2w^T x + 2b)$, це взагалі не змінить $h_{w,b}(x)$. Тобто g, також $h_{w,b}(x)$, залежить тільки на знак, але не на величину $w^T x + b$. Проте, замінюючи (w, b) з (2w, 2b) також призводить до множення функціонального запасу на а з коефіцієнтом 2. Таким чином, здається, що, використовуючи нашу свободу масштабування w і b, є змога зробити функціональний запас як завгодно великим без реальних значущих змін. Інтуїтивно зрозуміло, що може мати сенс нав'язати якусь умову нормалізації, наприклад ||w||2 = 1; тобто можна замінити (w, b) на (w/||w||2, b/||w||2), і замість цього розглянути функціональний запас (w/||w||2, b/||w||2).

Даючи навчальний набір $S = \{(x^i, y^i); i = 1, \dots, m\}$, також визначаємо запас функції (w, b) відносно S як найменший функціонал поля окремих навчальних прикладів. Позначаючи γˆ, тому це може бути написано:

$$\gamma = min_{i=1,\dots m}\gamma^{(i)} \tag{2.4}$$

Введемо означення геометричного поля (Рис 2.2).

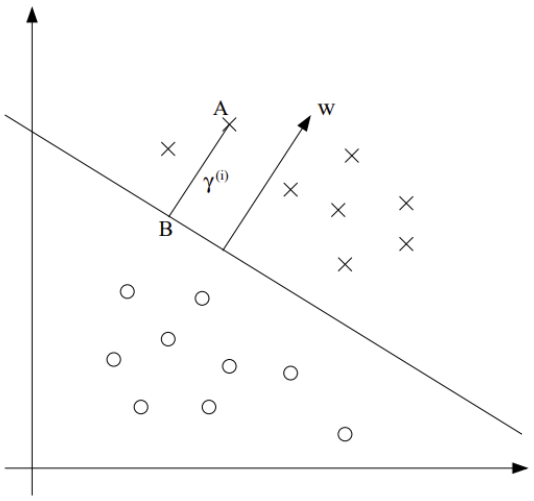


Рисунок 2.2 – Точки розподілені гіперплощиною

Границя рішення, що відповідає (w, b), показана разом із вектором w. Варто зауважити, що w є ортогональним (при 90^o) до розділової гіперплощини.

Якщо розглянути точку на A, яка представляє вхід $x^{(i)}$ деякого навчального прикладу з міткою $y^{(i)} = 1$. Його відстань до межі прийняття рішення, $\gamma^{(i)}$ що, задається лінією AB.

Знаходження значення $\gamma^{(i)}$. w/||w|| є вектором одиничної довжини вказуючи в тому ж напрямку, що й w. Оскільки A представляє $x^{(i)}$, тому знаходимо, що точка B задана $x^{(i)} - \gamma^{(i)} * w/||w||$. Але цей пункт лежить на межі рішення, і всі точки x на межі рішення задовольняють рівняння $w^T x + b = 0$. Отже,

$$w^T \left(x^{(i)} - \gamma^{(i)} \frac{w}{||w||} \right) + b = 0 \tag{2.5}$$

Вирішуючи для $\gamma^{(i)}$

$$\gamma^{(i)} = \frac{w^T x^{(i)} + b}{||w||} = \left(\frac{w}{||w||} \right)^T x^{(i)} + \frac{b}{||w||} \tag{2.6}$$

Це було розроблено для випадку позитивного прикладу навчання, де добре бути на «позитивній» стороні межі прийняття рішення. Більш загально, можна визначити геометричне поле (w, b) відносно навчального прикладу $(x^{(i)}, y^{(i)})$:

$$\gamma^{(i)} = y^{(i)} \left(\left(\frac{w}{||w||} \right)^T x^{(i)} + \frac{b}{||w||} \right) \tag{2.7}$$

Потрібно зауважити, що якщо ||w|| = 1, то функціональний запас дорівнює геометричному— таким чином, це дає спосіб співвідносити ці два різні поняття запасу. Крім того, геометричний запас є інваріантним

до зміни масштабу параметрів; тобто якщо замінити w на 2w і b на 2b, то геометричне поле не змінюється. Зокрема, тому що цієї інваріантності до масштабування параметрів при спробі підібрати w і b до навчальних даних є змога накласти довільне масштабне обмеження на w без зміни чогось важливого; наприклад, є можливість вимагати, щоб ||w|| = 1 або $|w_1| = 5$, або$|w_1 + b| + |w_2| = 2$, і будь-яке з них може бути задоволено просто зміною масштабу w і b.

Нарешті, дано навчальний набір $S = \{(x^i, y^i); i = 1, \dots, m\}$ також визначимо геометричний запас (w, b) відносно S має бути найменшим із геометричних полів на окремих навчальних прикладах:

2.3 Оптимальний запас класифікатора

Необхідно знайти межу рішення, яка максимізує (геометричний) запас, оскільки це буде відображати дуже впевнений набір прогнозів на тренувальному наборі та добре «відповідає» навчальним даним. Зокрема, це призведе до класифікатора, який розділяє позитивне та негативне навчання приклади з «зазором» (геометричним запасом).

Наразі припустимо, що дано навчальний набір, який є лінійним роздільний; тобто, що можна розділити позитивний і негативний приклади за допомогою деякої розділової гіперплощини. Можна поставити таку задачу оптимізації для знаходження максимального геометричного запасу:

$$\begin{gathered} max_{\gamma,w,b}\ \gamma \\ s.t.\ y^{(i)}\left(w^T x^{(i)} + b\right) \geq \gamma, i = 1, \dots, m \\ \left||w|\right| = 1 \end{gathered} \tag{2.8}$$

Метою є максимізувати γ, за умови, що кожен навчальний приклад має функціональний запас (принаймні обмеження γ. ||w|| = 1), крім того,

гарантує, що функціональний запас дорівнює геометричному запасу, тому також є гарантія що всі геометричні поля не менше γ. Таким чином, вирішення цієї проблеми буде результат (w, b) з найбільшим можливим геометричним запасом по відношенню до навчальний набір.

Обмеження "||w|| = 1" є не опуклим, і це, звичайно, проблема, що не вирішити стандартним програмним забезпеченням для оптимізації. Для досягнення результату, необхідно спростити поставлену проблему. Розглянемо:

$$max_{\gamma,w,b} \ \frac{\gamma}{||w||}$$
$$s.t. y^{(i)}\left(w^T x^{(i)} + b\right) \geq \gamma, i = 1, \dots, m \qquad (2.9)$$

Тут потрібно максимізувати γˆ/||w|| з урахуванням усіх функціональних запасів щонайменше γˆ. Оскільки геометричні та функціональні поля пов'язані між собою γ = γˆ/||w|, це дасть потрібну відповідь. Більш того, немає тепер обмеження ||w|| = 1. Мінусом є те, що зараз маємо невипуклу цільову $\frac{\gamma}{||w||}$ функцію. На попередньому кроці було додано довільне обмеження масштабування для w і b без змін. Це ключова ідея, яка буде корисною для подальших кроків. Потрібно ввести обмеження масштабування функціоналу запас w, b щодо навчального набору має бути 1:

$$\gamma = 1 \qquad (2.9)$$

Оскільки множення w і b на деяку константу призводить до функціонального запасу будучи помноженим на ту саму константу, це справді є обмеженням масштабування, і може бути задоволено зміною масштабу w, b. Підключивши це до нашої проблеми вище, і зауважуючи,

що максимізація γˆ/||w|| = 1/||w|| це те саме, що мінімізувати $||w||^2$, маємо таку проблему оптимізації:

$$min_{\gamma,w,b}\ \frac{1}{2}||w||^2$$
$$s.t. \qquad y^{(i)}\left(w^T x^{(i)} + b\right) \geq \gamma, i = 1, \dots, m \tag{2.10}$$

Тепер проблема трансформована в форму, яка може бути ефективною вирішено. Вище наведено задачу оптимізації з опуклою квадратичною ціллю та лише лінійними обмеженнями. Його рішення дає оптимальний класифікатор маржі. Цю проблему оптимізації можна вирішити за допомогою комерційних код квадратичного програмування (QP).

2.4 Формування запасу класифікатора

Раніше була поставлена наступна (основна) задачу оптимізації для пошуку оптимальний запас класифікатору:

$$min_{\gamma,w,b}\ \frac{1}{2}||w||^2$$
$$s.t. y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1, i = 1, \dots, m \tag{2.10}$$

Можна записати обмеження як:

$$g_i(w) = -y^{(i)}\left(w^T x^{(i)} + b\right) + 1 \leq 0. \tag{2.11}$$

Існує одне таке обмеження для кожного навчального прикладу. Зазначимо, що з в умови подвійної додатковості ККТ, матимемо $\alpha_i > 0$ лише для прикладів навчання, які мають функціональний запас, рівний

одиниці (тобто ті, що відповідають обмеженням, які виконуються з рівністю, $g_i(w) = 0$). Розглянемо малюнок нижче, на якому показано максимальне поле, що розділяє гіперплощину суцільною лінією.

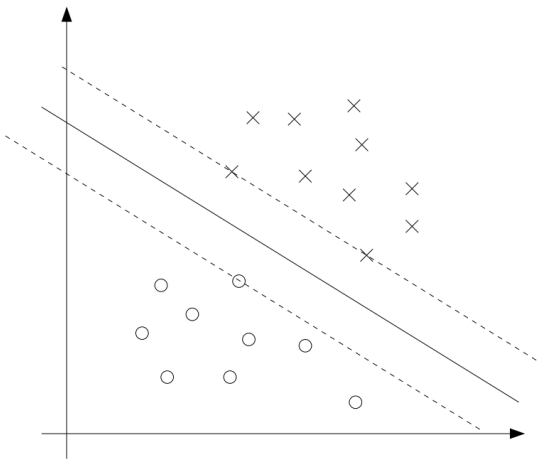


Рисунок 2.3 – Точки розподілені гіперплощиною і відступи

Точки з найменшими полями є саме тими, що найближчі до межі рішення; тут це три точки (один негативний і два позитивних приклади), які лежать на пунктирних лініях, паралельних межі прийняття рішення. Таким чином, лише три з α_i — саме ті, що відповідають цим трьом навчальні приклади—буде ненульовим при оптимальному розв'язанні нашої задачі оптимізації. Ці три точки називають опорними векторами проблема. Справа в тому, що число опорних векторів може бути набагато менше ніж розмір, навчальний набір стане в нагоді пізніше.

Заглядаючи вперед, у міру розвитку подвійної форми проблеми, одна ключова ідея, на яку варто звернути увагу, полягає в тому, що можна спробувати написати алгоритм у термінах лише скалярного добутку. Той факт, що можна виразити алгоритм з точки зору цих внутрішніх продуктів буде ключовим, коли введемо термін «трюк ядра». Коли будується лагранжіан для задачі оптимізації, маємо:

$$L(w,b,\alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i \left[y^{(i)} \left(w^T x^{(i)} + b \right) - 1 \right] \quad (2.12)$$

Зауважте, що є лише "α_i", але немає "β_i", множників Лагранжа, оскільки проблема має лише обмеження нерівності. Знайдемо подвійну форму задачі. Для цього потрібно спочатку мінімізувати $L(w,b,\alpha)$ відносно w і b (для фіксованого α), щоб отримати θ_ρ, встановивши похідні L відносно w і b рівними нулю. Тоді маємо:

$$\nabla_w L(w,b,\alpha) = w - \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} = 0 \quad (2.13)$$

Це означає, що

$$w = \sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)} \quad (2.14)$$

Що стосується похідної по b, то отримуємо

$$\frac{\partial}{\partial b} L(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i y^{(i)} = 0 \quad (2.15)$$

Якщо взяти визначення w у рівнянні (2.14) і підключити його назад до Лагранжіан (2.13) і спростити, отримуємо

$$L(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left(x^{(i)} \right)^T x^{(j)} - b \sum_{i=1}^{m} \alpha_i y^{(i)} \quad (2.16)$$

Але з рівняння (2.15) останній член повинен дорівнювати нулю, тому рівняння виглядатиме наступним чином:

$$L(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left(x^{(i)} \right)^T x^{(j)} \quad (2.17)$$

Поєднуючи це разом із обмеженнями $\alpha_i \geq 0$ та обмеження (2.13), отримуємо наступну задачу подвійної оптимізації:

$$max_{\alpha} W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j \left(x^{(i)}\right)^T x^{(j)}$$

$$s.t. \qquad \alpha_i \geq 0, \;\; i = 1, \dots, m$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0, \qquad (2.18)$$

Умови, необхідні для $p^* = d^*$ і умови ККТ (рівняння 2.5–2.11), які потрібно виконувати, справді задовольняються в проблемі оптимізації. Отже, можна розв'язувати дуальне завдання замість розв'язування первинної проблеми. Зокрема, у подвійній задачі вище маємо а задача максимізації, в якій параметрами є α_i. Для вирішення подвійної проблеми (тобто знайти α, які максимізують W(α) з урахуванням обмежень), використовується рівняння (2.10), щоб повернутися та знайти оптимальне w як функція α. Знайшовши w^*, розглядаючи первинну проблему, також легко знайти оптимальне значення для члену перехоплення b як

$$b^* = -\frac{max_{i:y^i=-1} w^{*T} x^{(i)} + min_{i:y^i=1} w^{*T} x^{(i)}}{2} \qquad (2.19)$$

Рівняння (2.13), дає оптимальне значення w через (оптимальне значення) α. Припустимо адаптували параметри нашої моделі до навчального набору, і тепер хочемо зробити а прогноз у новій точці введення x. Тоді можна було б обчислити $w^T x + b$ і передбачити y = 1 тоді і тільки тоді, коли ця величина більша за нуль. Але використовуючи (2.13), цю величину також можна записати:

$$w^T x + b = \left(\sum_{i=1}^{m} \alpha_i y^{(i)} x^{(i)}\right)^T x + b = \sum_{i=1}^{m} \alpha_i y^{(i)} \langle x^{(I)}, x \rangle + b \qquad (2.20)$$

Отже, якщо знайдено α_i, щоб зробити прогноз, треба обчислити величину, яка залежить лише від скалярного добутку у навчальному наборі. Крім того, α_i буде все бути нульовим, за винятком опорних векторів. Таким чином, багато доданків у сумі вище буде дорівнювати нулю, і потрібно знайти лише внутрішні продукти між ними x і опорні вектори (яких часто є невелика кількість) в порядок обчислення (2.20) і зробити прогноз.

2.5 Функції ядра метода опорних векторів

Алгоритми SVM використовують групу математичних функцій, відомих як ядра. Функція ядра полягає в тому, щоб вимагати дані як вхідні дані та перетворювати їх у бажану форму.

Різні алгоритми SVM використовують різні типи функцій ядра. Ці функції бувають різних типів, наприклад, лінійна, нелінійна, поліноміальна, радіальна базисна функція (Radial basis function) і сигмоїда.

Найбажанішим типом функції ядра є RBF. Оскільки він локалізований і має кінцевий відгук уздовж повної осі x.

Функції ядра повертають скалярний добуток між двома точками в надзвичайно зручному просторі функцій. Таким чином, визначивши поняття подібності, з невеликими обчислювальними витратами навіть у випадку просторів дуже великої розмірності.

Популярні функції ядра SVM:

1) лінійне ядро – це найпростіший тип ядра, зазвичай одновимірного за своєю природою. Це найкраща функція, коли є багато функцій. Лінійне ядро здебільшого є кращим для задач класифікації тексту, оскільки більшість таких типів проблем класифікації можна розділити лінійно. Лінійні функції ядра працюють швидше , ніж інші функції. Формула лінійного ядра

$$F(x, x_j) = sum(x, x_j) \quad (2.21)$$

де x, x_j представляють дані, які потрібно класифікувати;

2) поліноміальне ядро – це більш узагальнене представлення лінійного ядра. Вона менш ефективна та точна. Формула ядра полінома:

$$F(x, x_j) = (x, x_j + 1)^d \quad (2.22)$$

де «.» показує скалярний добуток обох значень,

d позначає ступінь. $F(x, x_j)$, що представляє межу рішення для розділення заданих класів;

3) радіальна базисна функція Гауса (RBF) – це одна з найбільш бажаних і використовуваних функцій ядра в svm. Зазвичай його вибирають для нелінійних даних. Це допомагає зробити належне розділення, коли немає попереднього знання даних;

Формула радіального базису Гауса:

$$F(x, x_j) = \exp(-gamma * ||x - x_j||^2) \quad (2.23)$$

Значення гами змінюється від 0 до 1. Найбільш переважним значенням для гами є 0,1 .

4) сигмовидне ядро – здебільшого його вважають за краще для нейронних мереж . Ця функція ядра схожа на двошарову перцептронну модель нейронної мережі, яка працює як функція активації для нейронів.

Вибір ядра SVM для набору даних. Оскільки лінійне ядро займає менше часу на навчання порівняно з іншими функціями ядра:

– лінійне ядро в основному є кращим для задач класифікації тексту, оскільки воно добре працює з великими наборами даних;

– ядра Гауса зазвичай дають хороші результати, коли немає додаткової інформації щодо даних, які недоступні;

– ядро RBF також є різновидом ядра Гауса, яке проектує багатовимірні дані, а потім шукає для них лінійне розділення;

– поліноміальні ядра дають хороші результати для задач, де всі навчальні дані нормалізовані.

2.6 Регуляризація та випадок лінійної нероздільності класів

Висновування SVM, представлене досі, припускало, що дані є лінійно роздільні. Під час відображення даних у просторі ознак великої розмірності через φ, як правило, збільшує ймовірність того, що дані можна відокремити, не можна гарантувати, що так буде завжди. Наприклад, на рисунку (Рис 2.4) показує оптимальний запас класифікатору, а коли додається один викид у верхній лівій області (правий малюнок), це призводить до того, що межа рішення становить а різке коливання, і отриманий класифікатор має набагато менший запас.

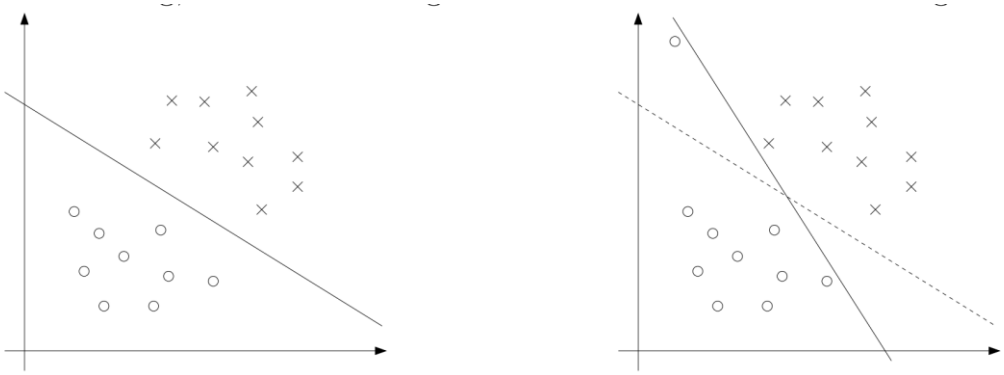


Рисунок 2.4 – Варіації нахилу гіперплощин

Щоб алгоритм також працював для нелінійно роздільних наборів даних щоб бути менш чутливими до викидів, переформулювано оптимізацію (використовуючи l_1регулярізація) наступним чином:

$$min_{\gamma,w,b}\ \frac{1}{2}||w||^2 + C\sum_{i=1}^{m}\xi_i$$
$$s.t.\, y^{(i)}\left(w^T x^{(i)} + b\right) \geq 1 - \xi_i, i = 1, \dots, m$$
$$\xi_i \geq 0, i = 1, \dots, m \quad (2.24)$$

Таким чином, приклади тепер можуть мати (функціональне) поле менше 1, і якщо приклад, функціональний запас якого становить $1 - \xi_i$, цільова функція збільшується на $C\xi_i$. Параметр C контролює відносна вага між двома цілями створення $||w||^2$ великий і забезпечення того, щоб більшість прикладів мати функціональний запас не менше 1. Як і раніше, можна сформувати лагранжіан:

$$L(w, b, \xi, \alpha, r) = \frac{1}{2}w^T w + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i\left[y^{(i)}(x^T w + b) - 1 + \xi_i\right] - \sum_{i=1}^{m} r_i\xi_i. \quad (2.25)$$

де α_i *і* r_iє множниками Лагранжа (обмеження ≥ 0).

Пізніше встановлюючи похідні відносно w і b до нуля, як і раніше, замінюючи їх повертаємо в, і, спрощуючи, отримуємо наступну подвійну форму проблема:

$$max_{\alpha}\, W(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j\langle x^{(i)}, x^{(j)}\rangle$$
$$s.t. 0 \leq \alpha_i \leq C, i = 1, \dots, m$$
$$\sum_{i=1}^{m}\alpha_i y^{(i)} = 0, \quad (2.26)$$

Як і раніше, w можна виразити через α_i як зазначено в рівнянні (9), так що після вирішення двоїстої проблеми можна продовжити використовувати рівняння (2.17), щоб робити наші прогнози. Зауважте,

дещо дивно, що додавання l_1 регуляризації, єдина зміна подвійної проблеми це те, що спочатку було обмеженням, що $0 \leq \alpha_i$тепер стало $0 \leq \alpha_i \leq C$.

2.7 Опорні векторні машини для класифікації

У цьому розділі висвітлено деякі базові кроки з опорних векторних машин (SVM) для проблем класифікації. Дано навчальний набір із N точок даних $\{y_k, x_k\}_{k=1}^N$, де $x_k \in \mathbb{R}^n$ k вхідна розмірність даних і $y_k \in \mathbb{R}$ k-й вихідні дані, опорний вектор. Методичний підхід спрямований на побудову класифікатора виду:

$$y(x) = sign\left[\sum_{k=1}^{N} \alpha_k y_k \psi\,(x, x_k) + b\right] \tag{2.27}$$

де α_k — додатні дійсні константи,

b — дійсна константа.

Для ψ(. , .) один зазвичай має такі варіанти: $\psi(x, x_k) = x_k^T$ (лінійний SVM); $\psi(x, x_k) = (x_k^T + 1)^d$ (поліном SVM ступеня d).

Класифікатор будується наступним чином. Наприклад, наявна наступна система рівнянь:

$$\begin{cases} w^T\varphi(x_k) + b \geq 1, & if\ y_k = +1 \\ w^T\varphi(x_k) + b \leq -1, & if\ y_k = -1 \end{cases}$$

що еквівалентно:

$$y[w^T\varphi(x_k) + b] \geq 1,\ k = 1, \ldots, N \qquad (3) \tag{2.28}$$

де $\varphi(.)$ є нелінійною функцією, яка відображає вхідний простір у вищий

мірний простір.

Однак ця функція явно не сконструйована. Щоб мати можливість порушити (2.28), у випадку, якщо розділова гіперплощина в цього вищого вимірного простору не існує, змінні ξ_kвводяться такими

$$\begin{cases} y[w^T\varphi(x_k)+b] \geq 1-\xi_k, & k=1,\dots,N \\ \xi_k \geq 0, & k=1,\dots,N \end{cases} \tag{2.29}$$

Відповідно до принципу мінімізації структурного ризику межа ризику є мінімізується формулюванням задачі оптимізації:

$$min_{w_i\xi_k}\imath_1(w_i\xi_k) = \frac{1}{2}w^Tw + c\sum_{k=1}^{N}\xi_k \tag{2.30}$$

відповідно до (4). Тому будується лагранжіан:

$$\mathcal{L}_1(w,b,\xi;\alpha_k,v_k) = \imath_1(w_i\xi_k) - \sum_{k=1}^{N}\alpha_k\{y_k[w^T\varphi(x_k)+b]-1+\xi_k\} - \sum_{k=1}^{N}v_k\xi_k \tag{2.31}$$

введенням множників Лагранжа$\alpha_k \geq 0, v_k \geq 0$ $(k=1,\dots,N)$. Розв’язок визначається сідловою точкою лагранжіана за допомогою обчислення

$$max_{\alpha_k v_k} min_{w,b,\xi}\mathcal{L}_1(w,b,\xi_k;\alpha_k,v_k) \tag{2.32}$$

можна отримати

$$\begin{cases} \frac{\partial \mathcal{L}_1}{\partial w} = 0 \ \rightarrow w = \sum_{k=1}^{N} \alpha_k y_k \varphi(x_k) \\ \frac{\partial \mathcal{L}_1}{\partial b} = 0 \rightarrow \sum_{k=1}^{N} \alpha_k y_k = 0 \\ \frac{\partial \mathcal{L}_1}{\partial \xi_k} = 0 \rightarrow 0 \leq \alpha_k \leq c, \ \ k = 1, \dots, N \end{cases} \tag{2.33}$$

що призводить до вирішення наступної задачі квадратичного програмування

$$\max_{\alpha_k} Q_1\big(\alpha_k; \varphi(x_k)\big) = -\frac{1}{2} \sum_{k,l=1}^{N} y_k y_i \varphi(x_k)^T \varphi(x_i) \alpha_k \alpha_l + \sum_{k=1}^{N} \alpha_k \tag{2.34}$$

Як

$$\begin{cases} \sum_{k=1}^{N} \alpha_k y_k = 0 \\ 0 \leq \alpha_k \leq c, k = 1, \dots, N \end{cases}$$

Тоді функція $\varphi(x_k)$ у (9) пов’язана з $\psi(x, x_k)$ за

$$\varphi(x)^T \phi(x_k) = \psi(x, x_k) \tag{2.35}$$

яка мотивована теоремою Мерсера. Зауважте, що для двошарових нейрон SVM, умова Мерсера виконується лише для певних значень параметрів к і θ.

Класифікатор (2.27) розроблений розв'язком

$$\max_{\alpha_k} Q_l(\alpha_k; \psi(x_k, x_l)) = -\frac{1}{2} \sum_{k,l=1}^{N} y_k y_i \psi(x_k, x_l) \alpha_k \alpha_l + \sum_{k=1}^{N} \alpha_k \tag{2.36}$$

з урахуванням обмежень у (9). Не потрібно обчислювати ні w, ні $\varphi(x_k)$ в

щоб визначити поверхню рішення. Тому що матриця пов'язана задача квадратичного програмування не є нескінченною, рішення (2.28) буде бути глобальним.

Крім того, можна показати, що гіперплощини (2.25) задовольняють обмеження $||w||_2 \leq a$ а мають VC-розмір h, який обмежений

$$h \leq min([r^2, a^2], n) + 1 \tag{2.37}$$

де [.] позначає цілу частину,

r — радіус найменшої кульки, що містить точки $\varphi(x_1), \dots, \varphi(x_N)$.

Знаходження цієї кулі здійснюється визначенням лагранжіан:

$$\mathcal{L}_2(r, q, \lambda_k) = r^2 - \sum_{k=1}^{N} \lambda_k (r^2 - ||\varphi(x_k) - q||_2^2) \tag{2.38}$$

де q – центр кулі,

λ_k– позитивні множники Лагранжа.

Подібно до (2.27) знаходимо, що центр дорівнює $q = \sum_k \lambda_k \varphi(x_k)$, де множники Лагранжа слідують за f

$$\max_{\lambda_k} Q_2(\lambda_k; \varphi(x_k)) = -\sum_{k,l=1}^{N} \varphi(x_k)^T \varphi(x_l) \lambda_k \lambda_l + \sum_{k=1}^{N} \lambda_k \varphi(x_k)^T \varphi(x_k)$$

Як

$$\begin{cases} \sum_{k=1}^{N} \lambda_k = 1 \\ \lambda_k \geq 0, k = 1, \dots, N \end{cases} \tag{2.39}$$

На основі (2.28) Q_2 також можна виразити через $\psi(x_k, \dots, x_l)$. Нарешті один вибирає опорну векторну машину з мінімальною розмірністю VC шляхом вирішення (2.28) та обчислення (2.29) з (2.31).

Виявилося ефективним для багатьох проблем класифікації. Для класифікацій бінарних класів, конструкції SVM оптимальна розділова гіперплощина між позитивними і негативні класи з максимальним запасом. Це може бути сформульовано як задачу квадратичного програмування.
що включає обмеження нерівності. Найменші квадрати Нещодавно було запропоновано формулювання SVM, яке називається LS-SVM [5, 16], яке включає обмеження рівності тільки.

Отже, розв'язок отримано розв'язуванням системи лінійних рівнянь. Ефективні та масштабовані алгоритми, такі як ті, що базуються на спряженому градієнті, можуть застосовувати для вирішення LS-SVM. Великі емпіричні дослідження показали, що LS-SVM можна порівняти з SVM з точки зору продуктивності узагальнення.

2.8 Адаптивна машина опорних векторів найменших квадратів з матричними входами

Для розглянутої задачі використання опорного вектора ефективною може бути машина (SVM) [4-6], яка оптимізує емпіричний критерій навчання ризику та коригує його параметри базується як на традиційному керованому навчанні, так і на концепція «нейронів у точках даних» [7].

Навчання SVM можна значно прискорити і звести до рішення систем лінійних рівнянь за допомогою так званих найменших квадратів Машини опорних векторів (LS-SVM) [8]. Ці нейроні мережі, які мають лише два шари інформації обробки (прихований шар ядерних функцій активації та вихідний шар регульованих синаптичних ваг) дуже швидкі порівняно з класичними CNN, але зі збільшенням навчання обсяги даних, вони стикаються з обчислювальними проблемами, особливо в режимі онлайн. Деякі з цих проблем можуть бути можна подолати шляхом комбінованого використання групового методу обробки даних (GMDH) та оптимізації емпір критерію ризику [9] або шляхом навчання системи «ковзанню».

вікно» [10]. Збільшення розміру вікна призводить до збільшення обчислювальних труднощів при реалізації мережеве навчання.

Навчання LS-SVM під час масивів зображень обробку можна спростити та прискорити, ввівши додаткова схема для активації першого прихованого шару самонавчання параметрів функцій.

Нехай вхідні дані є набором $x = \{x(1), x(2), x(k), \dots, x(n)\}$, які потрібно розділити на два класи довільної форми та членство кожного спостереження $x(k)$ у наборі даних x де кожен клас відомий апріорі. Тут традиційно передбачається що набір даних x складається з векторів спостережень $x(k) \in R^n$. Оскільки далі розглянемо проблему класифікації зображень, спостереження, у цьому випадку, є матриці $x(k) \in R^{n1xn2}$. Гауссіани з векторним аргументом часто використовуються як функції активації в традиційних SVM і LSSVMa.

$$\varphi_k(x) = exp\left(-\frac{||x - x(k)||^2}{2\sigma^2}\right) \tag{2.40}$$

де $||\circ||$ є символом евклідової норми.

Легко ввести матричне узагальнення такої активаційної функції:

$$\varphi_k(x) = exp\left(-\frac{Tr(x - x(k))(x - x(k))^T}{2\sigma^2}\right) \tag{2.41}$$

Використовуючи норму матриці Фробеніуса замість векторної норми, що узгоджується з нормою евклідового вектора. Нелінійне перетворення, реалізоване SVM, може записуватися в досить загальній формі:

$$\hat{y}(x) = w^T\varphi(x) + w_o \tag{2.42}$$

де $\varphi(x) = (\varphi_1(x), \varphi_2(x), \dots, \varphi_k(x), \dots, \varphi_{N+1}(x))^T$;

(N) – вектор функцій активації ядра;

$w = (w_1, w_2, \dots, w_k, \dots, w_N)^T \in R^n$– вектор налаштованої синаптики ваги;

$w = (w_1, w_2, \dots, w_k, \dots, w_N)^T \in R^n$ – параметр зсуву, також підлягає ідентифікації.

Отже, кількість параметрів мережі, які потрібно ідентифікувати, повністю визначається кількістю спостережень в навчальному наборі. В якості цільової функції використовується квадратичний критерій в LS-SVM:

$$J = \frac{1}{2}||w||^2 + \frac{\gamma}{2}\sum_{k=1}^{N} e^2(k) = \frac{1}{2}(w^T w + y\sum_{k=1}^{N} e^2(k)) \tag{2.43}$$

де $\gamma > 0$– параметр регуляризації,

$e(k) = y(k) - \hat{y}(k) = y(k) - w^t\varphi(k) - w_o$– помилка навчання,

y(k) – зовнішній опорний сигнал) з урахуванням N обмеження:

$$\begin{cases} y(1) = w^T\varphi_1\big(x(1)\big) + w_o + e(1), \\ \vdots \\ y(k) = w^T\varphi_k\big(x(k)\big) + w_o + e(k), \\ \vdots \\ y(N) = w^T\varphi_N\big(x(N)\big) + w_o + e(N) \end{cases} \tag{2.44}$$

Метод невизначених множників Лагранжа [11] використовується для вирішення проблеми оптимізації з урахуванням обмежень. Введення до розгляду функції Лагранжа

$$L(w, w_o, e, \lambda) = J + \sum_{k=1}^{N} \lambda(k)(y(k) - w^T \circ \varphi_k\big(x(k) - w_o e(k)\big)) \tag{2.45}$$

де $\lambda(k)$– невизначені множники Лагранжа і розв’язання системи рівнянь Куна-Таккера [12]:

$$\begin{cases} \nabla_w L(w, w_o, e, y) = w - \sum_{k=1}^{N} \lambda\,(k)\varphi_k\big(x(k)\big) = \overline{O_N} \\ \frac{\partial L(w,w_o,e,y)}{\partial w_o} = -\sum_{k=1}^{N} \lambda(k) = O, \\ \frac{\partial L(w,w_o,e,y)}{\partial e(k)} = \gamma e(k) - \lambda(k) = O, \\ \frac{\partial L(w,w_o,e,y)}{\partial \lambda(k)} = y(k) - w^T \varphi_k\big(x(k)\big) - w_o - e(k) = O, \end{cases} \tag{2.45}$$

отримуємо систему простих співвідношень

$$\begin{cases} w = \sum_{k=1}^{N} \lambda(k)\varphi_k\big(x(k)\big), \\ \sum_{k=1}^{N} \lambda(k) = 0, \\ \lambda(k) = \gamma e(k), \\ y(k) = w^T \varphi_k\big(x(k)\big) + w_o + e(k) \end{cases} \tag{2.46}$$

або в компактному вигляді:

$$\begin{pmatrix} 0 & I_N^T \\ I_N & \Omega(N) + \gamma^{-1} I_{NN} \end{pmatrix} \begin{pmatrix} w_o \\ \Lambda(N) \end{pmatrix} = \begin{pmatrix} 0 \\ Y(N) \end{pmatrix} \tag{2.47}$$

де $0 \in R^N$– вектор, що складається з нулів;

$I_N \in R^N$ – вектор що складається з одиниць;

$\Lambda(N) = (\lambda(1), \lambda(2), \dots, \lambda(k), \dots, \lambda(N))^T, I_{N^\circ N} - (N + N)$ – матриця, що складається з одиниць;

$\Omega(\mathrm{N}) = \{\Omega_y = \varphi_i^T\big(x(i)\big)\varphi_i\big(x(j)\big) = \Phi\big(x(i), x(j)\big)\}\ \Phi(x(i), x(j))$– ядерна функція у вигляді:

$$\begin{cases} \begin{pmatrix} w_o \\ \Lambda(N) \end{pmatrix} = \begin{pmatrix} 0 & I_N^T \\ I_N & \Omega(N) + \Upsilon^{-1} I_{NN} \end{pmatrix} \begin{pmatrix} 0 \\ Y(N) \end{pmatrix}, \\ \hat{y}(x) = \sum_{k=1}^{N} \lambda(k) \Phi(x, x * k)) + w_o, \\ \Phi\big(x, x(k)\big) = exp\left(-\frac{Tr(x - x(k)(x - x(k))^T}{2\sigma^2} \right) \end{cases}$$

(2.48)

Якщо спостереження з навчального набору надходять послідовно, тобто цей набір даних постійно зростає $x(N+1), x(N+2), ...,$, обчислювальні проблеми, пов'язані з необхідністю також інвертувати матрицю, утворену функціями ядра на кожному кроці збільшуються. Спростити процес вирішення можна за допомогою установки центрів активації функціонують аналогічно процесу тренування дзвоноподібної функції активації [13] а імовірнісна нейронна мережа [14].

Припустимо, що на основі навчального набору X містить N спостережень була синтезована матриця LS-SVM, що містить $\mathrm{N}+1$ синаптичних ваг, які були визначені шляхом розв’язування систему рівнянь (2.48). Нехай навчальний набір отримує а нове спостереження$x(N+1)$ з відомою приналежністю до а конкретний клас. Далі в початковому навчальному наборі знаходимо спостереження x(k) найближче до:

$$x(N+1) \tag{2.49}$$

сенс метрики Фробеніуса: далі позначення спостереження є $x^*(k)$– «переможець» за Т. Кохоненом [15]. Після що, використовуючи модифікацію матриці [16,17] правила самонавчання «переможець отримує все» (WTA), «підтягуємо» спостереження за переможцем $x^*(k)$до нового спостереження

$$x(N+1) \tag{2.50}$$

відповідно до процедури

$$\tilde{x}(k) = x^*(k) + \eta(k)(x(N+1) - x^*(k)) \tag{2.51}$$

де $0 \leq \eta(k) \leq 1$– параметр швидкості навчання.

В результаті, замість функції активації $\varphi_k(x)$, нова функція $\varphi_k(x)$з центром матриці

$$\tilde{x}(k) \tag{2.52}$$

формується. Варто зауважити також, що якщо $\eta(k) = 0$ спостереження $x(N+1)$ є ігноруються, тобто синаптичні ваги мережі не ігноруються налаштований. Якщо $\eta(k) = 1$ з центром $x^*(k)$ замінюється на нове спостереження $x(N+1)$. При цьому кількість синаптичні ваги дорівнюють $N+1$ залишається незмінним у система.

У процесі формування навчального набору може виникнути ситуація виникають при новому спостер $x(N+1)$ і "переможець" $x^*(k)$ належать до різних класів.

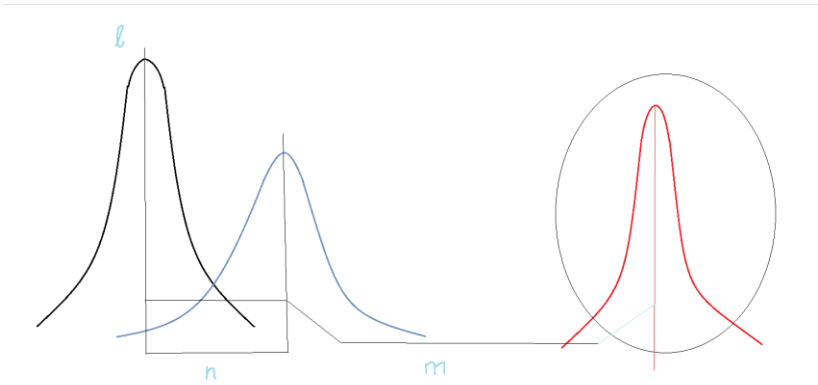


Рисунок 2.6 – Приклад «виштовхування» классу

У цьому випадку можна використовувати техніка векторного квантування навчання [18,19], яка є а узагальнення правила навчання WTA Т. Кохонена для навчання під наглядом. У той же час, якщо $x^*(k)$ і $x(N+$

1) належать до одного класу, центри скориговані згідно (2.51). Якщо вони належать до різних класів, то переможець «виштовхується». $x(N+1)$, і новий центр с функція активації формується за правилом вектора квантування

$$\tilde{x}(k) = x^*(k) - \eta(k)(x(N+1) - x^*(k)) \qquad (2.53)$$

Цей підхід до навчання LS-SVM за умов коли дані надходять в онлайн-режимі, як дозволяє потік істотно спростити і прискорити налаштування зображення система розпізнавання.

3 ОПИС СИСТЕМИ І ЕКСПЕРИМЕНТАЛЬНА ПЕРЕВІРКА ОТРИМАНИХ НАУКОВИХ РЕЗУЛЬТАТІВ

Ключовим моментом у вирішенні задачі налаштування параметрів алгоритму опорних векторів є вибір критерію якості одержуваних рішень. Відомим і активно розвивається підходом для вибору лінійних параметрів оптимальної складності є використання так званих зовнішніх критеріїв. У нашому випадку як такі можуть бути використані різні варіанти критеріїв, пов'язаних з точністю прогнозу на тестовій вибірці У цій роботі досліджується можливість розбиття вибірки на навчальну та тестову частини із залученням методів оптимального планування експерименту

3.1 Застосування методу опорних векторів

SVM корисні для категоризації тексту та гіпертексту, оскільки їх застосування може значно зменшити потребу в позначених екземплярах навчання як у стандартних індуктивних, так і в трансдуктивних налаштуваннях. Деякі методи неглибокого семантичного аналізу базуються на опорних векторних машинах.

Класифікацію зображень також можна виконувати за допомогою SVM. Експериментальні результати показують, що SVM досягають значно вищої точності пошуку, ніж традиційні схеми уточнення запитів, лише після трьох-чотирьох раундів відгуків про релевантність. Це також вірно для систем сегментації зображень, у тому числі тих, що використовують модифіковану версію SVM, яка використовує привілейований підхід, запропонований Вапником.

Класифікація супутникових даних, таких як дані SAR , з використанням контрольованої SVM.

Рукописні символи можна розпізнати за допомогою SVM.

Алгоритм SVM широко застосовується в біологічних та інших науках. Вони використовувалися для класифікації білків, до 90% сполук класифіковано правильно. Тести на перестановку на основі ваг SVM були запропоновані як механізм інтерпретації моделей SVM. Вагові коефіцієнти опорних векторних машин також використовувалися для інтерпретації моделей SVM у минулому. Постфакторна інтерпретація машинних моделей опорних векторів з метою ідентифікації особливостей, які використовуються моделлю для прогнозування, є відносно новою областю досліджень, яка має особливе значення в біологічних науках.

3.2 Структура системи

У багатьох реальних програмах, таких як прогнозування, класифікація та регресія, важливо створювати точні та ефективні моделі, які можуть обробляти шумні та великі набори даних. Традиційний алгоритм SVM широко використовується для таких додатків, але він може бути обчислювально дорогим і чутливим до вибору функції ядра. Щоб усунути ці обмеження, алгоритм найменших квадратів опорної векторної машини (LSSVMa) був запропонований як альтернатива, яка може забезпечити швидше навчання та кращу обробку зашумлених даних. Постановка проблеми полягає в розробці моделі LSSVMa, яка може точно передбачити або класифікувати цільову змінну для даного набору даних, одночасно мінімізуючи витрати на обчислення та впораючись із шумом у даних.

Метою алгоритму LSSVMa є створення точної моделі, яка може передбачити або класифікувати цільову змінну даного набору даних. Набір даних може бути шумним і містити велику кількість функцій і зразків, що може ускладнити побудову точної моделі. Алгоритм LSSVMa використовує функцію втрат за методом найменших квадратів, щоб

мінімізувати помилку регресії або помилку класифікації, і він може обробляти шумові дані краще, ніж традиційний алгоритм SVM.

Щоб створити модель LSSVMa, зазвичай виконуються наступні кроки:

1) попередня обробка даних: може знадобитися попередня обробка даних для обробки відсутніх значень, викидів або перетворення даних у більш придатний формат для алгоритму LSSVMa;

2) вибір функцій: Можливо, потрібно буде вибрати функції, які найбільше відповідають цільовій змінній, щоб зменшити розмірність набору даних і покращити продуктивність моделі;

3) вибір моделі: вибір функції ядра та параметра регуляризації може мати значний вплив на продуктивність моделі LSSVMa. Можна використовувати кілька ядерних функцій, таких як лінійні, поліноміальні та радіальні базисні функції, а параметр регуляризації можна вибрати за допомогою перехресної перевірки або інших методів вибору моделі;

4) навчання моделі: модель LSSVMa навчається за допомогою вибраної функції ядра та параметра регулярізації. Задача оптимізації формулюється як система лінійних рівнянь, які можна розв’язати аналітично за допомогою методів лінійної алгебри;

5) оцінка моделі: продуктивність моделі LSSVMa оцінюється за допомогою відповідних показників оцінювання, таких як середня квадратична помилка для проблем регресії або точність, точність, пригадування та бал F1 для проблем класифікації;

Постановка проблеми для LSSVMa полягає в тому, щоб розробити точну та ефективну модель, яка може обробляти шумні та великі набори даних, одночасно мінімізуючи обчислювальні витрати та чутливість до вибору функції ядра. Цю постановку проблеми можна застосувати до багатьох реальних програм, таких як прогнозування цін на акції, класифікація медичних зображень або виявлення шахрайства у фінансових операціях.

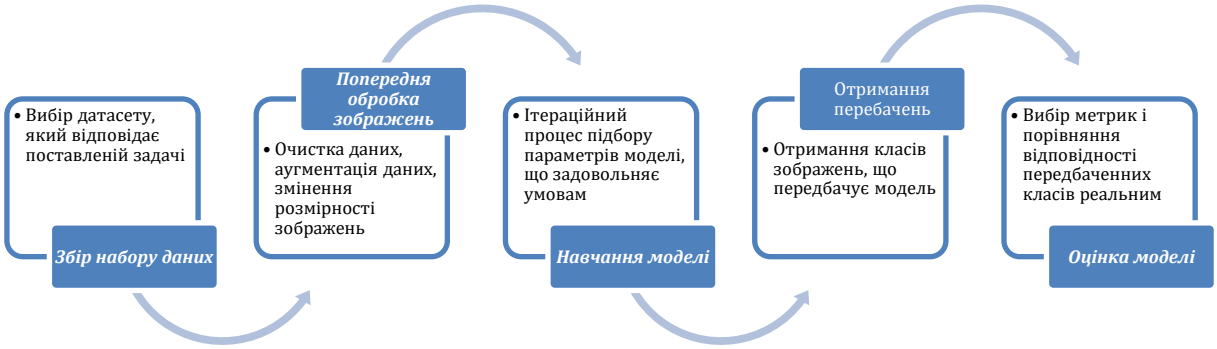


Рисунок 3.1 – Процес розпізнавання зображень

3.3 Опис набору даних

Для задачі класифікації був взятий набір даних фруктів. Загальна кількість зображень: 22495. Розмір навчального набору: 16854 зображення (один фрукт або овоч на зображення). Розмір тестового набору: 5641 зображення (один фрукт або овоч на зображення). Кількість класів: 33 (фрукти та овочі). Розмір зображення: 100x100 пікселів. Для аналізу порівняємо різні алгоритми класифікації, зокрема SVM, K-NN, Decision Tree, CNN спочатку для задачі бінарної класифікації, потім для мультикласової. Я також застосую аналіз основних компонентів, щоб зменшити розмірність набору даних, побачити дисперсію кожного класу, а потім спробую застосувати алгоритм класифікації, що має лише два виміри.

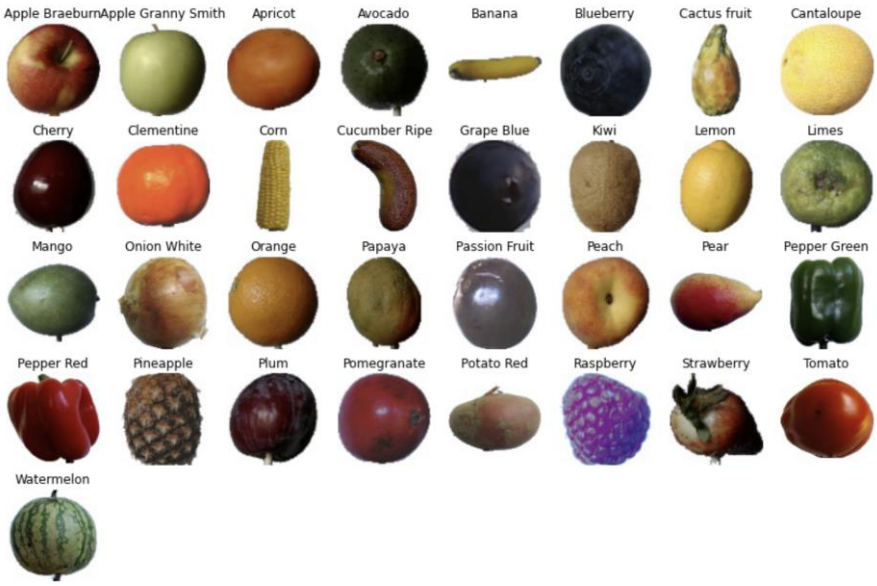


Рисунок 3.2 – Датасет

Кожне зображення перетворюється в масив numpy 100x100 для кожного розміру RGB (x3). Потім було зведено в один вектор (вектор характеристик зображення), а потім масштабовано, віднімаючи середнє значення набору даних, щоб виконати алгоритми класифікації.

Рисунок 3.3 – Варіація класу в датасеті

На зображенні нижче останньою матрицею є X_train із формою 30000x980.

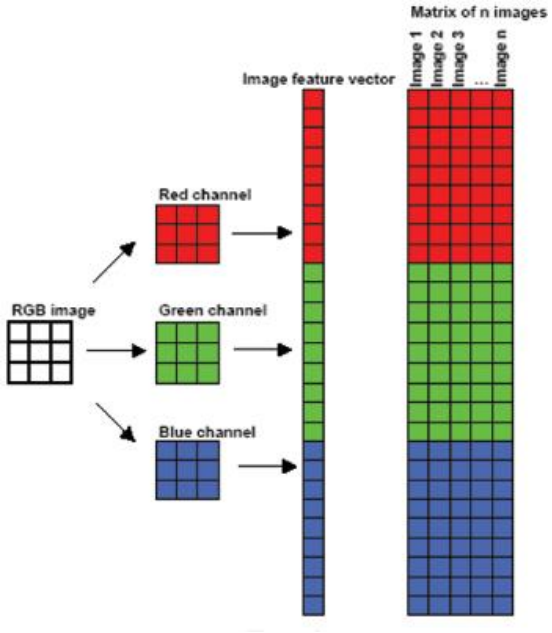


Рисунок 3.4 – Підготовка малюнку

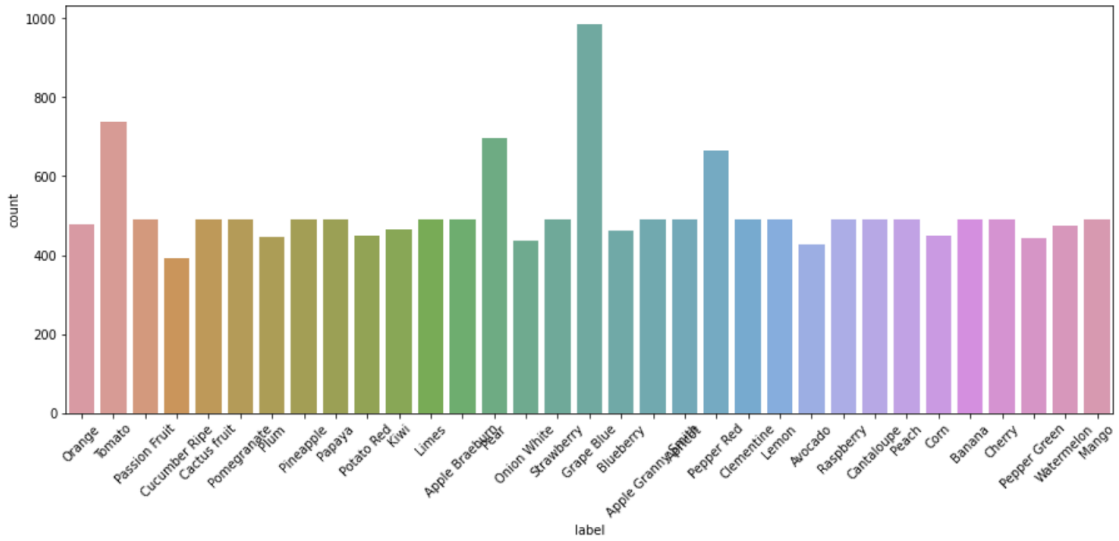


Рисунок 3.5 – Розподіл класів в датасеті

Щоб дізнатися, як наші дані виглядають у нижчому вимірі, потрібно зменшити розмірність набору даних у 2 або 3 вимірі, щоб була змога побудувати та візуалізувати їх. Для цього можна використати аналіз головних компонентів, але кращим рішенням може бути використання t-

SNE (T-розподілене стохастичне вбудовування сусідів) або MDS (багатовимірне масштабування), які добре підходять для нелінійних методів зменшення розмірності. – підходить для вбудовування високовимірних даних для візуалізації в низьковимірний дво- або тривимірний простір.

t-SNE – це те, що називається нелінійним зменшенням розмірності . t-SNE є чудовим інструментом для розуміння масивів даних великої розмірності. Це може бути менш корисним, якщо ви хочете виконати зменшення розмірності для навчання ML (не можна повторно застосувати таким же чином). Він не є детермінованим і не повторюється, тому щоразу, коли він виконується, він може давати інший результат. Але навіть з цими недоліками він все ще залишається одним із найпопулярніших методів у цій галузі.

З огляду на ці графіки не можна зробити припущення про лінійність набору даних.

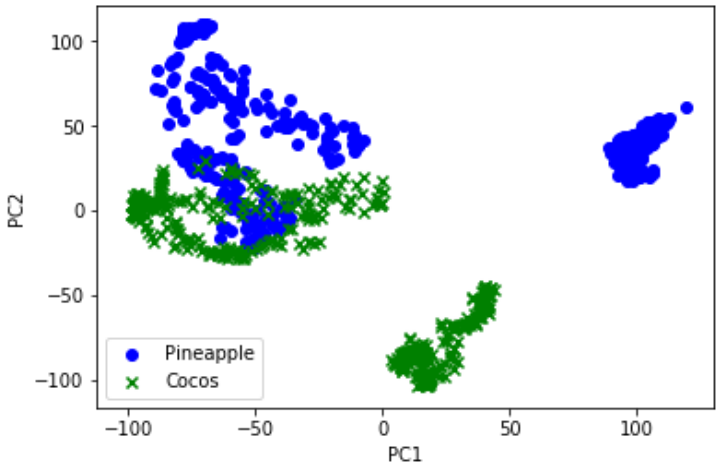


Рисунок 3.6 - Дані в двовимірному просторі

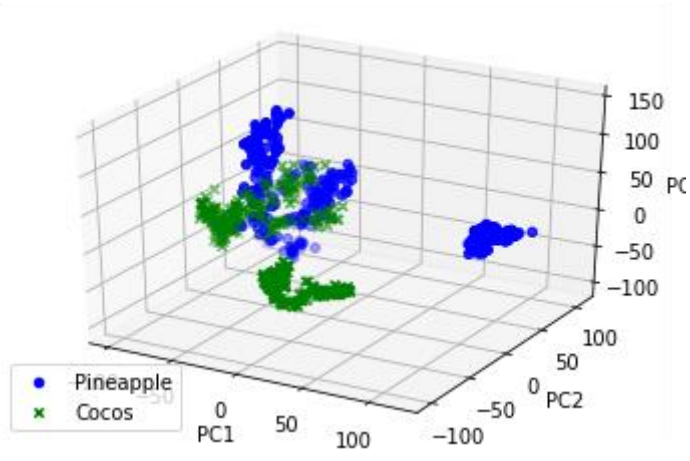


Рисунок 3.7– Дані в тривимірному просторі

3.4 Попередня обробка даних

Аналіз основних компонентів — це техніка, яка використовується для зменшення розмірності набору даних, зберігаючи якомога більшу кількість інформації. Дані повторно проектуються в нижчому вимірному просторі, зокрема потрібно знайти проекцію, яка мінімізує квадрат помилки при реконструкції вихідних даних.

Аналіз головних компонентів, або PCA, — це метод зменшення розмірності , який часто використовують для зменшення розмірності великих наборів даних шляхом перетворення великого набору змінних у менший, який все ще містить більшу частину інформації у великому наборі.

Зменшення кількості змінних у наборі даних природно відбувається за рахунок точності, але хитрість у зменшенні розмірності полягає в тому, щоб поміняти невелику точність на простоту. Оскільки менші набори даних легше досліджувати та візуалізувати, а також зробити аналіз даних набагато простішим і швидшим для алгоритмів машинного навчання без сторонніх змінних для обробки.

Отже, підсумовуючи, ідея PCA проста — зменшити кількість змінних у наборі даних, зберігаючи якомога більше інформації.

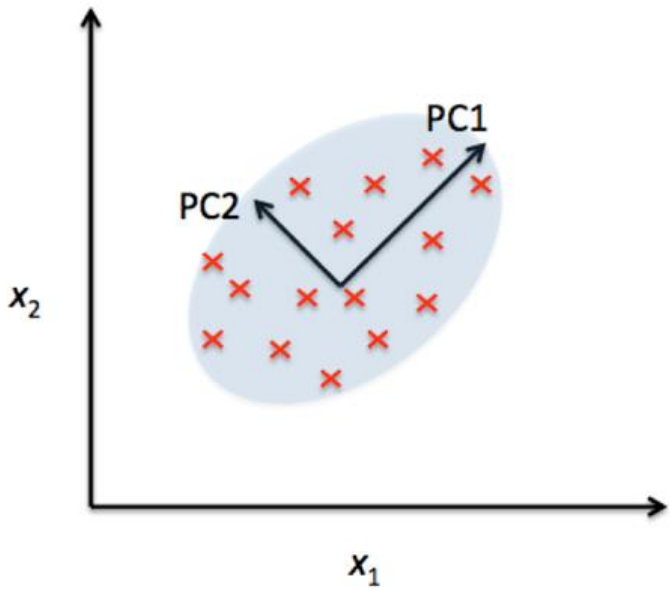


Рисунок 3.8 – Приклад PCA

Існує 3 різні техніки застосування PCA:

– послідовний;
– зразок коваріаційної матриці;
– декомпозиція сингулярного значення (SVD).

Перше, що потрібно зробити, це стандартизувати дані, тому для кожної вибірки потрібно відняти середнє значення повного набору даних, а потім розділити його на дисперсію, щоб отримати одиничну дисперсію для кожного екземпляру. Цей останній процес не є абсолютно необхідним, але він корисний, щоб дозволити ЦП працювати менше.

$$Z = \frac{X - \mu}{\sigma^2} \tag{3.6}$$

Потім потрібно обчислити коваріаційну матрицю за даними $\{x_1, x_2 \dots x_n\}$ з n кількість вибірок, коваріаційна матриця отримана за допомогою:

$$\sum = \frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})(x - \overline{x})^T$$

де

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

(3.7)

Або просто помноживши стандартизовану матрицю Z на її самотранспоновану:

$$COV(X) = ZZ^T \quad (3.8)$$

Головними компонентами будуть власні вектори коваріаційної матриці, відсортовані в порядку важливості за відповідними власними значеннями.

Більші власні значення ⇒ більш важливі власні вектори.

Вони представляють більшість корисної інформації про весь набір даних в одному векторі.

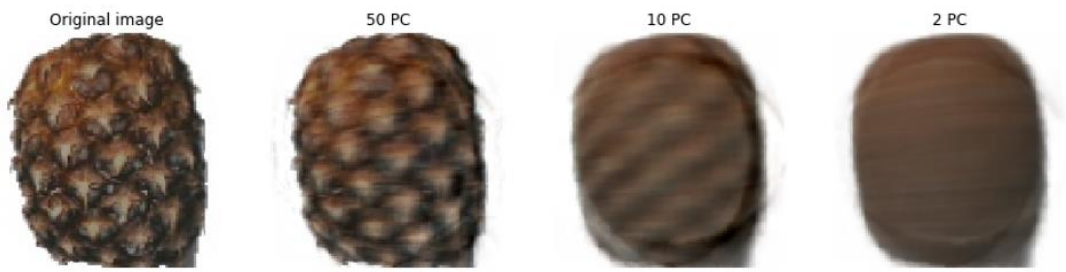


Рисунок 3.9 – Приклад PCA

З цих зображень можна зрозуміти, як, беручи до уваги лише зразок і отримуючи його основні компоненти з цілого набору даних, можна легко класифікувати плід, беручи до уваги лише низький числовий вимір замість усіх. Це означає набагато менше даних.

Очевидно, що для алгоритму класифікації точність класифікації буде нижчою, але замість цього час навчання буде швидшим, якби класи були лінійно роздільними, точність могла б бути задовільною.

Пояснення варіантів за використанням PC

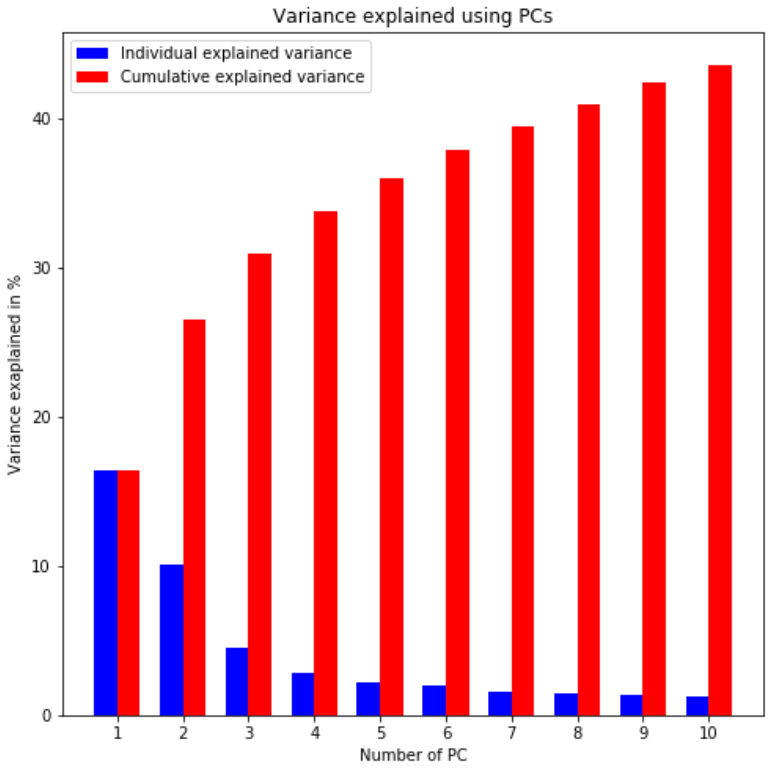


Рисунок 3.10 – Дисперсія, що пояснюється PCA

Як видно з цього графіка, перший PC фіксує лише 16-17% дисперсії, якщо використовувати перші два, відсоток отриманої інформації становить менше 30%.

У наступному розділі я також проведу класифікацію, використовуючи лише перші два виміри, це не для отримання хорошої

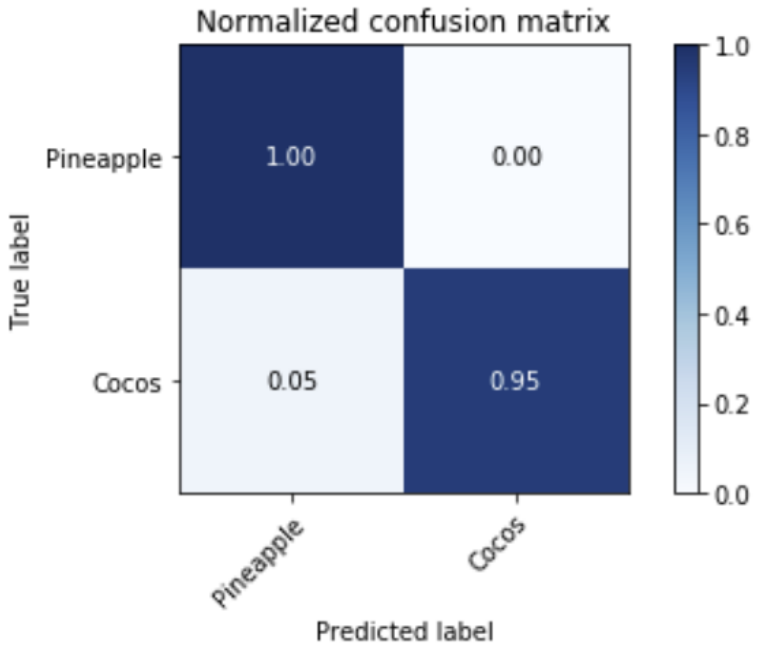
точності, тому що з таким низьким рівнем інформації класифікатори не зможуть дізнатися так багато, натомість зроблено для того, щоб показати межі прийняття рішень класифікаторів при використанні двох вимірів.

Використовуючи перші 10 PC, отримана дисперсія становить 40%, непоганий результат, враховуючи, що набір даних має сотні вимірів.

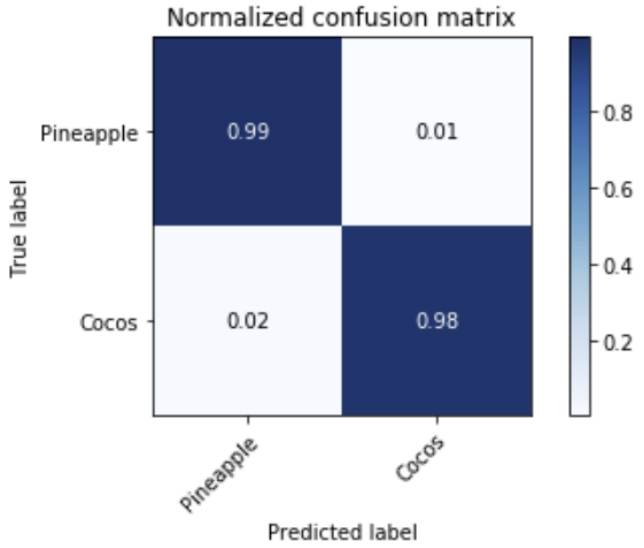
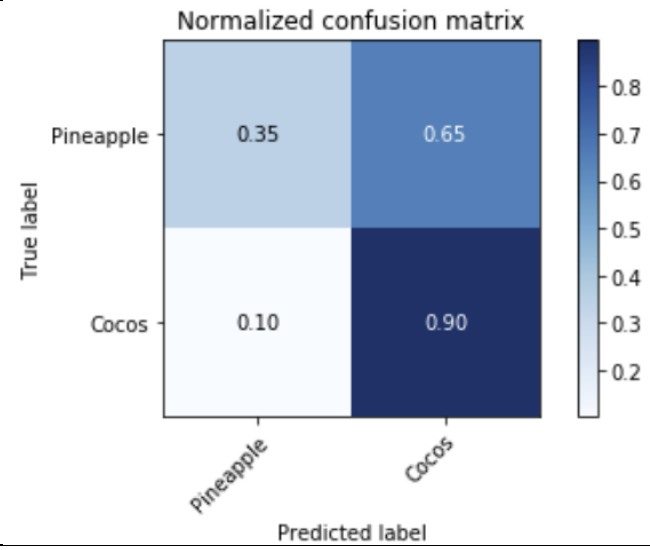
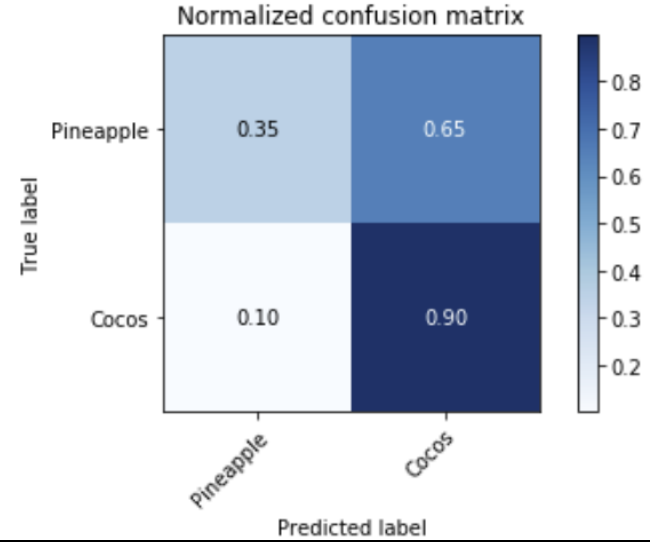
3.5 Порівняльний аналіз результатів експерименту

Метод опорних машин є досить ефективним алгоритмом для класифікації, на матриці плутанини можна помітити, що алгоритм може с високою точністю розмежовувати бінарні класи (Табл 3.1).

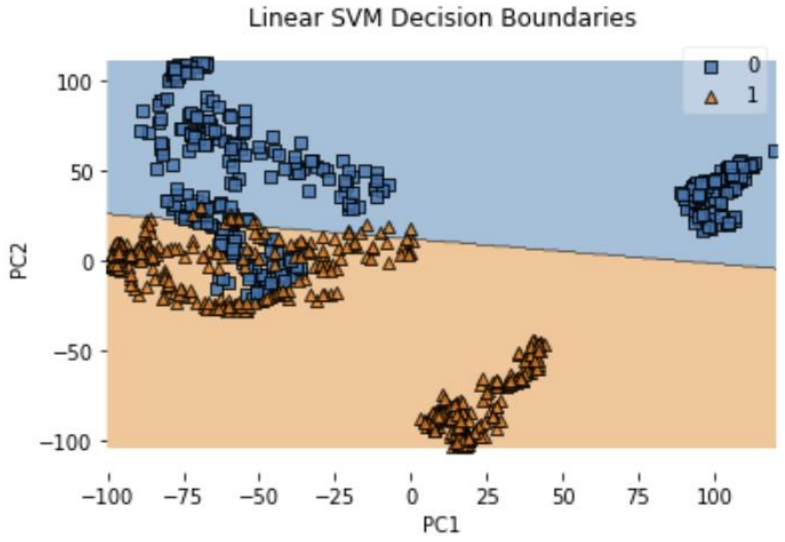
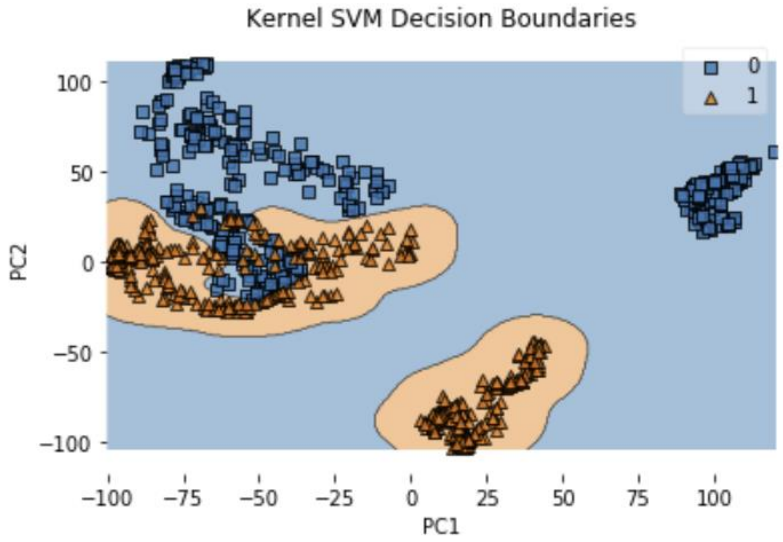
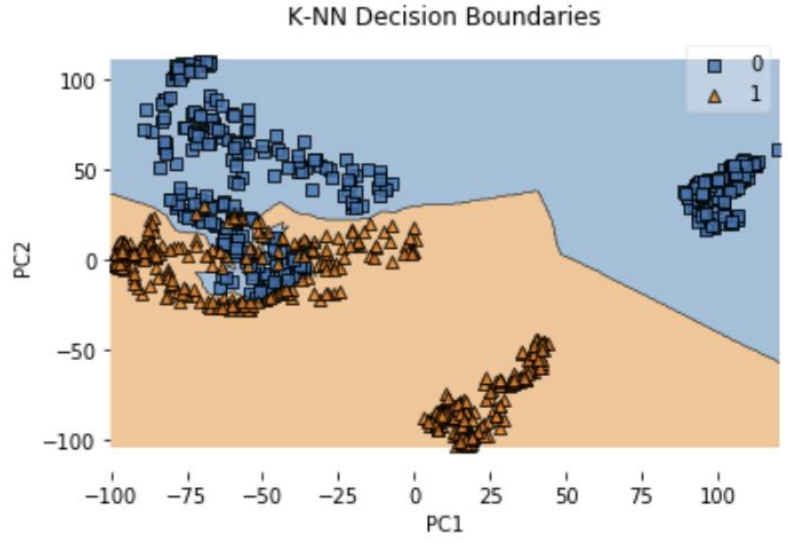
Таблиця 3.1 – Коваріаційна матриця бінарної класифікації

Алгоритм	Коваріаційна матриця	Точність
Linear SVM		97.59%

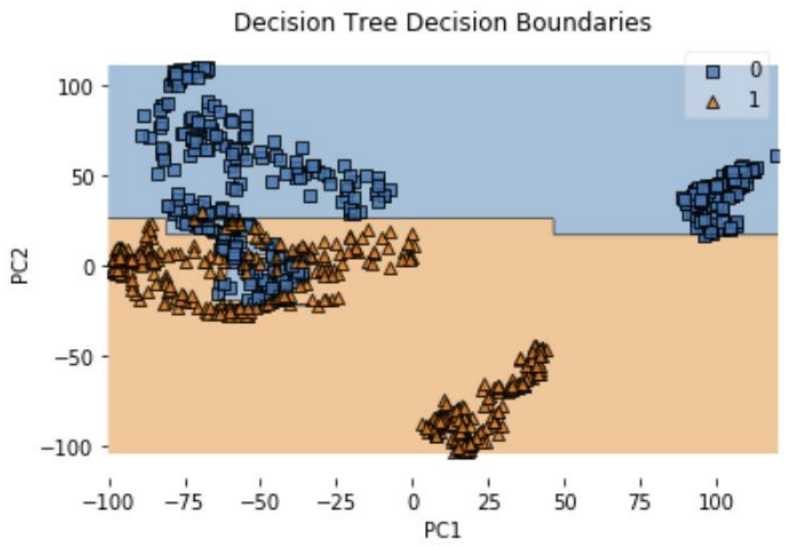
Продовження таблиці 3.1 – Коваріаційна матриця бінарної класифікації

Алгоритм	Коваріаційна матриця	Точність
KNN		98.80%
Decision Tree		62.35%
CNN		65.84%

Таблиця 3.2 – Розподіл точок даних з PCA

Алгоритм	Розподіл з PCA	Точність
LINEAR SVM + PCA		13.55%
KERNEL SVM + PCA		54.52%
KNN-PCA		13.86%

Продовження таблиці 3.2 – Розподіл точок даних з PCA

Алгоритм	Розподіл з PCA	Точність
Decision Tree - PCA		12.95%

Визначення K для алгоритму найближчих сусідів також вимагає експериментальної оцінки.

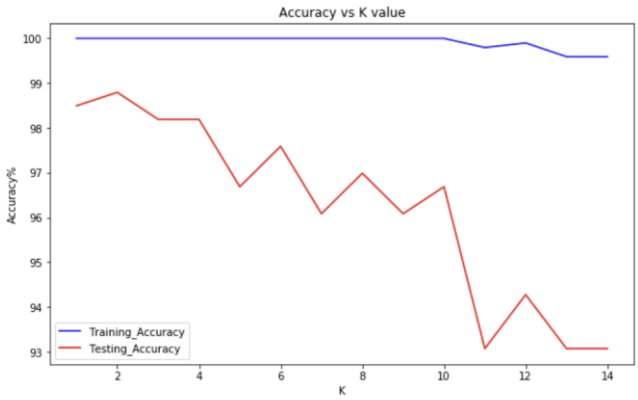


Рисунок 3.11 – Розподіл K найближчих сусідів

З цього графіка можна зрозуміти, як найкраще значення K дорівнює 2, оскільки точність тесту досягає найкращого показника точності, а потім починає зменшуватися.

Точність навчання все ще підтримує 100% точність, починаючи зі зменшення для останніх чисел K.

3.6 Оцінка моделі

Щоб знайти найбільш підходящий алгоритм для цього набору даних, будуть представлені різні методи оцінки: точність, матриця плутанини, крива ROC.

Щоб побудувати криву робочих характеристик приймача (ROC), потрібно обчислити TPR і FPR і вибрати ряд порогових значень для класифікації (AUG). Площа під кривою ROC, виконана графіка TPR і FPR використовується як матриця оцінки для різних класифікаторів

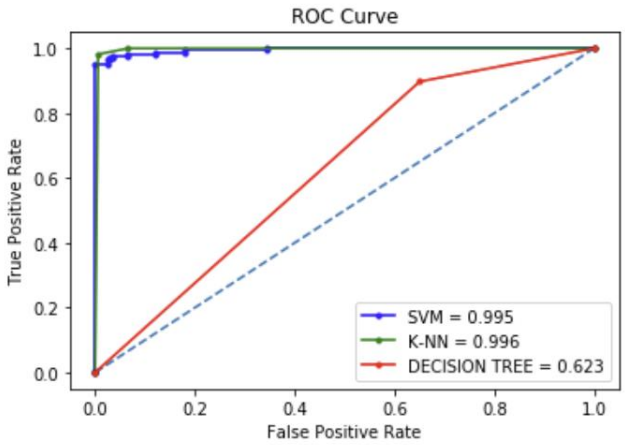


Рисунок 3.12 – ROC крива

Таблиця 3.3 – Точність алгоритмів для багато класової задачі

Алгоритм	Точність
SVM	96.46%
KNN	95.12%
Decision Tree	77.57%
LSSVM	98.12%
CNN	86.78%

Таблиця 3.4 – Швидкодія передбачень алгоритмів для багато класової задачі з різними розмірами пакетів

Алгоритм	Розмір пакету	Швидкість (секунди)
SVM	8	0.03
	16	0.09
	32	0.1
KNN	8	0.046
	16	0.12
	32	0.41
Decision Tree	8	0.52
	16	0.67
	32	0.871
LSSVMa	8	0.012
	16	0.028
	32	0.035
CNN	8	0.789
	16	0.892
	32	1.382

Як бачимо, алгоритм LSSVMa працює краще за інших, тепер розглянемо реззультати з примінення різних ядер для алгоритму.

Таблиця 3.5 – Приклад використання ядер LSSVMa

Ядро	Точність
Гаусс	98.80%
Поліноміальне	95.44%
Лінійне	97.34%

Побудуємо коваріаційну матрицю з використанням ядер поліноміального і лінійного.

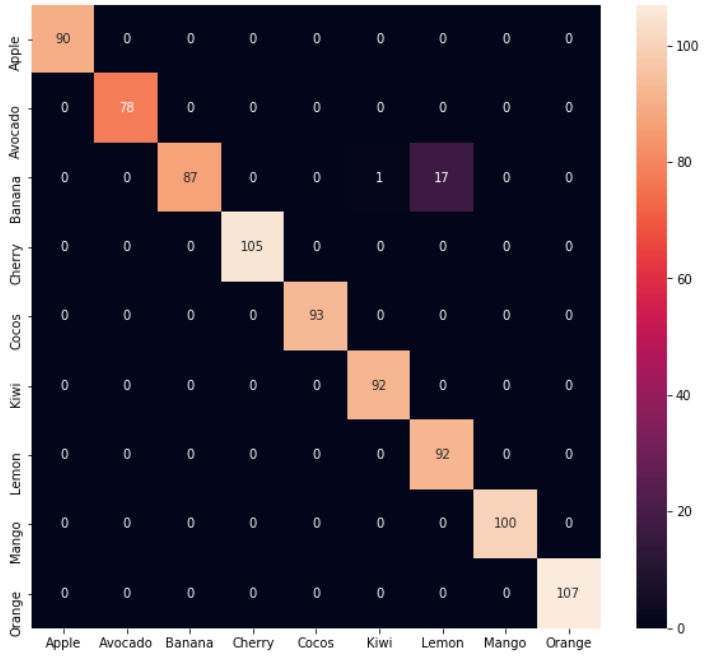


Рисунок 3.13 – Коваріаційна матриця для поліноміального ядра

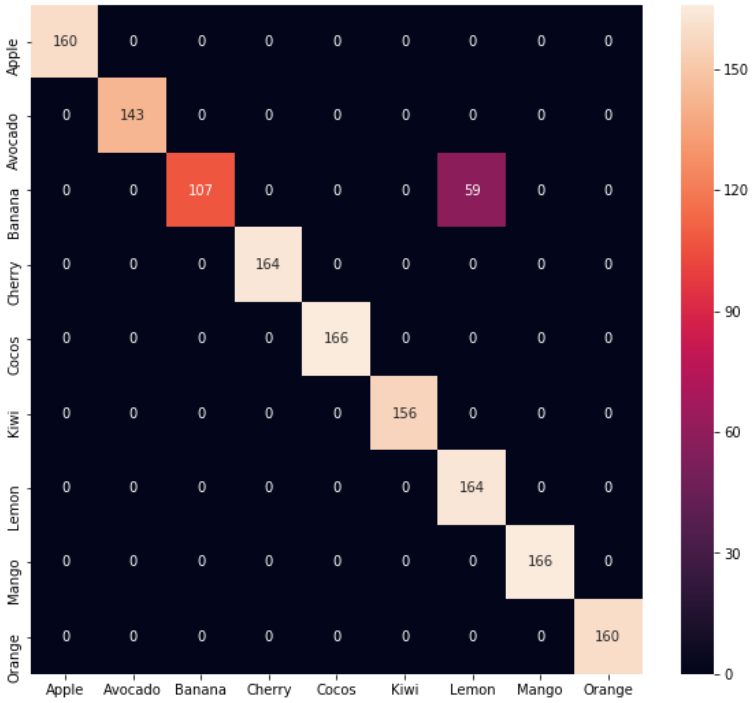


Рисунок 3.14 – Коваріаційна матриця для лінійного ядра

Варто також звернути увагу на результати нейромережі

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 98, 98, 5)         140
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 49, 49, 5)         0
_________________________________________________________________
flatten (Flatten)            (None, 12005)             0
_________________________________________________________________
dense (Dense)                (None, 33)                396198
=================================================================
Total params: 396,338
Trainable params: 396,338
Non-trainable params: 0
_________________________________________________________________
```

Рисунок 3.15 – Архітектура побудованої нейромережі

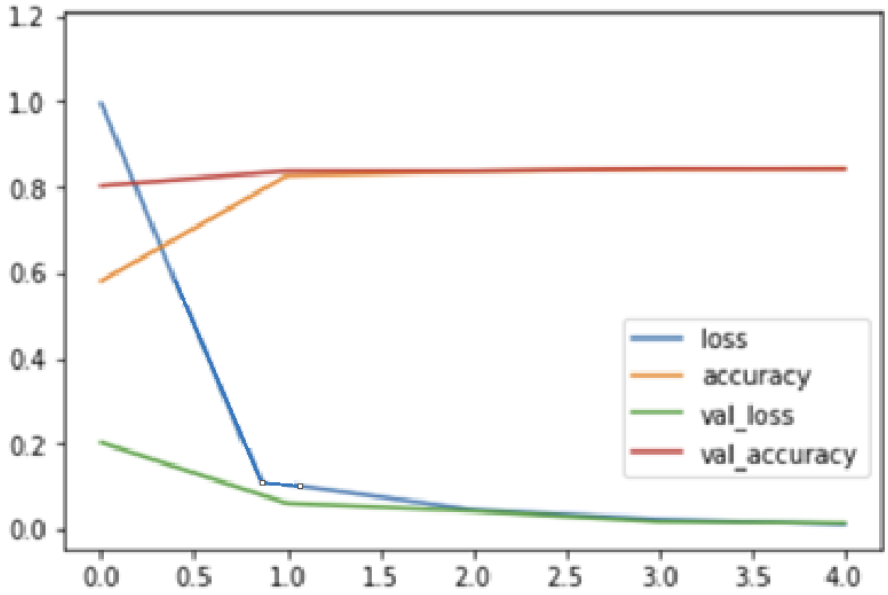


Рисунок 3.16 – Результати кожної ітерації навчання нейромережі

Як бачимо з результатів порівняльного аналізу, запропонований підхід працює краще за інших в умовах обмеженності даних і вимірювальних ресурсів. Для більш детальної оцінки варто також провести

аналіз на декількох датасетах, оскільки складність зображень може нести великий вплив на результат. Варто також виділити декілька переваг запропонованого підходу і методу опорних машин вцілому: система працює краще, коли між класами є чітка межа поділу; ефективно використовує пам’ять, значно швидше працює ніж існуючі методи для класифікації .

Незважаючи на це існують і неякі недоліки такого алгоритму: алгоритм не підходить для великих наборів даних; у випадках, коли кількість функцій для кожної точки даних перевищує кількість навчальних зразків даних, алгоритм працюватиме погано;

ВИСНОВКИ

В цій магістерській атестаційний роботі розроблено систему розпізнавання зображень, яка працює без попереднього перетворення зображень у векторну форму. Система базується на машині опорних векторів за методом найменших квадратів з матричними входами, де знаходяться центри активації функцій формуються безпосередньо зображеннями навчального набору. Система налаштування реалізовано на основі комбінованого навчання, яке включає традиційне навчання під наглядом, «ліниве» навчання на основі концепції «нейронів у точках даних».

Запропонована система здатна обробляти інформацію, що надходить послідовно в режимі онлайн. Отже, цей метод призначений для вирішення досить великого класу завдань в загальних рамках видобутку великих даних.

Визначення функції приналежності та системи правил є важливим з огляду на надійність моделі і точність. На відміну від моделей штучної нейронної мережі, представлена модель використовують лише невелику кількість історичних даних і забезпечує підвищення точності розпізнавання зображень.

Система може використати актуальну інформацію останню по часу, а вибірки мають невеликий об’єм. Запропонований метод обробки зображень розрахований на навчальні вибірки середнього і малого розміру, що дає можливість використати цей метод для оперативного аналізу в режимі онлайн.

В результаті експериментального перевірки запропонованих теоретичних результатів отримані високі показники точності, які щонайменше перевищують існуючі методи на базі дерев рішень і найближчих сусідів на 3-5%, з використанням матричного методу опорних векторів з адаптивним комбінованим навчанням активаційної функції,

також було проведено аналіз швидкості отримання передбачень, який показав підвищення швидкості на 0.2 секунди в прогнозуванні результатів на невеликих пакетах.

Запропонований підхід можна використовувати для виявлення шахрайства, щоб ідентифікувати та позначати шахрайські транзакції в режимі реального часу; в системах рекомендацій для підвищення точності персоналізованих рекомендацій, коли стають доступними нові дані користувача; в автономних транспортних засобах для постійного оновлення моделі новими даними, оскільки транспортний засіб збирає дані датчиків під час роботи.

Загалом матричний метод опорних векторів з адаптивним комбінованим навчанням активаційної функції можна використовувати в багатьох додатках, де дані надходять безперервно або потоками, а модель потрібно оновлювати в режимі реального часу, щоб надавати точні прогнози чи розуміння.

ПЕРЕЛІК ДЖЕРЕЛ І ПОСИЛАНЬ

1. Aggarwal C. C. Neural Networks and Deep Learning. Cham. Springer International Publishing, 2018. 174 p.
2. Bengio Y., Courville A., Goodfellow I. Deep Learning. MIT Press, 2016. 274 p.
3. Bishop C.M., Neural networks for pattern recognition, Oxford University Press, 1995. Cherkassky V., Mulier F., Learning from data: concepts, theory and methods, John Wiley and Sons, 1998.
4. Bodyanskiy Y., Deineko A., Brodetskyi F., and Kosmin D., "Adaptive least-squares support vector machine and its online learning," CEUR Workshop Proceedings, vol. 2762, Nov. 2020, Art. no. 3.
5. Bodyanskiy Y., V. Volkova, I. Pliss and, O. Boiko, “Matrix deep neural network and its rapid learning in Data Science tasks",” in Proceedings of the International Conference “Advanced Computer Information Technologies,” 2018, pp. 141-144,.
6. Bodyanskiy Y., Volkova V. та Skuratov M., "Matrix Neuro-Fuzzy Self-Organizing Clustering Network", Scientific Journal of Riga Technical University. Computer Sciences, т. 45, № 1, p. 54–58, 2011.
7. Cortes C. and Vapnik V., "Support-vector networks," Machine Learning, vol. 20, no. 3, pp. 273–297, Sep. 1995.
8. Fletcher R., Practical methods of optimization, Chichester and New York: John Wiley and Sons, 1987.
9. Golub G.H., Van Loan C.F., Matrix Computations, Baltimore MD: Johns Hopkins University Press, 1989.
10. Graupe D., Deep Learning Neural Networks. WORLD SCIENTIFIC, 2016.
11. Gray R., "Vector quantization", IEEE ASSP Magazine, т. 1, № 2, p. 4–29, квіт. 1984
12. Haykin S., Neural Networks: a Comprehensive Foundation, Macmillan College Publishing Company: Englewood Cliffs, 1994.
13. Kohonen, "Improved versions of learning vector quantization", 1990 IJCNN International Joint Conference on Neural Networks, San Diego, CA, USA, 1990.
14. Libenti J.C., Rappoport T.S. Smart Antennas for Wireless Communications: IS-95 and Third-Generation CDM Applications, Prentice Hall, HJ, USA, 1999. 101 p.R. Gray, "Vector quantization",

IEEE ASSP Magazine, т. 1, № 2, p. 4–29, квіт. 1984. https://doi.org/10.1109/massp.1984.1162229

15. Mei-qin, LIU, Sen-lin, Zhang, & Gang-feng, YAN. A new neural network model for the feedback stabilization of nonlinear systems, Liu et al. / J Zhejiang Univ Sci, 2008. 1023 p.
16. Ridella S., Rovetta S., Zunino R., “Circular backpropagation networks for classification,” IEEE Transactions on Neural Networks, Vol.8, No.1, pp.84-97, 1997.
17. Rowley H. A. Neural Network-Based Face Detection. IEEE Computer Society Conference on Computer Vision and Pattern Recognition – San Francisco, CA, 1996. 208 p.
18. Saad, R., Halgamuge, S. K., Stability of hierarchical fuzzy systems generated by Neuro-Fuzzy, Soft Computing, 2004. 416 p.
19. Saltelli, A., Chan, K., & Scott, E.M., Sensitivity analysis, Wiley & Sons Silvert, William, Fuzzy indices of environmental conditions. Ecological Modelling 2000, 119 p.
20. Setlak G., Bodyanskiy Y., Vynokurova O., and Pliss I., "Deep evolving gmdh-svm-neural network and its learning for data mining tasks," in 2016 Federated Conference on Computer Science and Information Systems, Sep. 11–14, IEEE, 2016
21. Steinwart and Christmann A., Support Vector Machines. New York: Springer, 2008.
22. Vandewalle J., Moor B. D., Gestel T. V., Brabanter J. D., and Suykens J. A. K., Least Squares Support Vector Machines. World Scientific Publishing Company, 2003
23. Vandewalle J., Moor B. D., Gestel T. V., Least Squares Support Vector Machines. World Scientific Publishing Company, 2003.
24. Vapnik V. N., The Nature of Statistical Learning Theory. New York: Springer, 1995.
25. Vapnik V., “Statistical learning theory,” John Wiley, New-York, 1998.
26. Zahirniak D. R., Chapman R., Rogers S. K., Suter B. W., Kabriski M., and Pyatti V., “Pattern recognition using radial basis function network,” Aerospace Application of Artificial Intelligence, pp. 249– 260, 1990.
27. Zahirniak D. R., Chapman R., Rogers S. K., Suter B. W., Kabriski M., and Pyatti V., “Pattern recognition using radial basis function network,” Aerospace Application of Artificial Intelligence,. 1990, 249– 260 p.
28. Русин Б. П. Системи синтезу, обробки та розпізнавання складноструктурованих зображень. Львів : Вертикаль, 1997. 262 с.

ДОДАТОК А

Відомість кваліфікаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	80 с.
	Інші документи	
2.	Презентаційні матеріали	15 плакатів

.	.	Прізвище та ініціали.	Підп.	Дата	
Розробив		Ревека К.І.			Розробка матричного методу опорних векторів з адаптивним комбінованим навчанням активаційної функції в рамках задачі розпізнавання образів
Перевірив		Чалий С.Ф..			
Н.контр.		Малєєва І.А.			
Затв.		Філатов В.О.			

Шифр групи	Код напр./спец.
СШІм-21-1	122
ХНУРЕ кафедра ШІ	