Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ ТА РЕАЛІЗАЦІЯ МЕТОДУ ОПОРНИХ ВЕКТОРІВ ДЛЯ ОБРОБКИ ДАНИХ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-22-1

Посашков В.Ю.
(прізвище, ініціали)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Кобилін О.А.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

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

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Посашкову Владиславу Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження та реалізація методу опорних векторів для обробки даних

затверджена наказом по університету від 3 листопада 2023 року № 1280Ст

2. Термін подання студентом роботи до екзаменаційної комісії 01 січня 2024 року.

3. Вихідні дані до роботи Науково-технічні публікації, дані Інтернет джерел та відомих наукових проектів, електронні документації

4. Перелік питань, що потрібно опрацювати в роботі

1. Аналіз предметної галузі та постановка задачі.
2. Штучні нейронні мережі з прямою передачею сигналів.
3. Машина опорних векторів.
4. Розробити застосунок для дослідження машини опорних векторів.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність дослідження методу опорних векторів, об'єкт, мета та постановка задачі для дослідження, тестові дата сети та приклади їх класифікації, апробація результатів роботи.

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

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	03.11.2023	
2	Аналіз завдання, підбір літератури	04.11.23-14.11.23	
3	Аналіз літератури з досліджуваної проблеми	15.11.23-25.11.23	
4	Аналіз методів розпізнавання зображень	26.11.23-30.11.23	
5	Дослідження методів машиного навчання	01.12.23-14.12.23	
6	Програмна реалізація	15.12.23-21.12.23	
7	Оформлення пояснювальної записки	22.12.23-30.12.23	
8	Перевірка на плагіат	31.12.2023	
9	Рецензування	02.12.2023	
10	Підготовка презентації та доповіді	08.01.2024	
11	Занесення роботи в електронний архів	09.01.2024	
12	Попередній захист кваліфікаційної роботи	10.01.2024	

Дата видачі завдання 3 листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Кобилін О.А.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 56 с., 1 табл., 30 рис., 45 джерел.

МЕТОД ОПОРНИХ ВЕКТОРІВ, КЛАСИФІКАЦІЯ ДАНИХ, ПІДТРИМКА ВЕКТОРІВ, ВІДРІЗКІВ, ВИБІР ЯДРА, ОПТИМІЗАЦІЯ ГІПЕРПАРАМЕТРІВ.

Об’єктом дослідження є обробки даних за допомогою машини опорних векторів.

Метою дослідження є розробка програмного засобу, який реалізує класифікацію даних за допомогою машини опорних векторів, реалізувати графічні візуалізації роботи алгоритму та показати вплив вибору параметрів моделі на точність моделі.

Використано метод класифікації машини опорних векторів. Проведено дослідження методів класифікації даних за допомогою алгоритмів навчання з учителем. Досліджено метод класифікації даних за допомогою машини опорних векторів та проблему перенавчання моделі, розроблено алгоритм класифікації даних за допомогою машини опорних векторів та перевірки залежності результатів класифікації моделі від її параметрів.

У результаті дослідження здійснена програмна реалізація системи обробки та класифікації даних для за допомогою машини опорних векторів та візуалізації результату класифікації.

SUPPORT VECTOR MACHINES, DATA CLASSIFICATION, SUPPORT VECTORS, KERNEL SELECTION, HYPERPARAMETER OPTIMIZATION.

The object of the research is data processing using Support Vector Machine (SVM).

The aim of the research is to develop a software tool for data classification using Support Vector Machine (SVM), create graphical visualizations of the algorithm’s operation, and investigate the impact of model parameter selection on its accuracy.

We used SVM for classification, researched supervised learning methods, addressed overfitting, developed an SVM-based algorithm, and implemented data processing software with result visualization.

As a result of the research, a software implementation for data processing and classification using Support Vector Machine (SVM) was carried out, along with the visualization of classification results.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

kNN– *k*-Nearest Neighbors (*k*-Найближчих Сусідів)

PCA– Principal Component Фnalysis (Метод Головних Компонентів

SVM– Support Vector Machines (Машина Опорних Векторів)

ВСТУП

Класифікація даних – це ключова функція в галузі машинного навчання та аналізу даних. Ця процедура вимагає віднесення об’єкта до певного класу на основі зразків даних, доступних для класифікатора. Для класифікації можуть бути використані різноманітні датасети, що включають зібрані дані про об’єкти, матриці відстаней, зображення тощо. Незважаючи на різноманіття даних, які надходять до класифікатора, основна мета полягає у відділенні об’єкта до одного з навчальних класів.

Алгоритми класифікації входять до розділу машинного навчання, відомого як навчання під наглядом. Такий тип навчання передбачає, що алгоритм отримує не тільки вхідні дані, але й відповіді на них, що дозволяє алгоритму навчатися на основі зразків. Задача алгоритмів навчання під наглядом полягає у встановленні зв’язків між вхідними об’єктами та їх результатами.

На сьогодні існує багато методів класифікації, які різняться своїми підходами до пошуку взаємозв’язків у даних для ефективної класифікації. Успішність класифікації об’єктів залежить не тільки від правильного вибору методу, але й від адекватної підготовки даних.

Однією з основних проблем у навчанні під наглядом є перенавчання. Перенавчання виникає, коли модель класифікації, натренована на прикладах з навчальної вибірки, стає надто «заточеною» під ці дані і не здатна адекватно класифікувати нові, невідомі дані. Тому для алгоритмів класифікації важливо не тільки запам’ятовувати дані, що надходять, але й узагальнювати залежності у даних.

На противагу перенавчанню, існує також проблема недонавчання, коли класифікатор стає занадто узагальненим і дає невірні результати. Недонавчання може бути пов’язане з недостатньою кількістю даних у навчальній вибірці або з параметрами моделі.

Ці проблеми часто виникають під час обробки даних, тому для успішної класифікації необхідне ретельне налаштування параметрів моделі і тщательний вибір та підготовка навчальних даних.

Таким чином, обробка даних з метою класифікації є важливим завданням сучасності. У цій роботі буде розглянута проблема обробки даних для класифікації з використанням методу опорних векторів (SVM), який використовується для відділення простору ознак на два класи через створення роздільної гіперплощини.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Навчання з учителем

Для кожної моделі інтелектуального аналізу даних необхідний процес навчання моделі. Цей процес важливий, оскільки у реальних умовах неможливо передбачити всі можливі варіанти даних, що надходять на класифікатор, та відповідні правильні відповіді, які він повинен видачати. Один із класів алгоритмів навчання – це алгоритми навчання з вчителем.

Навчання з вчителем є найбільш поширеним типом навчання моделі. У цьому випадку моделі подається прецедент, який складається з пари: дані про об’єкт і його результат чи відповідь. Під час розробки моделі з навчанням з вчителем потрібно створити алгоритм, який зможе встановити залежності всередині прецедентів і видачати відповідь із певною точністю [1-4].

Формальне визначення задачі може бути описане так:

Припустимо, що в нас є безліч об’єктів X, що представляють опис кожного з окремих об’єктів і їх відповідні результати $X = \{(x(1), y(1)), (x(2), y(2)), ..., (xm, ym)\}$, де m – кількість прикладів в навчальній вибірці і кожне спостереження є вектором $x(i) = \{x1, x2, ..., xn\}$, де n – кількість елементів опису одного спостереження та кожен із $y \in Y$ відповідає простору відповідей $Y=\{y(1), y(2), ..., y(k)\}$, де k – це кількість можливих відповідей (класів).

Має бути розроблена функція $f: X \rightarrow Y$, котра буде апроксимувати результат до істинного, тим самим зводячи до мінімуму функцію втрат як на тренувальній вибірці, так і на тестовій, а також на даних, що не були використані під час тренування.

Функція втрат $L(y, \hat{y})$ відображає розмір відмінності між відповіддю, отриманою в результаті класифікації, $y = f(x)$ та правильною відповіддю $\hat{y}$ – яка була зазначена в тренувальній чи тестовій вибірці. У задачах класифікації функція втрат може мати наступний вигляд:

$$(y, \hat{y}) = [y \neq \hat{y}]. \quad (1.1)$$

Для оцінки ефективності алгоритму використовується функція якості наданих відповідей, яка визначає емпіричний ризик функції f на обраній вибірці *Xm*:

$$R_{emp}(f, X^m) = \frac{1}{m}\sum_{i=1}^{m} L(f(x_i), \hat{y}_i). \quad (1.2)$$

Щоб знайти оптимальну модель, застосовується метод мінімізації емпіричного ризику, який полягає у виборі такої функції f з класу F , для якої буде забезпечено найменшу помилку на будь-якій вибірці для навчання *Xm*:

$$f = \underset{f}{\operatorname{argmin}}\, R_{emp}(f, X^m). \quad (1.3)$$

Проте мінімальна величина функції якості на тренувальній вибірці не гарантує, що конструйована модель ефективно відтворить коректні результати цільової залежності по всьому простору *X*. У такому випадку можливе явище перенавчання моделі. Перенавчання проявляється в тому, що алгоритм точно описує кожне спостереження, враховуючи навіть випадковий шум, що призводить до «підгонки» моделі під дані з тренувальної вибірки, втрачаючи здатність до адекватної реакції на нові спостереження, які не збігаються з тренувальною вибіркою. Така гіперфокусація на зниження емпіричної помилки до нульових значень може позбавити алгоритм можливості узагальнення [5-8]. Навчаючись на певній вибірці *Xm*, модель «запам’ятовує» лише певні дані та результати, що може призвести до неадекватної відповіді на нові дані. Щоб уникнути цієї проблеми, необхідно правильно підібрати параметри моделі та застосовувати методи, які зменшують ризик перенавчання.

1.2 Методи класифікації

Наразі існує чимало різноманітних алгоритмів класифікації. У цьому розділі розглянуто та описано деякі з найбільш вживаних методик [9-11], які застосовуються у сфері машинного навчання та аналізу даних.

Одним з найбільш популярних та інтуїтивно зрозумілих алгоритмів є метод k-найближчих сусідів (рис. 1.1). До основних етапів цього алгоритму відносять:

– обчислення відстаней від класифікованого об’єкта до всіх об’єктів у навчальному наборі даних;

– визначення k об’єктів, які мають найменшу відстань до об’єкта, що класифікується;

– присвоєння класифікованому об’єкту класу, до якого належить більшість з k найближчих сусідів.

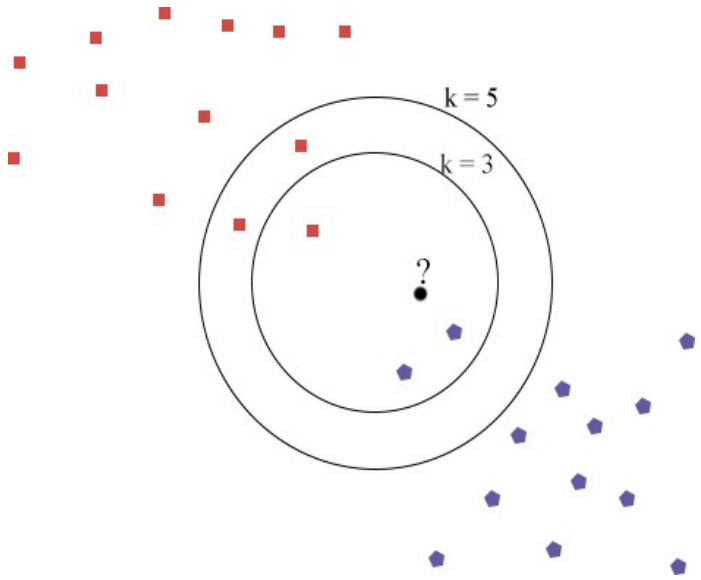


Рисунок 1.1 – Алгоритм kNN

Слід підкреслити, що перед використанням методу *k*-найближчих сусідів необхідно здійснити нормалізацію даних для забезпечення однакової ваги всіх атрибутів. Класичний метод передбачає розрахунок відстаней в евклідовому просторі.

Іншим значущим напрямком в алгоритмах класифікації є метод дерев рішень. Ці алгоритми визначають вплив атрибутів на результат, на основі чого будується дерево рішень, яке дозволяє класифікувати об’єкт. Дерево складається з вузлів, кожен з яких описує правило для вхідних даних. Розгалуження вузлів відбувається згідно з класифікацією об’єктів за певною умовою.

Об’єкт класифікації має пройти через всі вузли дерева, доки не дійде до листкової вершини, яка символізує певний клас. Клас, що відповідає цій вершині, присвоюється об’єкту. Класифікацію можна виразити через кон’юнкцію правил, яким підпорядковується спостереження.

Ще одним відомим алгоритмом є наївний байєсовський класифікатор. Він відрізняється простотою і іноді дуже високою ефективністю, оскільки передбачає незалежність між атрибутами. На основі теореми Байєса визначається імовірність приналежності об’єкта до класу, і вибирається клас з найбільшою ймовірністю.

Далі розглядатиметься алгоритм машини опорних векторів (SVM), який є одним із найпопулярніших методів класифікації. Його сутність полягає у створенні поділяючої гіперплощини, яка повинна бути однаково віддаленою від опорних векторів обох класів.

1.3 Штучні нейронні мережі з прямою передачею сигналів

Штучні нейронні мережі – це обчислювальні системи, що імітують роботу мозку живого організму, спрощуючи реальну модель мозку. Біологічний нейрон – це складна система, яка не лише передає та обробляє сигнал, але й підтримує власну життєдіяльність.

Математичний нейрон складається з набору входів для прийому сигналів, ваг, які призначені для кожного входу, суматора, що обчислює суму

добутків вхідних сигналів на ваги, активаційної функції та виходу нейрона. Розглянемо модель такого нейрона (рис. 1.2).

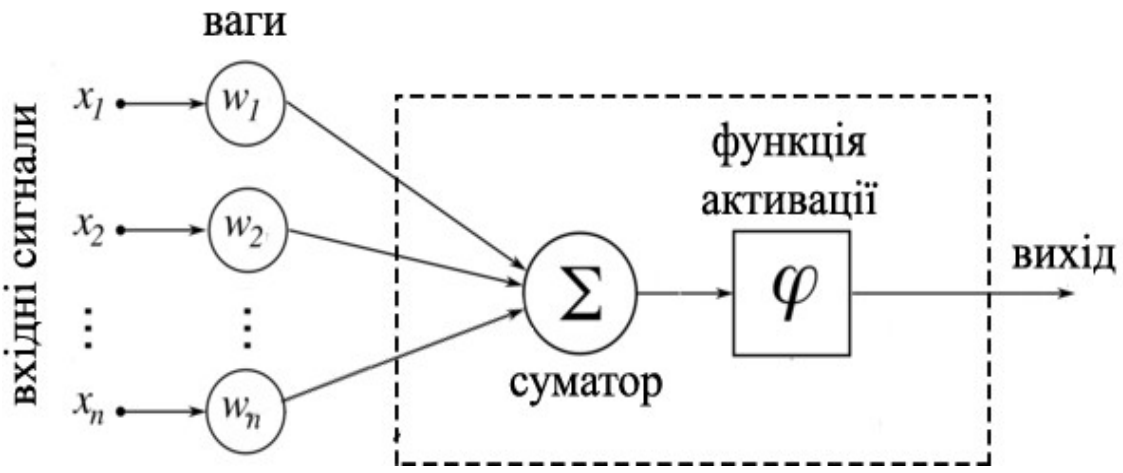


Рисунок 1.2 – Приклад схеми штучного нейрону

Кожен нейрон має входи *X*, через які приймається сигнал. Кожний вхідний сигнал помножується на відповідну вагу w_i. Після цього суматор обчислює суму цих добутків.

$$\sum_{i=0}^{n} x_i w_j = W^t X. \quad (1.4)$$

Після обчислення суми добутків отриманий результат подається в активаційну функцію φ, яка визначає вихідний сигнал нейрона:

$$out = \varphi(W^T X). \quad (1.5)$$

Функція активації визначає, як нейрон перетворює суму добутків входів і ваг у вихідний сигнал [12-15]. Тож нейрон характеризується своїми вагами та функцією активації. Вибір конкретної функції активації залежить від поставленої задачі і обраної стратегії. Поширені функції активації представлені таблиці 1.1.

Таблиця 1.1 – Активаційні функції

Назва	**Формула**	**Область значень**
Порогова	$\begin{cases} 0, & W^T X < T \\ 1, & W^T X \geq T \end{cases}$	{0, 1}
Знакова	$\begin{cases} -1, & W^T X < 0 \\ 1, & W^T X \geq 0 \end{cases}$	{-1, 1}
Сигмоідальна	$\frac{1}{1,\ W^T X \geq 0}$	(0, 1)
Лінійна	$W^t X$	(-∞, +∞)
Полулінійна	$\begin{cases} 0, & W^T X < 0 \\ W^T X, & W^T X \geq 0 \end{cases}$	(0, +∞)
Радіально-базисна	$e^{-(W^T X)}$	(0, 1)
Гіперболічний тангенс	$\frac{e^{2(W^T X)} - 1}{e^{2(W^T X)} + 1}$	(0, 1)

Нейронні мережі з прямим поширенням сигналу, відомі також як feedforward neural networks, відрізняються лінійною структурою, де інформація пересувається в одному напрямку, починаючи з вхідного шару, через приховані шари, і в кінцевому результаті до вихідного шару. Ця лінійність і відсутність зворотних зв’язків забезпечують певні переваги у швидкості обчислень і спрощеності тренування мережі [16-18].

Кожен прихований шар у мережі прямого поширення може розглядатися як рівень абстракції, де кожен наступний шар вибудовує складніші представлення даних на основі вихідних даних попереднього шару. Це дозволяє мережі «вчитися» і «розуміти» складні закономірності та залежності вхідних даних, що є критично важливим для задач, таких як розпізнавання образів, обробка природної мови і багато інших.

Важливою складовою при роботі з нейронними мережами є вибір функції активації, оскільки саме вона визначає, як сигнали від нейронів перетворюються і передаються між шарами. Залежно від складності задачі та типу даних, можуть використовуватися різні типи активаційних функцій, кожна з яких має свої особливості та переваги.

Мережі прямого поширення є основою для багатьох сучасних архітектур глибинного навчання. Розвиток таких мереж з часом призвів до створення складніших структур, які включають згорткові нейронні мережі (CNN), рекурентні нейронні мережі (RNN), мережі з довготривалою пам'яттю (LSTM), та інші, кожна з яких оптимізована для певних видів задач та даних. Розглянемо структуру нейронної мережі (рис. 1.3).

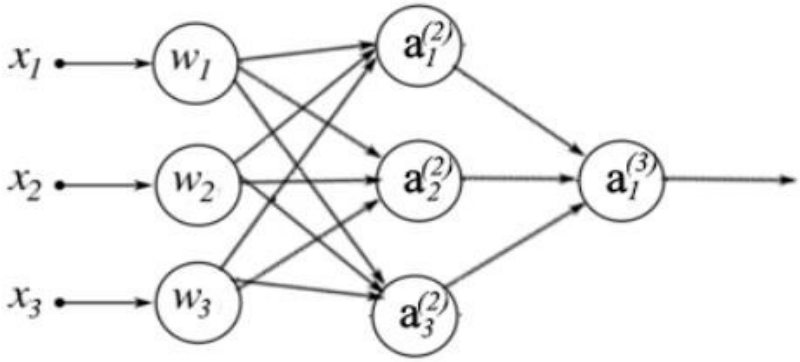


Рисунок 1.3 – Структура нейронної мережі прямого поширення

Результат активації нейрона і у шарі $j \in L$ можна позначити як $a(j)$, де j визначає порядковий номер шару в нейронній мережі, що складається з L шарів Матриця ваг, яка зв'язує шар j з наступним шаром $j+1$, позначається як $\theta(j)$ і включає ваги W для кожного нейрону на даному шарі. Таким чином:

$$a_i^{(j)} = \varphi\left(\sum_{k=0}^{m} \theta_{ik}^{(j-1)} x_k\right), \tag{1.6}$$

де m – це кількість вхідних сигналів на нейрон і шару j.

Вихід нейронної мережі представлятиме собою активації нейронів останнього шару L, що є кінцевим результатом роботи мережі:

$$h_\theta(x) = a^{(L)}. \tag{1.7}$$

Комплексність нейронної мережі визначається кількістю її прихованих шарів та кількістю нейронів у кожному з цих шарів. Включення цих додаткових шарів дозволяє нейронній мережі більш гнучко моделювати складні та нелінійні відносини серед даних.

У процесі навчання з вчителем, нейронній мережі подаються приклади з навчального набору даних, які складаються з пар вхідних даних та відповідних бажаних виходів. Навчання відбувається ітеративно, де кожна ітерація включає обчислення прямого та зворотного розповсюдження сигналів, за якими визначаються помилки та обчислюються градієнти для коригування параметрів мережі. Існують два основні режими навчання: послідовний, коли корекція параметрів відбувається після кожного окремого прикладу, і пакетний, коли корекція параметрів відбувається на основі кумулятивного градієнта, обчисленого за весь набір прикладів. Повний цикл представлення всіх прикладів навчальної множини, який закінчується одним коректуванням параметрів, називається епохою навчання.

З математичної точки зору, процес навчання полягає у мінімізації функції втрат через оптимізацію вагових коєфіцієнтів θ. Як правило, в якості цільової функції використовується сума квадратів помилок між прогнозованими та реальними значеннями виходів мережі [19-21]:

$$\boxed{E(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(h_\theta\left(x^{(i)}\right) - y\left(x^{(i)}\right)\right)^2}, \quad (1.8)$$

де N – кількість представлених пар прецедентів у навчальній вибірці.

Методи навчання, які застосовуються для налаштування вектора параметрів адаптивного елемента, використовують тільки ту інформацію, яка доступна в межах цього елемента. Коректувальна зміна $\Delta\theta i$ для параметра θi розраховується на основі локального градієнта $\delta\theta i$ цього параметра і залежить від змін цього градієнта протягом певної кількості епох навчання. Метод градієнтного спуску є одним з найбазовіших і найпоширеніших

методів для навчання мережі. Зміна параметра відбувається згідно з певним математичним виразом, який враховує ці градієнти.

$$\theta_{i+1} = \theta_i + \theta_i. \tag{1.9}$$

При цьому коригуюча зміна $\Delta\theta_i$ обчислюється таким чином:

$$\Delta\theta i = -\varepsilon\delta\theta i, \tag{1.10}$$

де ε – це коефіцієнт швидкості навчання.

У формулі (1.10) наявність мінуса перед коефіцієнтом необхідна для того, щоб змінювати параметр в напрямку зменшення значення функції втрат. Вибір коефіцієнта швидкості навчання є критично важливим: якщо він буде занадто малим, процес навчання затягнеться і потребуватиме більше ітерацій; з іншого боку, занадто великий коефіцієнт може призвести до нестабільності навчання, коли параметр змінюється настільки значно, що «пропускає» оптимальні значення. Зазвичай величина коефіцієнта вибирається на основі певних умов.

$$0 < \varepsilon < 1. \tag{1.11}$$

2 МАШИНА ОПОРНИХ ВЕКТОРІВ

Машина опорних векторів (SVM) є потужним інструментом в машинному навчанні, що підпадає під категорію навчання під наглядом. Ця модель використовується для розв'язання задач класифікації та регресії і базується на принципі структурного ризику мінімізації, що забезпечує оптимальне рішення.

У контексті бінарної класифікації, метод опорних векторів знаходить гіперплощину або набір гіперплощин у високовимірному просторі, які можуть використовуватися для класифікації даних. Ціллю є вибір гіперплощини з максимальним маржином, тобто найбільшою можливою відстанню між гіперплощиною та найближчими точками даних з обох класів, які називаються опорними векторами. У ідеалі, коли дані лінійно роздільні, існує чітка границя, що розділяє класи (рис. 2.1).

SVM включає розширення для роботи з нелінійно роздільними даними, застосовуючи техніку, відому як «ядерний трюк», яка дозволяє ефективно виконувати класифікацію в багатовимірних просторах, не підвищуючи реальної розмірності даних. Ця можливість робить SVM надзвичайно потужним і гнучким інструментом в широкому спектрі застосувань.

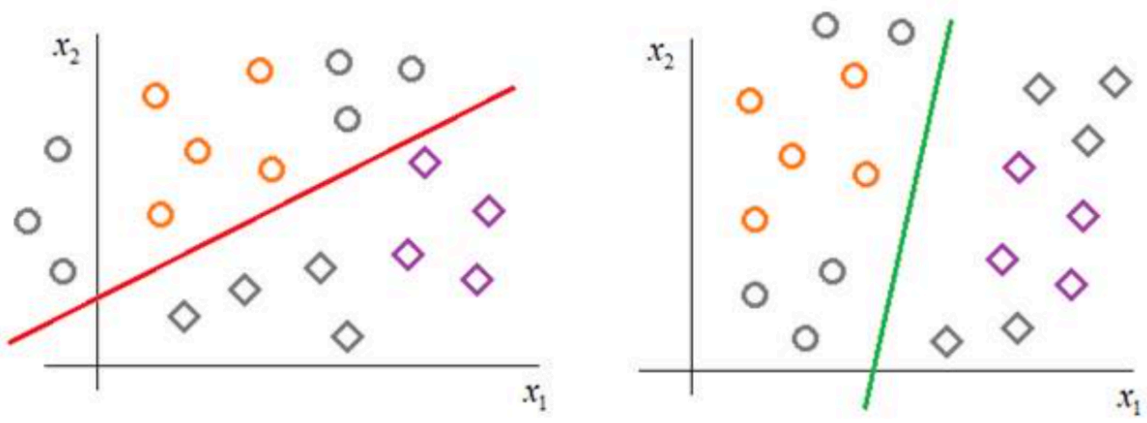


Рисунок 2.1 – Приклад роботи SVM на лінійно роздільній вибірці

На рисунку 2.1 опорні вектори позначені сірими точками. Вони використовуються для конструювання гіперплощини, яка ефективно розділяє простір спостережень на два класи. Вхідні спостереження класифікуються до першого або другого класу залежно від їх положення відносно цієї розділової гіперплощини.

У випадках, коли дані не є лінійно роздільними, використовується техніка ядер, що дозволяє перевести ознаковий простір у вищу розмірність, де класи стають лінійно роздільними. Ядра відіграють роль активаційних функцій в нейронних мережах опорних векторів, також відомих як SVM, які є частиною більш широкого класу нейронних мереж з прямим поширенням.

SVM розширює парадигму класичних штучних нейронних мереж, таких як багатошарові перцептрони, радіально-базисні функції та поліноміальні мережі, і включає в себе використання ядерних функцій для реалізації нелінійного відображення вхідних даних. Ці мережі оптимізують емпіричний ризик, мінімізуючи помилку між прогнозованими і реальними значеннями, що робить їх ефективними у багатьох застосуваннях, таких як ідентифікація, розпізнавання образів, апроксимація функцій, моделювання та багато інших областей.

2.1 Лінійно розділима вибірка

Ключовим елементом у процесі синтезу мережі SVM є опорні вектори, термін, який було введено В. Н. Вапником. Ці вектори представляють найбільш значущу підмножину даних з навчальної вибірки, яка використовується для визначення розділової гіперплощини в процесі навчання. Підхід до мінімізації емпіричного ризику можна продемонструвати на прикладі простої задачі розпізнавання образів, де маємо навчальну вибірку з відомими класифікаціями для набору даних. У цій задачі, кожен елемент навчальної вибірки містить вектор ознак *x(i)* та мітку класу *y(i)*, де у

приймає значення +1 або -1 в залежності від класу, до якого належить цей вектор. Рівняння, яке описує гіперплощину, що розділяє два лінійно розділені класи, визначається математичною формулою:

$$w^T x + b = 0, \tag{2.1}$$

де *w – (n × 1)* - вектор ваги, що підлягають визначенню у [22-25] узагальненим портретом;

b – скаляр, визначаючий зміщення, а приналежність даних конкретного класу визначається парою нерівностей:

$$\begin{cases} w^T x^{(i)} + b = 0 \geq 0, \text{для } y = +1, \\ w^T x^{(i)} + b = 0 \geq 0, \text{для } y = +1, \end{cases} \tag{2.2}$$

де і відповідає одному із номерів спостереження та знаходиться у проміжку $1 \leq i \leq m$

Для конкретного набору тренувальних даних, загальної моделі та зміщення b, вводиться концепція зони розділення p. Ця зона визначається відстанню від розділової гіперплощини до найближчих точок xi, які належать до різних класів. Задача мінімізації емпіричного ризику вимагає виявлення такої гіперплощини, яка б максимізувала цю зону розділення. Точки, які знаходяться найближче до цієї ідеальної гіперплощини, називаються опорними векторами.

Виразимо оптимальну гіперплощину за допомогою наступного рівняння:

$$x^T w^* + b^* = 0, \tag{2.3}$$

$$D(x) = x^T w^* + b^*. \tag{2.4}$$

Формулюючи відстань від будь-якої точки x до гіперплощини заданої рівнянням (2.3), ми можемо розглянути вектор x як результат додавання наступних компонентів:

$$\mathrm{x} = \hat{\mathrm{x}} + \tilde{\xi}\frac{\mathrm{w}^*}{\|\mathrm{w}^*\|}, \tag{2.5}$$

де $\hat{x}$ – проекція x на гіперплощину (2.3);

ξ – алгебраїчний параметр відстані, що приймає значення більше нуля на «позитивній» стороні гіперплощини и менше нуля на «негативній».

Оскільки $D(\hat{x}) = 0$, очевидно, що:

$$D(x) = x^T w^* + b^* = \tilde{\xi}\|\mathrm{w}^*\|, \tag{2.6}$$

звідки:

$$\tilde{\xi} = \frac{\mathrm{D(x)}}{\|\mathrm{w}^*\|} \tag{2.7}$$

Аналізуючи формулу (2.6), можемо констатувати, що відстань від початку координат до гіперплощини, представленої рівнянням (2.3), визначається як b поділене на норму вектора w^*. Якщо зміщення b^* є позитивним, то це свідчить про те, що початок координат розташований з позитивного боку гіперплощини. У випадку, коли b^* менше нуля, початок координат знаходиться з негативного боку. Коли b^* дорівнює нулю, поділяюча гіперплощина проходить прямо через початок координат.

Оптимальна гіперплощина має відповідати більш строгим умовам [26-30], ніж наведено в нерівностях (2.2):

$$\begin{cases} w^T x^{(i)} + b = 0 \geq 1, \text{для } y = +1, \\ w^T x^{(i)} + b = 0 \leq -1, \text{для } y = -1, \end{cases} \quad (2.8)$$

Пари (x, y), які перетворюють будь-яку з нерівностей (2.8) на рівність, є опорними векторами. Вони розташовані найближче до розділової гіперплощини, що робить їх найскладнішими для класифікації. Ключова концепція методу опорних векторів полягає в тому, що ідентифікація опорних векторів з допомогою навчальної вибірки надає достатню інформацію для класифікації інших точок даних. З рівняння (2.7) витікає, що відстань до оптимальної гіперплощини для будь-якого опорного вектора x(s) визначається наступним чином [31-34]:

$$\xi = \frac{D\left(x^{(s)}\right)}{\|w^*\|} = \begin{cases} \frac{1}{\frac{D\left(x^{(s)}\right)}{\|w^*\|}}, \text{для } y^{(s)}=+1, \\ -\frac{1}{\frac{D\left(x^{(s)}\right)}{\|w^*\|}}, \text{для } y^{(s)} = -1, \end{cases} \quad (2.9)$$

Знак «плюс» вказує, що опорний вектор *x(s)* розташований на позитивній стороні ідеальної гіперплощини, а «мінус» вказує на його розташування на негативній стороні. Відповідно до рівняння (3.9), ширина зони розділення задається як:

$$p = 2\xi = \frac{2}{\|w^*\|}, \quad (2.10)$$

де максимізація *p* еквівалентна мінімізації Евклідової норми вектору ваг ваг $\|w\|$.

Отже, мінімізація емпіричного ризику в контексті заданої навчальної вибірки *X = {(x(1), y(1)), (x(2), y(2)), ..., (x(m), y(m))}*, полягає у знаходженні таких параметрів моделі, які мінімізують задану цільову функцію:

$$E(w) = \frac{1}{2}\|w^*\|^2, \quad (2.11)$$

при обмеженнях виду (2.7) або, що, те ж саме:

$$y^{(k)}(w^T x^{(k)} + b) \geq 1, k = 1, 2, \ldots, N. \quad (2.12)$$

Очевидно, що формули (2.10), (2.11) формулюють класичну задачу квадратичного програмування [35].

Метод невизначених множників Лагранжа може бути використаний як прозорий і зрозумілий підхід до аналізу цієї проблеми, який зосереджується на визначенні критичних точок спеціально сформованої функції, лагранжіана, що в даному контексті приймає форму:

$$L(w, b, \lambda) = 2\, wTw - \sum_{k=1}^{N} \lambda_k (y^{(k)}(w^T x^{(k)} + b) - 1), \quad (2.13)$$

де λk – є невід'ємними множниками Лагранжа.

Знаходження оптимального рішення зводиться до пошуку сідлової точки лагранжіана (2.12), яка визначається системою Куна-Таккера, і полягає в тому, щоб знайти такі параметри, при яких похідні лагранжіана по всім змінним дорівнюють нулю, умови Куна-Таккера виконуються, а множники Лагранжа є невід'ємними [36-39]:

$$\begin{cases} \frac{\delta L(w,b,\lambda)}{\delta w} = 0, \\ \frac{\delta L(w,b,\lambda)}{\delta b} = 0, \\ \frac{\delta L(w,b,\lambda)}{\delta \lambda_k} \leq 0, \lambda_k \geq 0, k = 1,2, \ldots, N. \end{cases} \quad (2.14)$$

З першого рівняння (2.14) системи Куна-Таккера випливає, що шуканий вектор ваг можна визначити як суму добутків вхідних векторів на їх відповідні множники Лагранжа та класові мітки:

$$\boxed{w = \sum_{k=1}^{N} \lambda_k y^{(k)} x^{(k)}}, \tag{2.15}$$

З другого рівняння видно, що сума добутків множників Лагранжа та класових міток повинна дорівнювати нулю:

$$\boxed{\sum_{k=1}^{nN} \lambda_k y^{(k)} = 0}. \tag{2.16}$$

З виразів (2.15), (2.16) випливає, що шуканий вектор ваг можна визначити як суму добутків вхідних векторів на їх відповідні множники Лагранжа та класові мітки. З другого рівняння видно, що сума добутків множників Лагранжа та класових міток повинна дорівнювати нулю:

$$\boxed{\lambda_k (y^{(k)} (w^T x^{(k)} + b) - 1) = 0, k = 1, 2, \ldots, N}. \tag{2.17}$$

Хоча для унікальної оптимальної гіперплощини може існувати безліч рішень множників Лагранжа, які не можна визначити аналітично, у сідловій точці має виконуватися певна умова, що задовольняє нульовий добуток множників Лагранжа на відповідні класифікаційні умови для кожного вектора.

Для розв’язання цієї складності можна розглянути двоїсту задачу квадратичного програмування [40-42], рішення якої еквівалентно знаходженню оптимальної гіперплощини. Це відповідає фундаментальному принципу загальної теорії оптимізації, згідно з яким, якщо існує рішення для

прямої задачі оптимізації, то існує і рішення для відповідної двоїстої задачі, і обидва рішення співпадають.

Щоб знайти множники Лагранжа, необхідно підставити отримані вирази (2.15) та (2.16) до лагранжіан:

$$\begin{aligned} L(w, b, \lambda) &= \frac{1}{2} w^T w - \sum_{k=1}^{N} \lambda_k y^{(k)} \left(w^T x^{(k)} + b \right) - 1 \Big) = \\ &= \frac{1}{2} w^T w - \left(w^T w - \sum_{k=1}^{N} \lambda_k \right) = \\ &= \sum_{k=1}^{N} \lambda_k - \frac{1}{2} \left\| \sum_{k=1}^{N} \lambda_k y^{(k)} x^{(k)} \right\|^2 . \end{aligned} \tag{2.18}$$

Таким чином, задача зводиться до пошуку критичних точок функції

$$\Phi(\lambda) = \sum_{k=1}^{N} \lambda_k - \frac{1}{2} \left\| \sum_{k=1}^{N} \lambda_k y^{(k)} x^{(k)} \right\|^2 . \tag{2.19}$$

Щоб знайти множники Лагранжа, необхідно підставити отримані вирази до лагранжіану, що призводить до функції, яка є різницею лінійної функції та квадратичної функції з негативною визначеністю. Задача полягає у знаходженні максимального значення функції *Φ(λ)* з урахуванням умов (2.16) та невід'ємності множників Лагранжа, після чого відбувається оптимізація за допомогою відповідного алгоритму оптимізації.

2.2 Лінійно нерозділима вибірка

У випадку коли навчальний датасет є лінійно нероздільним, з'являється потреба в трансформації існуючого простору даних до простору більш високої розмірності, що відкриває можливості для ефективного розділення категорій. Машина опорних векторів (SVM) пропонує унікальне рішення для цього завдання, використовуючи концепцію ядерного перетворення.

Серед варіантів трансформації використовуються так звані Гауссіани, які описуються наступним чином:

$$\varphi(x) = \exp\left(-\frac{\|x - x(s)\|^2}{2\sigma^2}\right). \quad (2.20)$$

де σ – це параметр, що регулює ширину «вікна» Гауссового ядра, можуть використовуватися для відображення даних у новий простір, де лінійне розділення стає можливим.

Альтернативним методом є поліноміальні ядра, які визначаються як:

$$\varphi(x) = (x^T x^{(s)} + r)^d. \quad (2.21)$$

де r – є параметром зсуву, а d є степенем поліному, дозволяють розглядати нелинійні взаємозв'язки між ознаками.

Однією з ключових характеристик SVM є те, що кількість нейронів у прихованому шарі може бути точно визначена наперед і відповідає кількості опорних векторів, знайдених під час тренування. Ці опорні вектори – це елементи даних, які лежать найближче до рішучої границі між класами, і є вирішальними у визначенні остаточної форми розділової гіперплощини:

$$\begin{cases} \varphi(x^{(s)})^T w^* = 1, \text{для } y = +1 \\ \varphi(x^{(s)})^T w^* = -1, \text{для } y = -1 \end{cases}. \quad (2.22)$$

Використання опорних векторів призводить до обчислювально складного процесу, оскільки процедура оптимізації вимагає налаштування численних вільних параметрів, які можуть суттєво впливати на результати моделювання. SVM зазвичай використовуються в пакетному режимі, працюючи з даними фіксованого розміру.

Проте, SVM також відрізняються складністю процесу навчання, особливо при вирішенні задачі оптимізації з обмеженнями у формі нерівностей, що вимагає значних обчислювальних ресурсів, особливо для великих наборів даних. Це обмеження може бути особливо критичним для великих датасетів, де кількість тренувальних прикладів є великою, оскільки складність навчання зростає пропорційно до обсягу даних.

Успіх застосування SVM в значній мірі залежить від грамотного вибору ядер та їх параметрів, які повинні бути адаптовані до специфіки конкретної задачі. Це вимагає глибокого розуміння як математичних основ методу, так і природи оброблюваних даних.

2.3 Збалансована вибірка

У багатьох задачах класифікації реальних даних виявляється, що розподіл класів у тренувальному наборі даних є незбалансованим. Тобто, кількість зразків одного класу набагато перевищує кількість зразків іншого класу. Ця незбалансовість може призвести до серйозних проблем при навчанні класифікаторів, включаючи машину опорних векторів (SVM). Зазвичай стандартна реалізація SVM має тенденцію перекосити результати в користь чисельно більшого класу, що може призвести до низької точності та надмірного помилкового класифікації меншого класу. Для вирішення цієї проблеми використовуються методи збалансованого SVM.

Один із способів збалансування класів у SVM – це зміна ваг класів. У стандартній реалізації SVM всі приклади мають однакову вагу при оптимізації. У методі зміни ваг класів важливим класам надають більшу вагу, тим самим зменшуючи вплив перекосу в результатах. Це досягається шляхом встановлення відповідної ваги для кожного класу при навчанні SVM. Ваги можуть бути налаштовані з урахуванням розподілу класів у навчальному наборі.

Оптимізаційна задача для зміни ваг класів може бути записана наступним чином:

$$\min_{\alpha} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{N} \alpha_i . \quad (2.23)$$

обмеження:

$$\begin{gathered} 0 \leq \alpha_i \leq C_i \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{gathered} . \quad (2.24)$$

де C_i - гіперпараметр, що визначає вагу класу *i*.

Інший підхід полягає у зверненні до зразків меншого класу. Цей метод включає в себе створення додаткових прикладів для меншого класу шляхом копіювання існуючих зразків або шляхом генерації штучних зразків, які схожі на існуючі. Основна ідея полягає в тому, щоб збільшити кількість прикладів меншого класу, зробивши її схожою на більший клас. Це допомагає створити більш збалансований навчальний набір даних для SVM. Математично це можна представити як:

$$x' = x + \lambda * (x_n - x) . \quad (2.25)$$

де x_n– це один з сусідів *x* з більшого класу, а λ - параметр, що регулює розмір штучного зразка відносно оригінального.

Цей підхід дозволяє збалансувати навчальний набір даних та покращити результати класифікації, зменшуючи вплив незбалансованості класів. Ви можете дослідити різні методи генерації штучних зразків і визначити їх ефективність для вашої конкретної задачі класифікації.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ АЛГОРИТМІВ

3.1 Аналіз засобів програмної реалізації методу опорних векторів

Для реалізації методу опорних векторів (SVM) була обрана мова програмування Python [43]. Ця скриптова мова є ідеальним вибором для вирішення поставленої задачі, оскільки вона має численні переваги порівняно з іншими мовами програмування.

Однією з головних переваг Python є те, що інтерпретатор цієї мови доступний практично на всіх платформах та операційних системах. Це забезпечує платформонезалежність та легкість встановлення на різних середовищах.

Крім того, Python відомий своєю постійною розширюваністю, що дозволяє розробникам легко додавати нові функції та бібліотеки для вирішення конкретних завдань. Відкритий вихідний код мови надає можливість будь-якому програмістові завантажити та модифікувати код для власних потреб.

Також, дуже важливою особливістю цієї мови є наявність різноманітних бібліотек для завантаження даних, візуалізації, статистичних обчислень, обробки природної мови, обробки зображень і багато іншого. Цей великий набір інструментів пропонує фахівцям з роботи з даними (data scientists) великий набір інструментів загального і спеціального призначення. Однією з основних переваг використання Python є можливість безпосередньо працювати з програмним кодом за допомогою терміналу або інших інструментів, таких як Google Colab.

Google Colab є особливо корисним інструментом для фахівців з аналізу даних та машинного навчання. Ця платформа надає безкоштовний доступ до потужних обчислювальних ресурсів, таких як графічні процесори (GPU) та одиниці обробки графіки (TPU), що значно прискорюють розрахунки при великих обсягах даних та складних моделях машинного навчання. Крім того,

Google Colab дозволяє працювати над проектами в режимі реального часу у браузері (рис. 3.1), обмінюючись кодом та результатами аналізу з іншими користувачами.

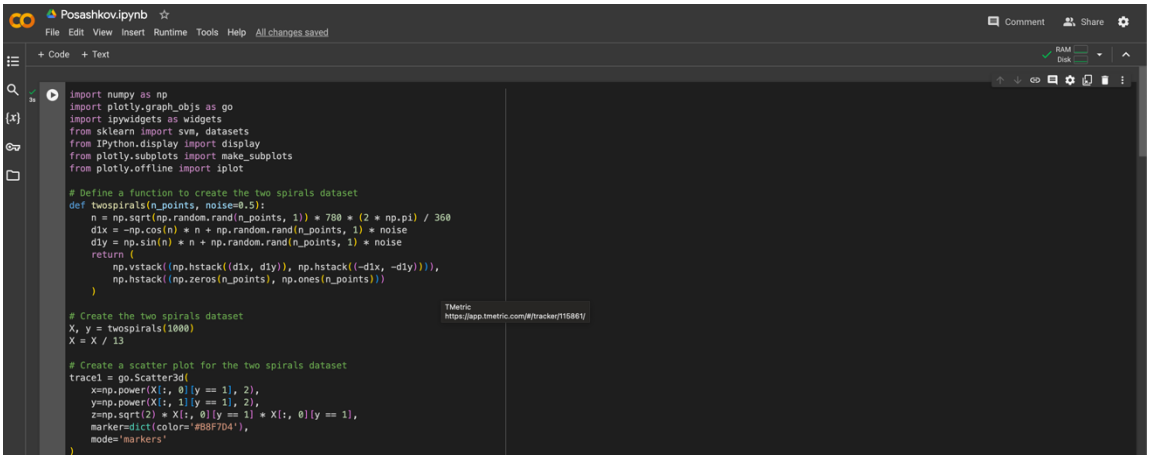


Рисунок 3.1 – Графічний інтерфейс середи розробки Google Colab

Ця інтерактивна середовище для запуску програмного коду в браузері є ідеальним вибором для розвідувального аналізу даних, розробки та налаштування моделей машинного навчання, а також для документування та візуалізації результатів. Його можна використовувати для створення та виконання коду на мові програмування Python, встановлення необхідних бібліотек та інструментів, а також для зручного зберігання та обміну ноутбуками, в яких зберігаються всі кроки аналізу та експерименти. Таким чином, Google Colab робить процес розробки та дослідження методу опорних векторів (SVM) та інших аналітичних завдань більш доступним та продуктивним.

Scikit-learn – проект з відкритим вихідним кодом, що надає можливість вільно використовувати та поширювати його, і дозволяє кожній людині легко отримувати вихідний код для вивчення його внутрішньої структури та роботи «за лаштунками» . Проект Scikit-learn постійно розвивається і вдосконалюється завдяки активній спільноті користувачів. Він містить ряд сучасних алгоритмів машинного навчання та повну документацію по кожному з них. Scikit-learn є найвідомішою інструментальною бібліотекою

Python для машинного навчання та широко використовується в промисловості і науці. У мережі Інтернет доступний великий вибір навчальних матеріалів і прикладів програмного коду для роботи з ним. Scikit-learn також добре поєднується з іншими науковими інструментами Python.

NumPy – це один з основних пакетів для наукових обчислень в Python, і він має велику функціональність для роботи з багатовимірними масивами і високорівневими математичними функціями, такими як операції лінійної алгебри та генерація псевдовипадкових чисел. У бібліотеці Scikit-learn, яка використовується для розробки методу опорних векторів (SVM), масиви NumPy виступають як основна структура даних, і всі дані, які використовуються, повинні бути перетворені в масиви NumPy (рис. 3.2).

```
import numpy as np
array = np.array([[1, 2, 3], [4, 5, 6]])
print(array)
```

```
[[1 2 3]
 [4 5 6]]
```

Рисунок 3.2 – Масив NumPy

Також важливо відзначити дистрибутив Anaconda, який є ідеальним рішенням для великомасштабної обробки даних, прогнозної аналітики і наукових обчислень. Anaconda вже містить в собі пакети NumPy, Jupyter Notebook і Scikit-learn, що робить його зручним вибором для фахівців з аналізу даних, які шукають готове середовище для роботи.

Для візуалізаці ї даних була використана бібліотека Plotly. У Python існують численні бібліотеки для візуалізації даних, такі як Matplotlib, Seaborn, Bokeh, Pygal та інші, але Plotly вирізняється як бібліотека, яка дозволяє будувати інтерактивні графіки та використовувати їх без обмежень. Plotly надає можливість створювати різні типи графіків, такі як точкові

діаграми, графіки ящиків, тривимірні графіки, стовпчаті діаграми, теплові карти, дендрограми та інші. Завдяки вбудованому режиму офлайн, Plotly можна використовувати без необхідності реєстрації та публікації графіків онлайн. З великою кількістю прикладів та докладною документацією, Plotly стає потужним інструментом для візуалізації даних у наукових дослідженнях та аналізі даних. Використання цієї бібліотеки спростило створення інтерактивних графіків для відображення результатів класифікації в нашому дослідженні та полегшило їх розуміння.

Також щодо переваги бібліотеки Plotly можна відзначити ще можливість використання Plotly на різних платформах та мовах програмування. Окрім Python, Plotly підтримує інші мови, такі як R, JavaScript, та MATLAB. Це дає можливість робити візуалізацію даних на різних етапах дослідження, а також обмінюватися результатами з колегами, які працюють у різних середовищах.

3.2 Експериментальні дослідження

Під час наукових експериментів розглянемо кілька наборів даних, за допомогою яких модель машини опорних векторів (SVM) буде навчена. Також розгялнемо вплив на результати обробки і класифікації даних різних параметрів моделі.

За допомогою інструменту Plotly покажемо процес класифікації даних за допомогою моделі машини опорних векторів. Перший приклад представлений за допомогою набору даних «Double donut», який складається з двох розташованих одна в одній окружностей. Для завантаження цього набору даних до пам’яті використовується функція «make_circles» з пакету «datasets» бібліотеки Scikit-learn. У цю функцію передаються кількість спостережень, рівень шуму та фактор масштабу між внутрішньою та

зовнішньою окружностями. Графік (рис 3.3) відображає зовнішній вигляд набору даних та приналежність кожної окружності до певного класу

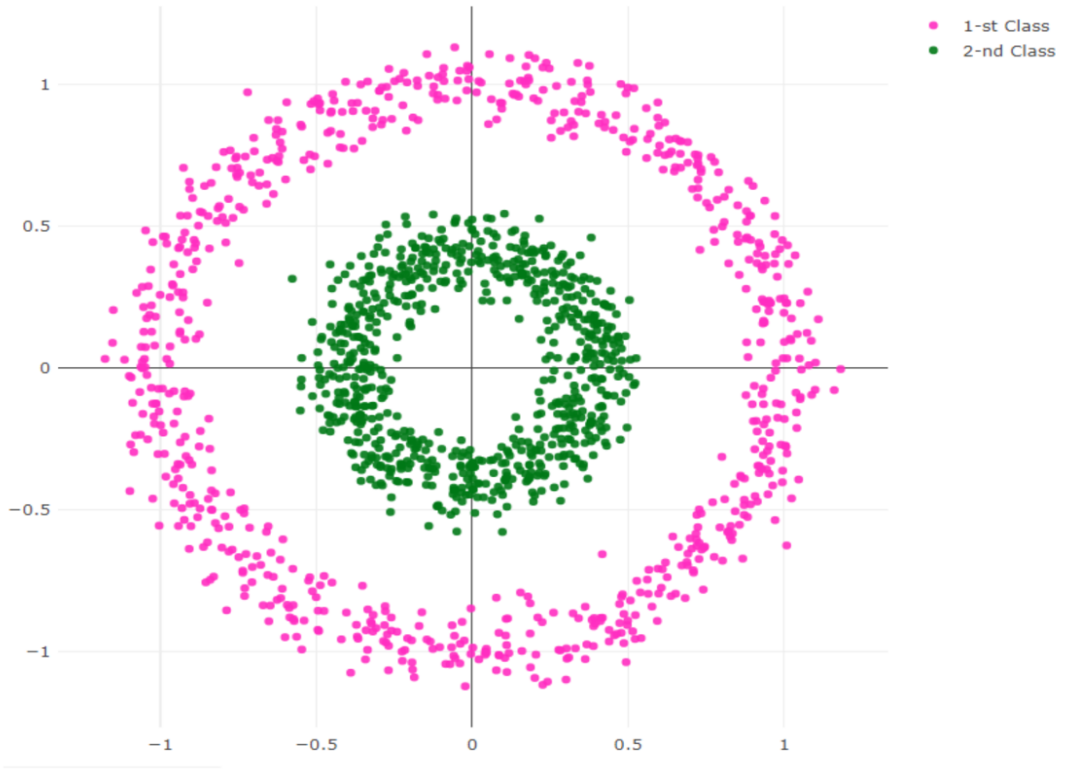


Рисунок 3.3 – Вибірка двох окружностей

Наступим кроком проведемо навчання моделі машини опорних векторів на цьому наборі даних і вивчимо вплив вибору різних параметрів на її функціональну працездатність. У моделі машини опорних векторів існує два ключових параметри, які мають значущий вплив на процес навчання. Окрім того, необхідно визначити оптимальну ядерну функцію, через яку буде відбуватися навчання моделі.

Зазвичай для ядра можуть бути вибрані різні опції, такі як поліноміальне, радіально базисне або лінійне ядро. Лінійне ядро відповідає відсутності ядра і означає використання даних без будь-якої додаткової обробки. У даному випадку розглядається можливість використання

поліноміального або радіально базисного ядра. Проте перед вибором ядра, ми проведемо експеримент для оцінки, як лінійне ядро працює на даному наборі даних (рис 3.4).

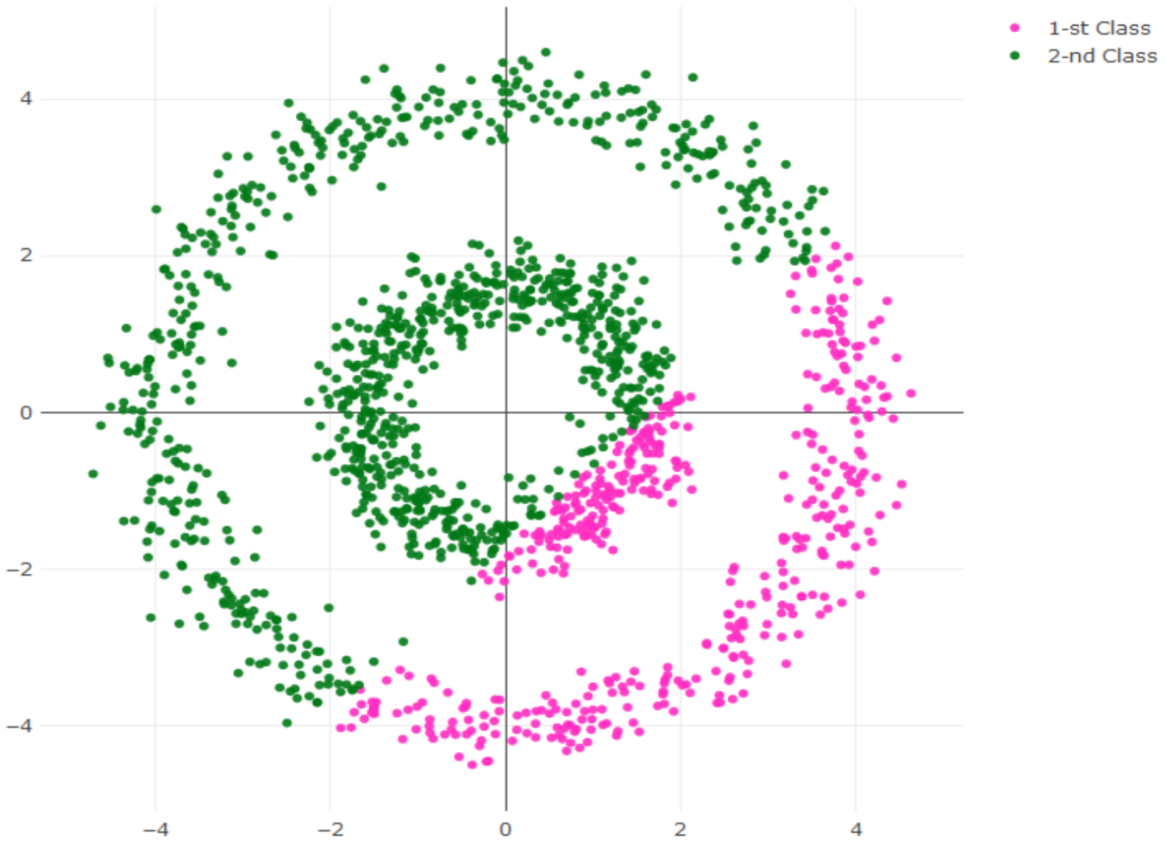


Рисунок 3.4 – Результат класифікації за допомогою SVM моделі з лінійним ядром

Наступним кроком розглянемо процес класифікації даного набору даних, використовуючи модель машини опорних векторів, де в якості ядра використовується радіально базисне ядро (2.20). В процесі оптимізації за допомогою градієнтного спуску, що оптимізує функцію (2.19), використовується параметр штрафу C за невірні класифікації, і цей параметр необхідно вибирати вручну.

Отже, для досягнення адекватної класифікації необхідно визначити параметр так, щоб класифікатор був надто загальним, але й не перенавчений.

Після модифікації вихідного шару машини опорних векторів, ми отримуємо ймовірності належності кожному з класів. З використанням цих ймовірностей ми можемо побудувати 3D-графік, який відображатиме ймовірність належності заданої точки до одного з класів.

Далі розглянемо вплив параметра σ з формули (2.20) на результати класифікації. Для цього нам потрібно вибрати параметр γ:

$$\boxed{\gamma = \frac{1}{2\sigma^2}}. \tag{3.1}$$

Збільшення параметра γ призведе до більшого перенавчання моделі або надто загального класифікатора. Початкове значення параметра γ вибране рівним 0.15. Розглянемо результати, а саме поверхню ймовірності належності до другого класу (рис 3.5) та результати класифікації (рис 3.6).

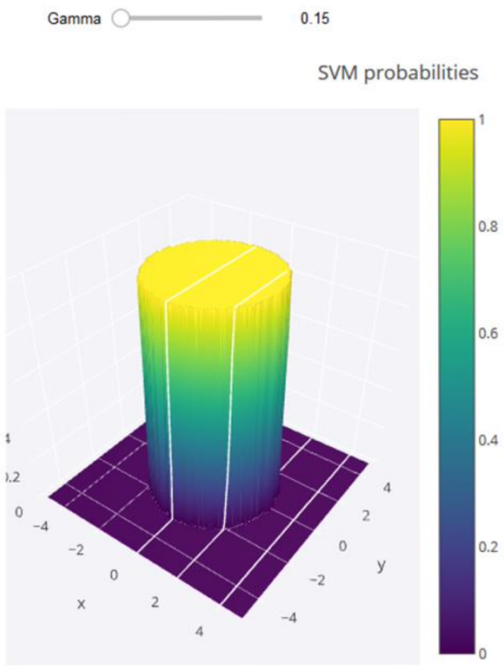


Рисунок 3.5 – Поверхня вірогідності настання 2-го класу

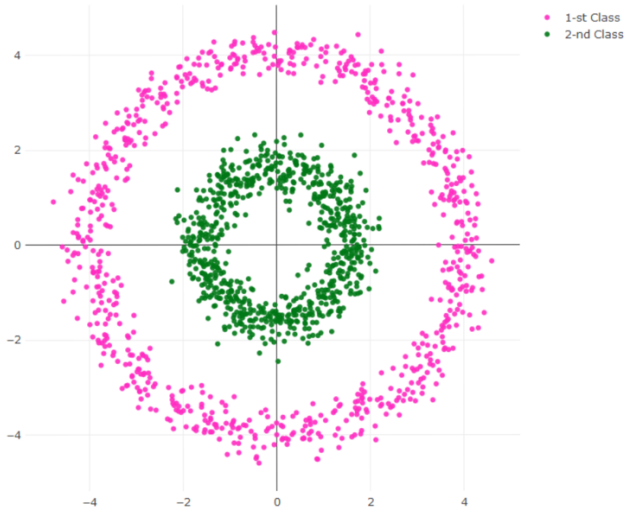


Рисунок 3.6 – Результат класифікації моделі

Отже, можемо зробити висновок, що вибір радіально базисної функції був обґрунтованим, і побудована поділяюча гіперплощина виглядає правильно. Проте, важливо відмітити, що всі спостереження, які розташовані поза межами зовнішнього кола, будуть призначені до першого класу, в той час як інші - до другого.

Також важливо враховувати вплив збільшення параметра γ. Давайте розглянемо результати роботи цієї моделі при значенні $\gamma = 10$ (рис 3.7).

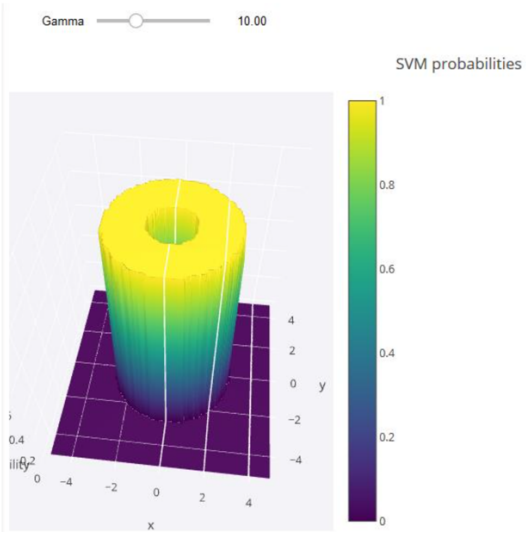


Рисунок 3.7 – Поверхня ймовірності настання 2-го класу при змінених параметрах

Можемо бачити що внутрішня область внутрішнього кола призначається до першого класу (рис 3.7), що відрізняється від попередньої моделі. Це спостереження підтверджує, що зі збільшенням значення параметра радіально базисної функції наша модель стає менш узагальненою і більш схильною до перенавчання на один з класів.

Наступним кроком є вибір параметру штрафу *C*, який визначатиме штраф за невірну класифікацію моделі під час навчання. На прикладі першої моделі, де параметр радіально базисної функції дорівнює 0.15, ми розглянемо вплив параметра *C* на результати класифікації. Розглянемо результат класифікації при малих значеннях C (рис 3.8), та результат коли параметр C дорівнює великому значенню (рис 3.9).

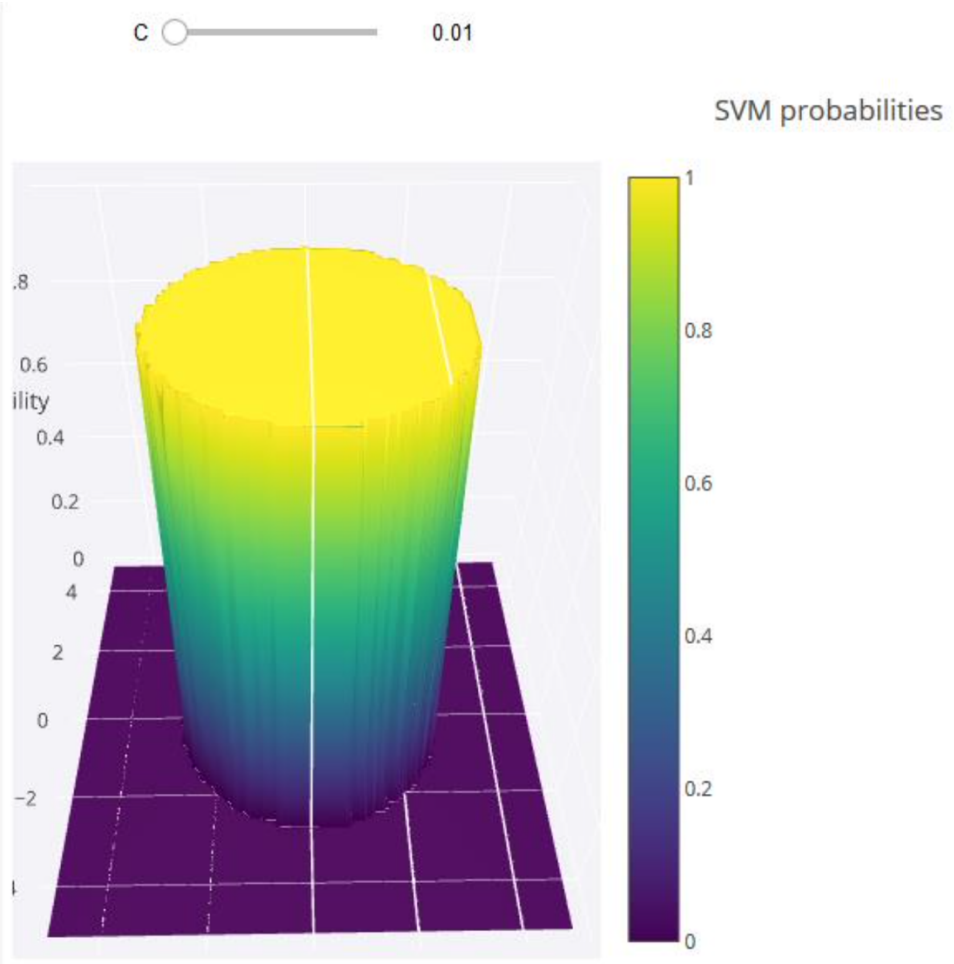


Рисунок 3.8 – Результати при малому значенню параметру C

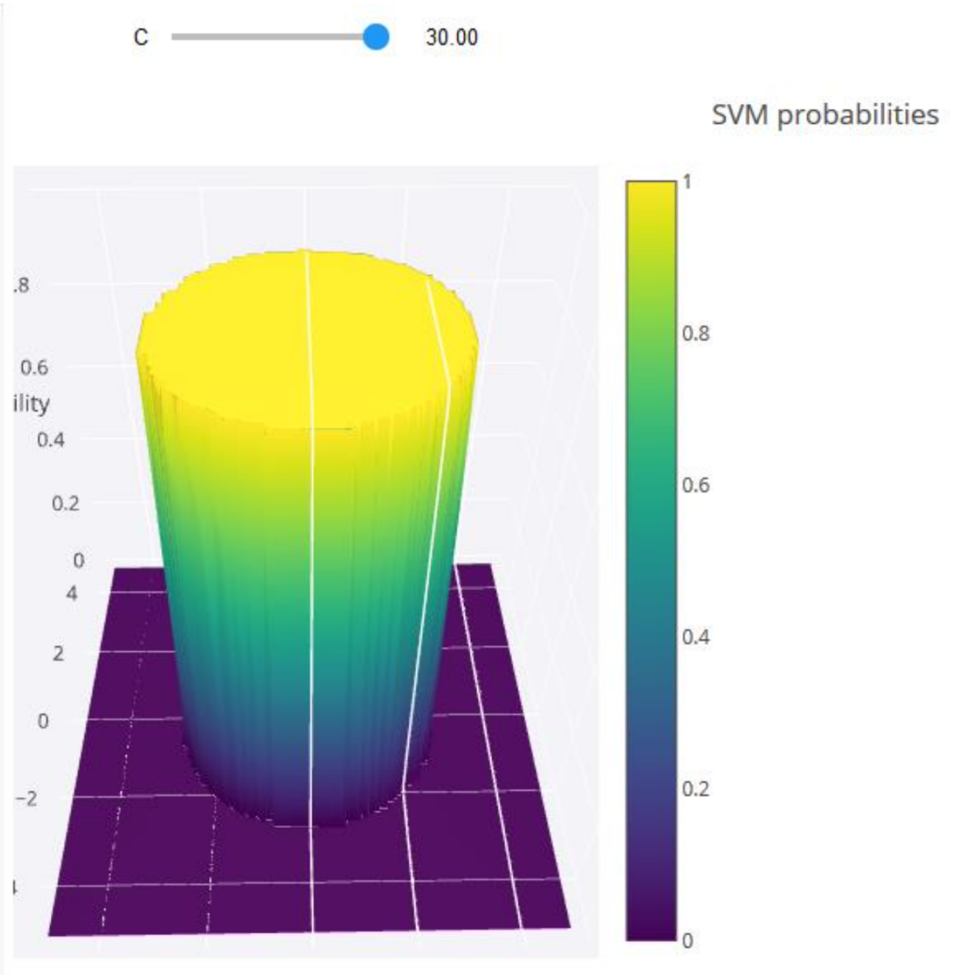


Рисунок 3.9 – Результати при великому значенню параметру C

Виходячи з результатів на вибірці з двома окружностями, можна припустити, що параметр штрафу не має значущого впливу на цей набір даних. Це може бути пов’язано з тим, що при поточних параметрах поділяюча гіперплощина швидко знаходиться і помилка під час навчання дуже мала.

Наступним прикладом для аналізу використаємо набір даних «Подвійна спіраль». Цей набір можна отримати з відкритих джерел або створити самостійно за допомогою функцій. Він складається з двох спіралей, які розташовані одна в одній. Розглянемо як виглядає цей набір даних (рис 3.10).

Наступним кроком перейдемо до побудови моделі SVM, яка буде використовувати для класифікації. Важливо відзначити, що даний простір

спостережень, можливо, не можна лінійно розділити. Проте, для перевірки, як модель машини опорних векторів з лінійним ядром веде себе в цьому випадку, ми побудуємо її і візуалізуємо результати.

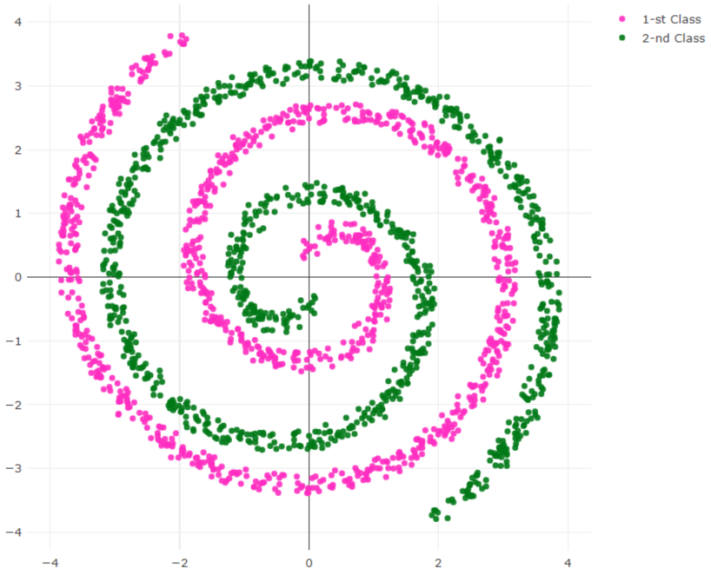


Рисунок 3.10 – Вибірка «Дві спіралі»

Розглянемо результат класифікація вибірки «Дві спіралі» за допомогою моделі машини опорних векторів з лінійним ядром (рис 3.11).

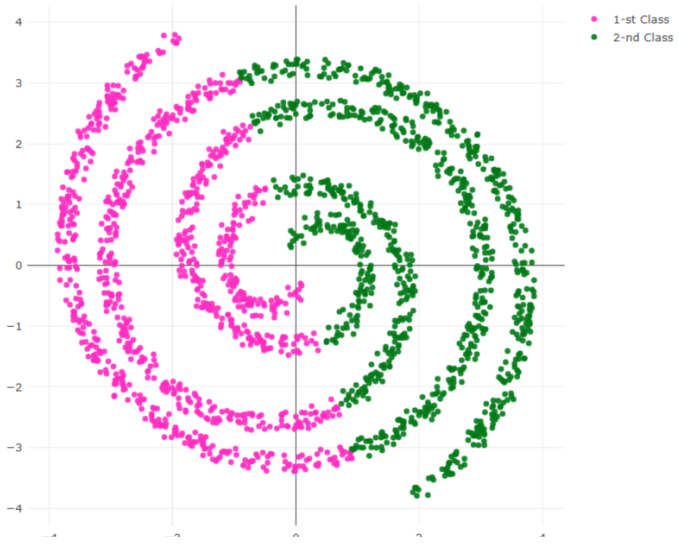


Рисунок 3.11 – Результати класифікації вибірки «Дві спіралі» з лінійним ядром класифікатора

Як бачимо (рис 3.11), лінійний класифікатор розділив простір спостережень на дві частини, намагаючись мінімізувати похибку класифікації. Проте лінійний класифікатор не досягає бажаного результату в цьому випадку, і тому варто розглянути використання радіально базисного ядра.

Спробуємо навчити класифікатор з радіально базисним ядром на цьому наборі даних. Щоб продемонструвати результати навчання класифікатора, побудуємо площину ймовірностей належності до одного з класів. Представлена площина (рис 3.12) ймовірностей належності до першого класу при параметрах штрафу та параметру радіально базисного ядра, які дорівнюють одиниці.

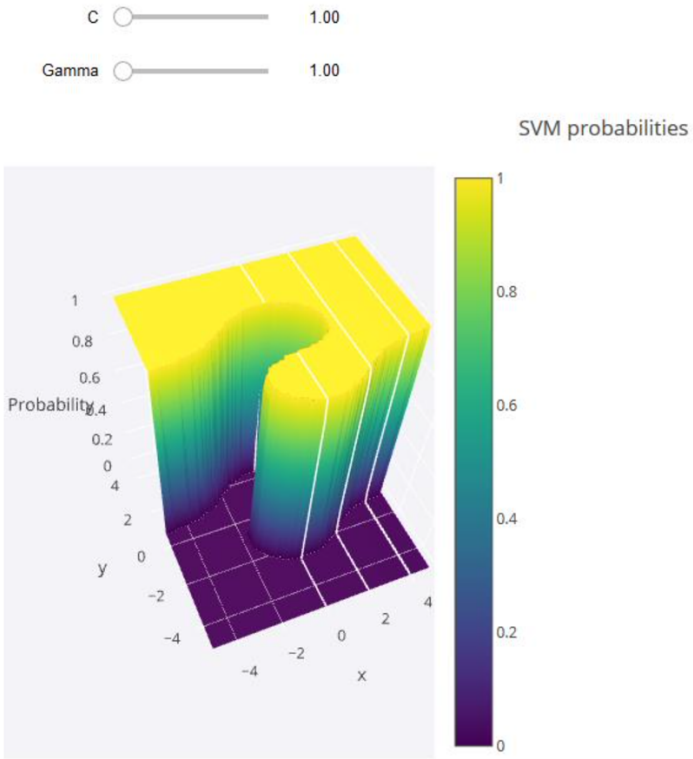


Рисунок 3.12 – Площина ймовірностей настання першого класу на вибірці «Дві спіралі»

Як видно, площина (рис 3.12) погано відображає цей набір даних, оскільки параметри моделі були вибрані недостатньо точно для ефективної класифікації (рис 3.13). Ця проблема походить від того, що малі значення параметрів штрафу та радіально базисної функції у складних моделях призводять до недостатньо навченої моделі, яка не може правильно узагальнити дані. Проте, з іншого боку, збільшення цих параметрів може призвести до перенавчання моделі.

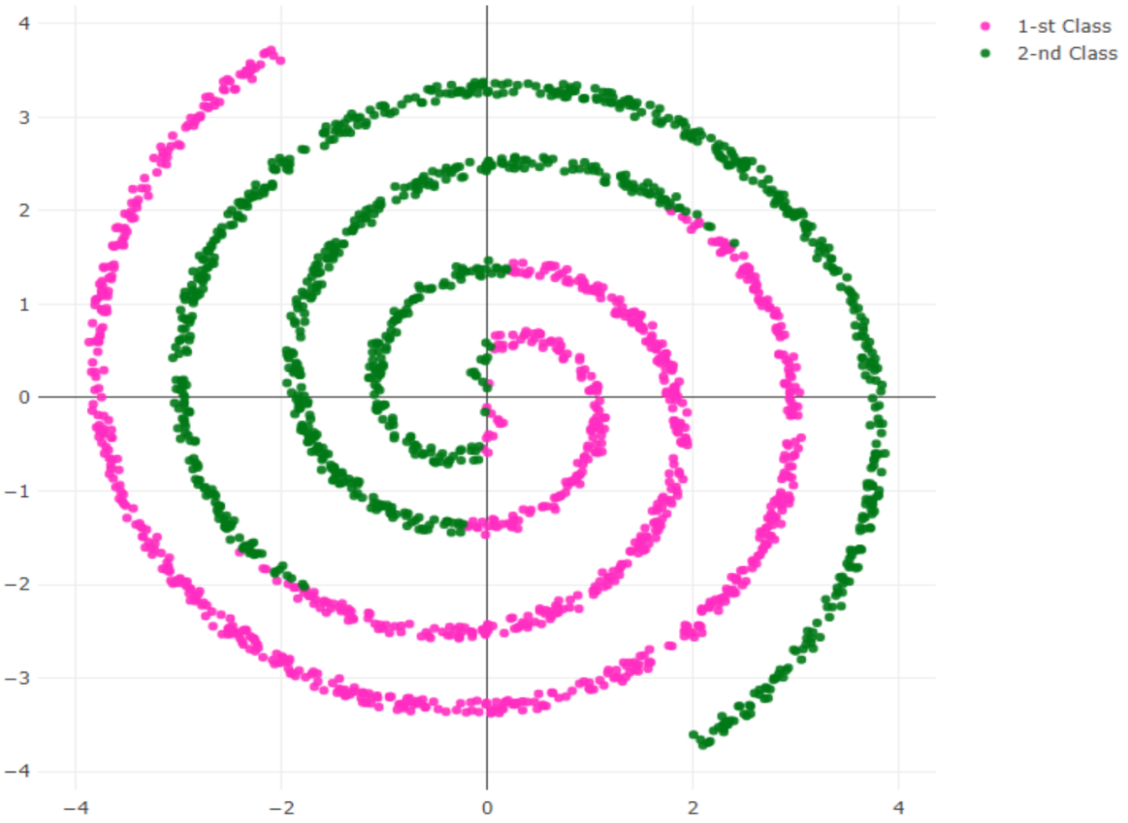


Рисунок 3.13 – Результат класифікації моделі з невеликими значеннями параметрів

Як видно з попереднього зображення (рис 3.13), класифікація класів проведена недостатньо точно, навіть для спостережень, що використовувалися під час навчання моделі. Цю неефективність можна пояснити тим, що модель є дуже загальною і не здатна створити правильну роздільну гіперплощину через обмеженість параметрів моделі.

Для вирішення цієї проблеми необхідно збільшити параметр штрафу (рис 3.14) або збільшити параметр радіально базисної функції (рис 3.15). Проведемо аналіз графіка площини ймовірностей віднесення до першого класу при збільшенні параметра штрафу та параметра радіально базисної функції відповідно.

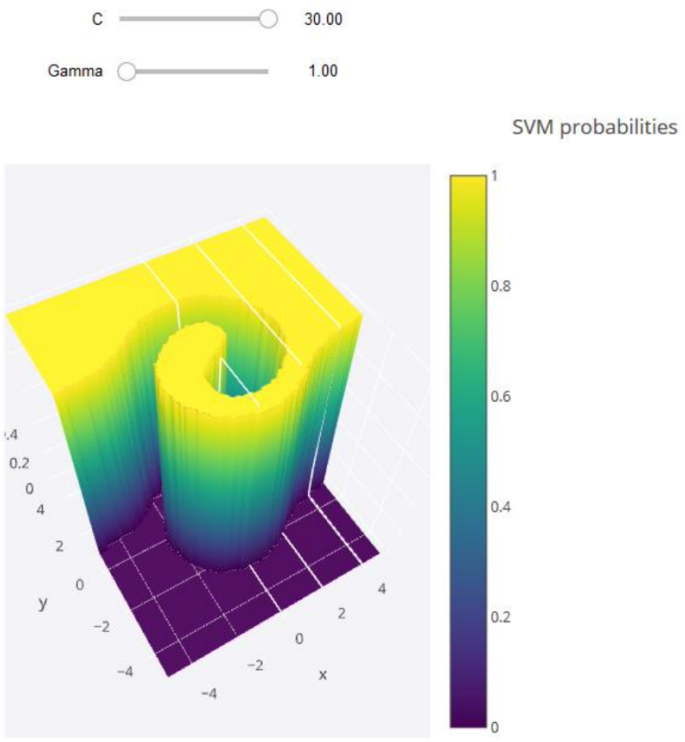


Рисунок 3.14 – Результати при збільшенні параметру штрафів

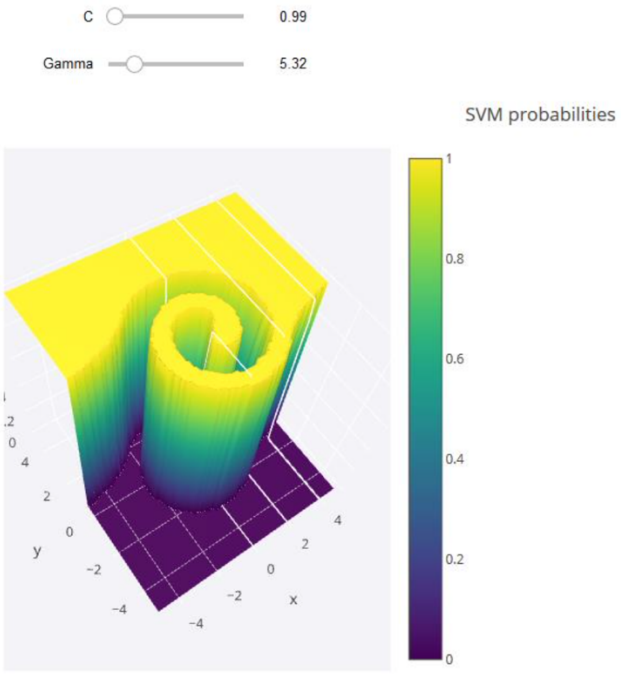


Рисунок 3.15 – Результати при збільшенні параметру радіально базисної функції

Аналізуючи результати, наведені на графіках на рисунку 3.14 та рисунку 3.15, можна визначити, що зміна обох параметрів має значний вплив на результати класифікації. На рисунку 3.15 видно, що зі збільшенням значення параметра радіально базисної функції модель стає більш "прив’язаною" до конкретних спостережень у вибірці. Проте при виборі оптимального значення цього параметра, модель стає більш загальною, що дозволяє правильно класифікувати спостереження, які не були включені у навчальний набір. Занадто великі значення параметрів можуть призвести до втрати здатності моделі до узагальнення, і вона може правильно класифікувати лише ті спостереження, які були у навчальному наборі (рис 3.16).

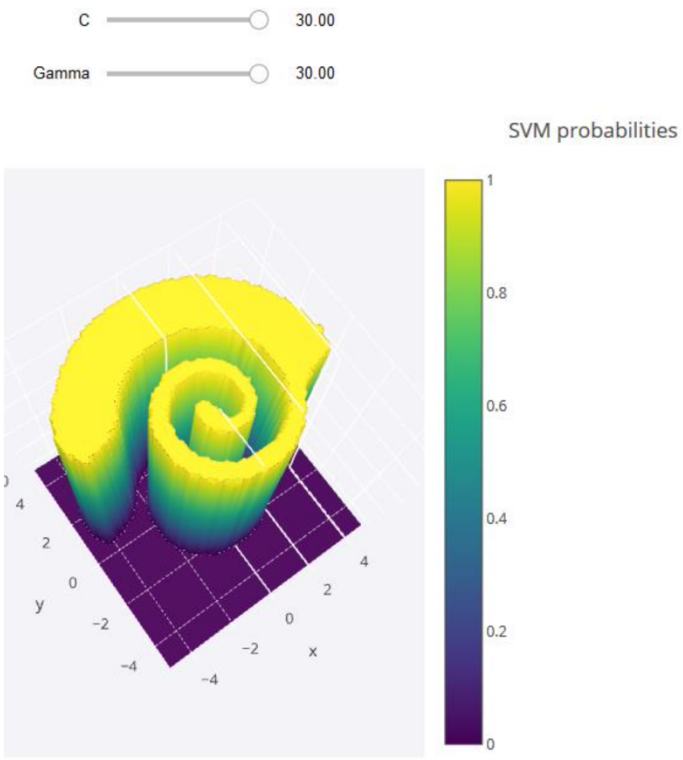


Рисунок 3.16 – Результат моделі при великих значеннях параметрів

Додатково, розглянемо працездатність алгоритму машини опорних векторів на більш складних наборах даних. У даному контексті ми розглядаємо набір даних «Breast Cancer Wisconsin (Diagnostic) Data Set», який містить інформацію про рак молочних залоз [44-45]. Цей набір даних включає в себе різні характеристики, що були обчислені на основі цифрових результатів тонколічильної аспіраційної пункційної біопсії, проведеної на молочній тканині.

Для кожної клітини обчислюються різні характеристики, на основі яких формується цей набір даних. Серед таких характеристик можна вказати:

– радіус (середню відстань від центру клітини до точок по її периметру);

– текстуру (стандартне відхилення масштабів);

– периметр;

– область;

– гладкість (локальне відхилення довжини радіуса);

– компактність;

– увігнутість (виразність увігнутих частин контуру);

– кількість увігнутих точок (кількість увігнутих частин контуру);

– симетрія.

На основі цих характеристик формується набір даних, для кожної з яких обчислюються середнє значення, найгірше значення та стандартна похибка серед клітин. Кожній клітині також надається відповідний атрибут: «M» - для злоякісних клітин, «B» - для доброякісних. Оскільки в наборі даних багато атрибутів, для візуалізації використовується метод зменшення розмірності даних за допомогою аналізу головних компонент. Проте перед застосуванням цього методу, дані піддаються нормалізації. Розглянемо результати зменшення розмірності даних методом аналізу головних компонент (рис 3.17).

Як можна спостерігти, лінійна поділяюча поверхня не є можливою для цієї скомпресованої вибірки. Проте, ми можемо чітко визначити, що більшість спостережень, розташованих праворуч, належать до злоякісних новоутворень.

При спробі скористатися тривимірною компресією даних, ми можемо спробувати виділити кожен клас окремо і побудувати поділяючу гіперплощину для більш точної класифікації.

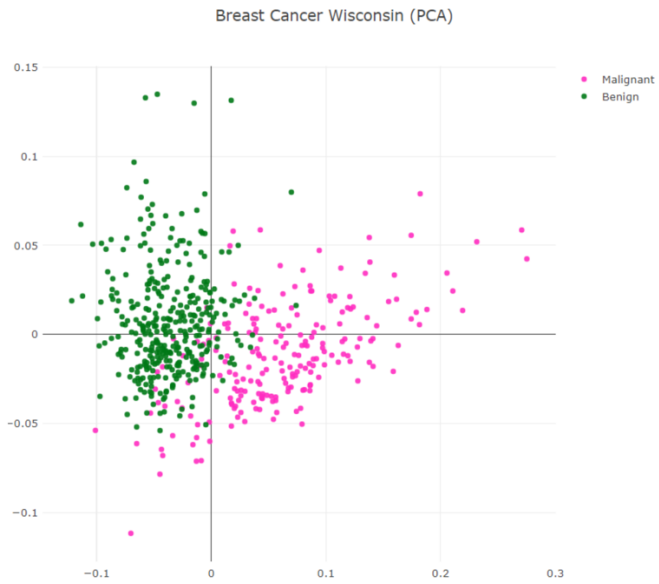


Рисунок 3.17 – Скомпресована вибірка «Breast Cancer Wisconsin»

Як можна відзначити (рис 3.17), поділити цю скомпресовану вибірку лінійно не можливо. Проте ми можемо чітко спостерігати, що більшість спостережень, що знаходяться справа, відносяться до злоякісних новоутворень. Отже ми можемо провели компресію даних в тривимірний простір, у надії, що це допоможе нам виділити кожен клас окремо та побудувати поділяючу гіперплощину для більш точної класифікації (рис 3.18).

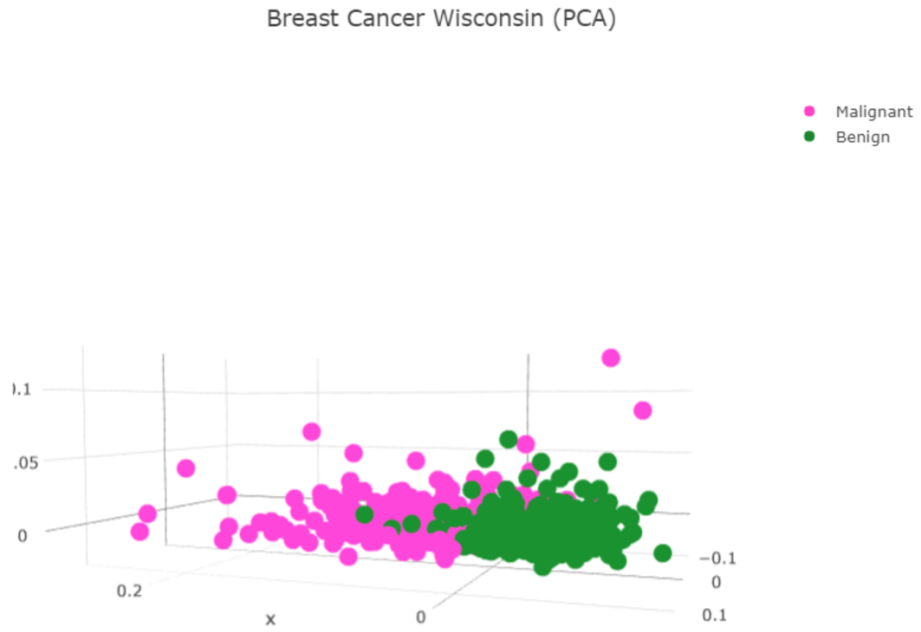


Рисунок 3.18 – Скомпресована вибірка у тривимірний простір

Отже, згідно з вищевикладеними результатами, можна відзначити, що розташування доброякісних новоутворень у вибірці є концентрованим приблизно в одному місці. Ця закономірність натякає на можливість розділення даної вибірки гіперплощиною з метою успішної побудови моделі класифікації.

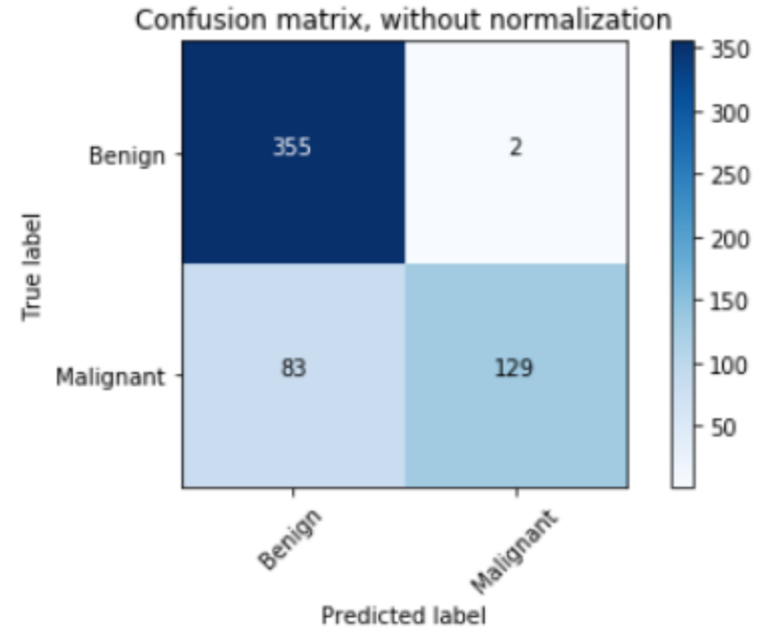
Створимо модель машини опорних векторів для даної вибірки без компресії, використовуючи лінійне ядро (рис 3.19).

```
Model score: 0.8506151142355008
Malignant score: 0.6084905660377359 Benign score: 0.9943977591036415
```

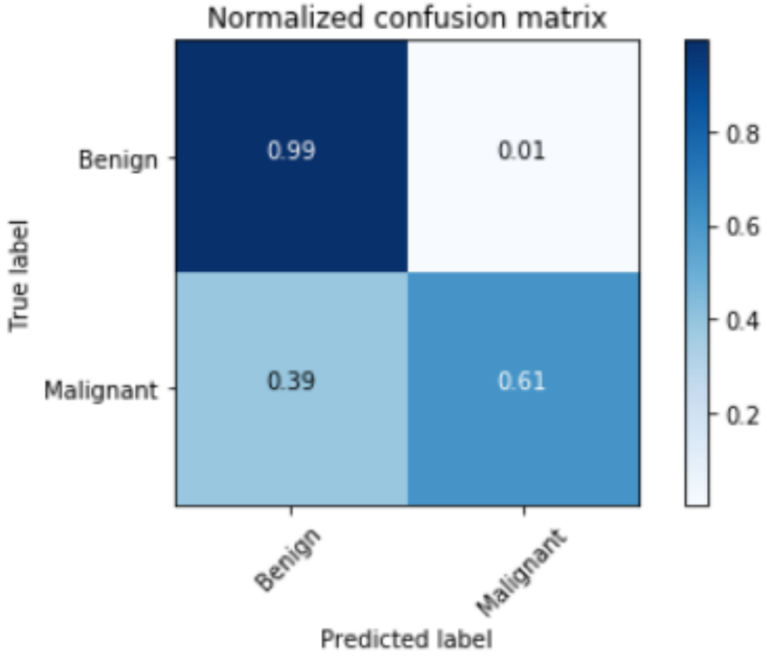
Рисунок 3.19 – Результати класифікації

На основі цих результатів можна зазначити, що точність моделі становить 85%. Серед злоякісних новоутворень правильно класифіковано приблизно 60% випадків, в той час як серед доброякісних новоутворень точність становить близько 99,5%. За цими даними можна зробити висновок, що класифікація злоякісних новоутворень за допомогою моделі машини опорних векторів дає недостатньо задовільні результати, оскільки більше половини випадків були невірно класифіковані. Проте, класифікація доброякісних новоутворень має високу точність.

Враховуючи низьку точність класифікації злоякісних новоутворень, можна зробити висновок, що побудована модель не є придатною для практичного застосування, так як точність 85% не забезпечує адекватних результатів для інформування пацієнтів або для використання в діагностиці. Ми також провели побудову матриці помилок на основі результатів даної моделі (рис 3.20).



а)



б)

Рисунок 3.20 – Матриці похибки машини опорних векторів з лінійним ядром:

а) – кількісна оцінка; б) – оцінка в процентному співвідношенні

На рисунку 3.21 показано як виглядає простір спостережень при класифікації за допомогою машини опорних векторів з лінійним ядром.

Отже, враховуючи отримані результати, можна прийти до висновку, що використання лінійного ядра для даної моделі не є оптимальним рішенням.

Спробуємо використати радіально базисне ядро замість лінійного (рис 3.22).

```
Model score: 1.0
Malignant score: 1.0 Benign score: 1.0
```

Рисунок 3.22 – Результат класифікації з радіально базисним ядром

Як видно з отриманих результатів, модель з радіально базисним ядром демонструє 100% правильних класифікацій, що практично неможливо для реальних даних. Це свідчить про те, що модель страждає від перенавчання та занадто сильно пристосувалася до навчальних даних, недостатньо узагальнюючи їх. Щоб перевірити адекватність моделі з радіально базисним ядром, розділимо вибірку на навчальну та тестову, де розмір тестової вибірки складає 20% від загального обсягу.

При встановленні параметра радіально базисної функції на рівні 0.1, що відповідає попередній моделі (рис 3.23).

```
Model score: 0.5789473684210527
Malignant score: 0.0 Benign score: 1.0
```

Рисунок 3.23 – Результати класифікації на тестовій вибірці

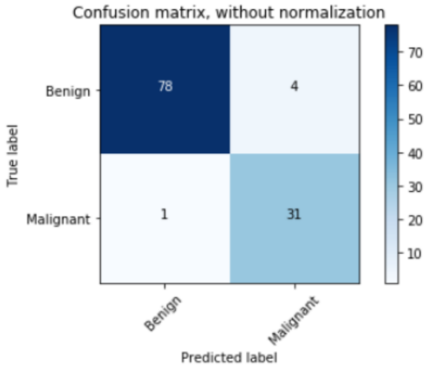
Отримана модель призначає всі спостереження класу доброякісних новоутворень. Це свідчить про те, що попередня модель була перенавчена та надто «запам’ятала» всі спостереження класу злоякісних новоутворень, а всі інші автоматично класифікувалися як доброякісні. Для уникнення перенавчання необхідно зменшити параметр радіально базисної функції, що, як показано на попередніх вибірках, допомагає зробити модель більш

узагальненою. Результати моделі з параметром радіально базисної функції, що дорівнює 0.001 (рис 3.24).

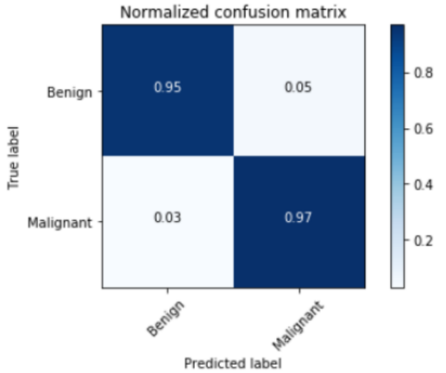
```
Model score: 0.956140350877193
Malignant score: 0.96875 Benign score: 0.9512195121951219
```

Рисунок 3.24 – Результат моделі зі модифікованим параметром радіально базисної функції

Ці результати свідчать про те, що дана модель є більш узагальненою в порівнянні з попередньою. Точність на тестовій вибірці становить більше 95%, що є високим показником. Майже 97% злоякісних новоутворень було правильно визначено. Матриці помилок представлені на рисунках 3.25.



а)



б)

Рисунок 3.25 – Матриці похибок моделі з радіально базисним ядром: а) – кількісна оцінка; б) – оцінка в процентному співвідношенні

Розглянемо графік класифікації даної вибірки (рис 3.26).

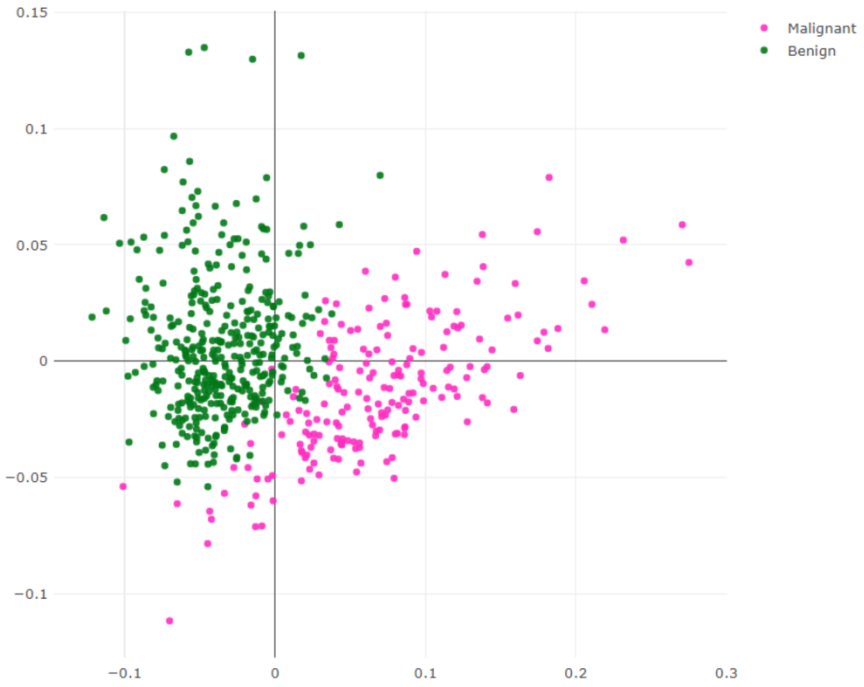


Рисунок 3.26 – Результати класифікації моделі з налаштованим параметром радіально базисної моделі

Таким чином, аналізуючи результати всіх розглянутих моделей під час експериментів, можна зробити висновок, що машина опорних векторів демонструє адекватні результати при класифікації, проте лише при належному налаштуванні параметрів моделі.

ВИСНОВКИ

В ході даної кваліфікаційної роботи була розглянута предметна область і визначена формальна постановка задачі, включаючи поняття навчання з учителем і класифікації. Поверхнево були описані основні алгоритми класифікації, включаючи метод машини опорних векторів і його принцип роботи. Для бінарної класифікації і мультикласової класифікації використовується підхід «один проти всіх». Спостереженням, що надходить на вхід моделі, присвоюється клас залежно від розташування спостереження від поділяючої гіперплощини, яка розділяє простір ознак на дві частини. Для мультикласової класифікації створюється кілька моделей, кількість яких дорівнює кількості класів.

Основною метою даної роботи було розв’язання проблеми обробки даних з метою класифікації. Класифікатор на основі машини опорних векторів повинен бути навчений на прецедентах, що виявляють собою пару – спостереження-відповідь. Під час проведення експериментальних досліджень було розглянуто три вибірки даних, на основі яких було продемонстровано, як вибір параметрів моделі машини опорних векторів впливає на результати обробки даних та їх класифікацію. За допомогою побудованих графіків були наглядно відображені результати проведеного досліду.

З цих результатів можемо зробити висновок, що коректна обробка даних та класифікація напряму залежать від точного підбору параметрів моделі. Для досягнення більш точних результатів у майбутньому дослідженні може бути важливим подальше вдосконалення і налаштування параметрів моделі машини опорних векторів. Результати цієї роботи можуть бути корисними при розробці систем класифікації і обробки даних в різних сферах застосування, таких як медицина, фінанси, інформаційна безпека та інші.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Nils, J. N. (1998). Introduction to Machine learning. Robot. Lab. Dep. Comput. Sci. Stanf. Univ, 1, 1-209.

2. Ben-Hur, A., & Weston, J. (2010). A user's guide to support vector machines. Data mining techniques for the life sciences, 223-239.

3. Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering, 160(1), 3-24.

4. Mhaskar, H. N., & Micchelli, C. A. (1993). How to choose an activation function. Advances in Neural Information Processing Systems, 6.

5. Hicken, J., Alonso, J., & Farhat, C. (2012). Introduction to multidisciplinary design optimization.

6. Wang, L. (Ed.). (2005). Support vector machines: theory and applications (Vol. 177). Springer Science & Business Media.

7. Haykin, S. (1998). Neural networks: a comprehensive foundation. Prentice Hall PTR.

8. Suykens, J. A. (2001). Support vector machines: a nonlinear modelling and control perspective. European Journal of Control, 7(2-3), 311-327.

9. Duda, R.O., Hart, P.E., (1973). Pattern Classification and Scene Analysis, Wiley.

10. Fletcher, R. (2008). Practical Methods of Optimization, Wiley.

11. Rumelhart, D.E., McClelland, J.L. (1986). Parallel Distributed Processing: Explorations in the Microstructure of Cognition Cambridge – MA.

12. Ye, L., Keogh, E. (2009). Time series shapelet: a new primitive for data mining, KDD `09 Proceedings of 15th ACM SIGKDD international conference on Knowledge discovery and data mining.

13. Bertsekas, D.P. (2017). Dynamic Programming and Optimal Control, Athena Scientific.

14. Reitz, K., Schlusser, T. (2016). The Hitchhiker's Guide to Python: Best Practices for Development, O'Reilly Media.

15. Müller, A.C., Guido, S. (2016). Introduction to Machine Learning with Python: A Guide for Data Scientists, O'Reilly Media.

16. Yoon, S.T., Hwang, E., (2007). A leaf image retrieval scheme based on partial dynamic time wraping and two-level filterin, CIT: 7th IEEE International Conference on Computer and Information Technology.

17. Lowe, D.G. (2004). Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision.

18. Hills, J., Baranauskas, E., Lines, J., Mapp, J., Bagnall, A. (2014). Time-series Classification with Shapelets, Journal Data Mining and Knowledge Discovery.

19. URL: https://docs.python.org/3.6/whatsnew/3.6.html#improved- modules (дата звернення: 02.10.2023).

20. Toxic Comments Dataset URL: https://www.kaggle.com/c/jigsaw- toxic-comment-classification-challenge (дата звернення: 25.10.2023)

21. Ramos, J. (2003, December). Using tf-idf to determine word relevance in document queries. In Proceedings of the first instructional conference on machine learning (Vol. 242, No. 1, pp. 29-48).

22. Tvoroshenko I., Pomazan V., Gorokhovatskyi V., and Kobylin O. (2023) Application of video data classification models using convolutional neural networks, International Journal of Academic and Applied Research, 7(11), pp. 134-145.

23. Rabotiahov, A., Kobylin, O., Dudar, Z., & Lyashenko, V. (2018, February). Bionic image segmentation of cytology samples method. In 2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET) (pp. 665-670). IEEE.

24. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень.

25. Работягов, А. В., Ляшенко, В. В., & Кобылин, О. А. (2016). Сегментация сложных изображений цитологических препаратов.

26. Lyashenko, V., Mohammad, A., & Kobylin, O. (2015). Experiments with Fusion of Images with Use of Wavelet Transformation in Problems of the Text Information Analysis.

27. Kobylin, O., Vyskrebentseva, S., & Petrova, R. (2019). Обробка даних, що містять пропуски в задачах кластеризації. Системи управління, навігації та зв'язку. Збірник наукових праць, 5(57).

28. Oleg, K., Sergii, M., & Mykhailo, S. (2017, October). Video Clustering via Multidimensional Time-Series Analysis. In Proceedings of the 9th International Conference on Information Management and Engineering (pp. 60-63). ACM.

29. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2018, August). Representative Based Clustering of Long Multivariate Sequences with Different Lengths. In 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP) (pp. 545-548). IEEE.

30. Bodyanskiy, Y., Kobylin, I., Rashkevych, Y., Vynokurova, O., & Peleshko, D. (2018, February). Hybrid fuzzy-clustering algorithm of unevenly and asynchronously spaced time series in computer engineering. In 2018 14th International Conference on Advanced Trends in Radioelecrtronics, 60 Telecommunications and Computer Engineering (TCSET) (pp. 930-935). IEEE.

31. Bodyanskiy, Y., Vynokurova, O., Kobylin, I., & Kobylin, O. (2016). Adaptive fuzzy clustering of short time series with unevenly distributed observations in Data Stream Mining tasks. Information Technology and Management Science, 19(1), 23-28.

32. Lyashenko V., Kobylin O., Selevko O. (2020) Wavelet Analysis and Contrast Modification in the Study of Cell Structures Images. International Journal of Advanced Trends in Computer Science and Engineering. 9(4). – 4701-4706.

33. Mashtalir, V., Ruban, I., & Levashenko, V. (Eds.). (2019). Advances in Spatio-Temporal Segmentation of Visual Data (Vol. 876). Springer Nature.

34. Kobylin, O., & Lyashenko, V. (2016). Contrast Modification as a Tool to Study the Structure of Blood Components.

35. Kobylin, O. A., Gorokhovatskyi, V. O., Tvoroshenko, I. S., & Peredrii, O. O. (2020). The application of non-parametric statistics methods in image classifiers based on structural description components. Telecommunications and Radio Engineering, 79(10).

36. Kobylin, O., & Lyashenko, V. (2014). Comparison of standard image edge detection techniques and of method based on wavelet transform.

37. Gorokhovatskiy, V. A., Kobylin, O. A., & Kulikov, Y. A. (2015). Application of Granulation of Feature Descriptions in Structural Image Recognition. Telecommunications and Radio Engineering, 74(6).

38. Kuzminska, O., Mazorchuk, M., Morze, N., & Kobylin, O. (2019, June). Digital learning environment of ukrainian universities: The main components to influence the competence of students and teachers. In International Conference on Information and Communication Technologies in Education, Research, and Industrial Applications (pp. 210-230). Springer, Cham.

39. Kinoshenko, D., Kobylin, O., Mashtalir, S., & Stolbovyi, M. (2019, March). Metric video retrieval speedup by irrelevant data elimination. In Eleventh International Conference on Machine Vision (ICMV 2018) (Vol. 11041, pp. 176-183). SPIE. 61

40. Бодянский, Е. В., Винокурова, Е. А., Пелешко, Д. Д., Кобылин, И.О., & Кобылин, О. А. (2017). Нечёткая кластеризация временных рядов с неравномерными и асинхронными тактами квантования. Системи обробки інформації, (5), 47-54.

41. Lyashenko, V., Matarneh, R., Kobylin, O., & Putyatin, Y. (2016). Contour detection and allocation for cytological images using Wavelet analysis methodology.

42. Lyashenko, V., Kobylin, O., & Ahmad, M. A. (2014). General methodology for implementation of image normalization procedure using its wavelet transform.

43. Gorokhovatskyi V., Tvoroshenko I., Kobylin O., and Vlasenko N. (2023) Search for visual objects by request in the form of a cluster representation for the structural image description, Advances in Electrical and Electronic Engineering, 21(1), pp. 19-27.

44. Yakovleva, O., Kovtunenko, A., Liubchenko, V., Honcharenko, V., & Kobylin, O. (2023). Face Detection for Video Surveillance-based Security System (COLINS-2023). In CEUR Workshop Proceedings (Vol. 3403, pp. 69-86).

45. Tvoroshenko I., Gorokhovatskyi V., Kobylin O., and Tvoroshenko A. (2023) Application of deep learning methods for recognizing and classifying culinary dishes in images, International Journal of Academic and Applied Research, 7(9), pp. 57–70.