Міністерство освіти і науки України
Харківський національний університет
радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

АТЕСТАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Нечітка машина опорних векторів для класифікації.
.
(тема)

Виконав:
студент 2 курсу, групи СШІм-18-2
Косьмін Д.С.
(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 (СШІ)
(повна назва спеціалізації)

Керівник к.т.н. Узлов Д.Ю.
(посада, прізвище, ініціали)

Допускається до захисту

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

2020 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-проіфесійна або освітньо-наукова)

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

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

ЗАВДАННЯ
НА АТЕСТАЦІЙНУ РОБОТУ

студентові Косьміну Данилу Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Нечітка машина опорних векторів для класифікації данних
затверджена наказом університету від 30 березня 2020 р. № 480 Ст
2. Термін подання студентом роботи до екзаменаційної комісії ____ _____ 20__ р.
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

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 галузі і постановка завдання	31.03.2020 – 15.04.2020	виконанно
2	Розробка методу нечіткої машини опорних векторів для класифікації	15.04.2020 – 20.04.2020	виконанно
3	Створення імітаційної моделі	20.04.2020 – 30.04.2020	виконанно
4	Тестування і опрацювання імітаційної моделі	29.04.2020 – 07.05.2020	виконанно
5	Оформлення пояснювальної записки	20.04.2020 – 08.05.2020	виконанно
6	Оформлення графічних матеріалів та візуалізація результатів	01.05.2020 - 08.05.2020	виконанно
7	Попередній захист	14.05.2020	виконанно
8	Захист перед ЕК	19.05.2020	

Дата видачі завдання _____ _____ 20 __ р.

Студент _____
(підпис)

Керівник роботи _____ _____к.т.н. Узлов Д.Ю._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Записка пояснювальна: 80 с., 36 рис., 1 табл., 2 дод., 22 джерел.

FUZZY, PYTHON, SVM, НАВЧАННЯ З УЧИТЕЛЕМ, НАВЧАННЯ МОДЕЛІ, ПОДІЛЯЮЧА ГІПЕРПЛОЩИНА, ПРОБЛЕМА ПЕРЕНАВЧАННЯ, ПАРАМЕТРИ МОДЕЛІ

Об’єкт дослідження – процес обробки даних за допомогою машини опорних векторів.

Мета роботи – створення програмного засобу, який реалізує мультикласифікацію даних за допомогою нечіткої машини опорних векторів, реалізувати графічні візуалізації роботи алгоритму та показати вплив вибору параметрів моделі на точність моделі.

Методи дослідження – аналіз літератури, Internet ресурсів, документацій і практична реалізація.

Результат атестаційної роботи – у результаті проведеної роботи був створений програмний засіб обробки даних для класифікації за допомогою машини опорних векторів та візуалізації даних та результатів, яка задовольняє вимогам технічного завдання. Для реалізації було використано: Python, Jupyter Notebook, Plotly, Scikit-learn.

РЕФЕРАТ

Пояснительная записка: 80 с., 36 рис., 1 табл., 2 прил., 22 источник.

FUZZY, PYTHON, SVM, НАВЧАННЯ З УЧИТЕЛЕМ, НАВЧАННЯ МОДЕЛІ, ПОДІЛЯЮЧА ГІПЕРПЛОЩИНА, ПРОБЛЕМА ПЕРЕНАВЧАННЯ, ПАРАМЕТРИ МОДЕЛІ

Объект исследования – процесс обработки данных с помощью машины опорних векторов.

Цель работы - создание программного средства, реализующий мультиклассификацию данных с помощью нечеткой машины опорных векторов, реализовать графические визуализации работы алгоритма и показать влияние выбора параметров модели на точность модели.

Методы исследования – анализ литературы, Internet ресурсов, документаций и практическая реализация.

Результат аттестационной работы – в результате проведенной работы было создано программное средство обработки данных для классификации с помощью машины опорных векторов и визуализации данных и результатов, которая удовлетворяет требованиям технического задания. Для реализации были использованы: Python, Jupyter Notebook, Plotly, Scikit-learn.

ABSTRACT

Note explanatory: 80 p., 36 fig., 1 tabl., 2 ann., 22 sources.

FUZZY, HYPERPLANE, MODEL PARAMETERS, OVERFITTING PROBLEM, PYTHON, SUPERVISED LEARNING, SVM, TRAINING MODELS

The object of the research – the process of processing data using a support vector machines.

The purpose of the work is to create a software tool that implements the classification of data using a support vector machines, implement graphical visualizations of the algorithm's work and show the influence of the choice of model parameters on the accuracy of the model.

Methods of the research – analysis of literature, Internet resources, documentation and practical implementation.

The result of attestation work – the programmed data processing tool was created for multiclassification using a fuzzy support vector machines and visualization of data and results that meets the requirements of the specification. The following was used for implementation: Python, Jupyter Notebook, Plotly, Scikit-learn.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

kNN – k-Nearest Neighbors – k-найближчих сусідів;

PCA – Principal component analysis – Метод головних компонентів;

SVM – Support Vector Machines – Машина опорних векторів.

ВСТУП

Класифікація даних являється одною із найважливіших задач машинного навчання та інтелектуального аналізу даних. Метою класифікації є визначення класу об'єкта, до якого він належить, ґрунтуючись на даних навчальної вибірки класифікатору. Набором даних для класифікатора можуть служити різноманітні вибірки зібраних даних о певних об’єктах, матриці відстаней, зображення. Не дивлячись на різноманітні дані, які можуть поступати на класифікатор, головною метою є диференціація цього об’єкту на один із класів, на яких проходило навчання.

Алгоритми класифікації відносяться до одного із розділів машинного навчання, який називається навчанням з учителем. Навчання з учителем передбачає, що на вхід алгоритму, крім навчальних даних, буде надходити множина відповідей, яким ці дані відповідають. Таким чином, перед алгоритмами, які відносяться до алгоритмів навчання з учителем, стоїть завдання відновлення залежності між об'єктами, які подаються на вхід, і їх відповідями.

На сьогоднішній день існує велика кількість методів класифікації, які відрізняються підходами до пошуку залежностей у даних для проведення успішної класифікації. Успішна класифікація об'єктів залежить не тільки від правильного вибору методу класифікації, а й від правильного підходу до предобробці даних.

Однією з глобальних проблем алгоритмів навчання з учителем є проблема перенавчання. Перенавчання моделі – це явище, в ході якого, модель класифікації, натренований на прикладах з навчальної вибірки, є сильно «заточеною» під навчальні дані, але є непридатною для адекватної класифікації даних, які не беруть участі в навчанні. Тому, для алгоритмів класифікації необхідна не тільки здатність запам'ятовувати данні, які подаються на вхід моделі, а й узагальнювати залежності, які виникають

всередині даних.

Крім перенавчання моделі може виникнути проблема недонавчання. В даному випадку, класифікатор буде занадто узагальненим і давати невірні результати класифікації. Недонавчання може бути пов'язано з малою кількістю даних в навчальній вибірці, або з параметрами моделі.

Дані проблеми часто зустрічаються в ході обробки даних, тому для успішної реалізації класифікації необхідна ретельна настройка параметрів моделі та ретельний вибір та предобробка навчальних даних.

Однією з поширених проблем класифікації являється мультикласовий випадок, коли одному спостереженню належить декілька класів. Основні методи класифікації не можуть вирішити дану задачу, тому їх необхідно модифікувати для даного випадку.

Таким чином, на сьогоднішній день обробка даних з метою мультикласифікації є дуже важливим завданням. У даній роботі буде розглянута проблема обробки даних для мультикласифікації за допомогою нечіткої машини опорних векторів. Метод опорних векторів (SVM – Support Vector Machine, Машина Опорних Векторів) – метод, який використовується для класифікації простору ознак на два класи за допомогою побудови поділяючої гіперплощини. Щоб обійти проблему некласифікованих регіонів мультикласових даних буде запропонований метод нечіткої машини опорних векторів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Навчання з учителем

Для кожної моделі інтелектуального аналізу даних необхідний процес навчання моделі. Процес навчання пов'язаний з тим, що в задачах реального світу неможливо визначити всі можливі варіанти даних, що надходять на класифікатор, і відповідні їм правильні відповіді, які він повинен видавати.

Одним з класів алгоритмів навчання є алгоритми навчання з учителем. Навчання з учителем є найпоширенішим випадком навчання моделі[1]. В даному випадку на вхід моделі подається прецедент, який представляє з себе пару об'єктів – дані про об'єкт і його результат або відповідь. В ході розробки моделі типу навчання з учителем повинен бути створений алгоритм, який зможе відновлювати залежності всередині прецедентів і видавати відповідь з певною точністю.

Формальна постановка задачі може бути описана наступним чином:

Припустимо, що в нас є безліч об'єктів X, що представляють опис кожного з окремих об'єктів і їх відповідні результати $X = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{m}, y^{m})\}$, де m – кількість прикладів в навчальній вибірці і кожне спостереження є вектором $x^{(i)} = \{x_1, x_2, \ldots, x_n\}$, де n – кількість елементів опису одного спостереження та кожен із $y \in Y$ відповідає простору відповідей $Y=\{y^{(1)}, y^{(2)}, \ldots, y^{(k)}\}$, де k – це кількість можливих відповідей (класів).

Необхідно створити функцію $f: X \rightarrow Y$, яка буде наближати результат до реального, тим самим мінімізуючи функцію втрат, як на навчальній, так і на тестовій вибірках, а також на даних, які не беруть участі у навчанні. Функція втрат $L(y, \hat{y})$ величину відхилення відповіді, отриманого за класифікації, $y = f(x)$ від правильної відповіді $\hat{y}$ – яка була надана в навчальній тестовій вибірках. Для задач класифікації

функція втрат може виглядати наступним образом:

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

Для аналізу результатів роботи алгоритму вводиться функція якості отриманих відповідей, що представляє собою емпіричний ризик функції f на заданій вибірці X^m:

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

Для знаходження кращої моделі використовується метод мінімізації емпіричного ризику, що представляє собою пошук найкращої функції $f \in F$, при якій буде досягнуто мінімальне значення помилки на довільній навчальній вибірці X^m:

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

Однак, мінімальне значення функції якості на навчальній вибірці не є гарантією того, що побудована модель буде правильно відновлювати коректні результати цільової залежності між усім простором X. В даному випадку існує ймовірність перенавчання моделі[2]. Перенавчання характеризується тим, що алгоритм описує кожне спостереження без можливого шуму, що підганяє модель на дані з навчальної вибірки і призводить до неможливості адекватної роботи зі спостереженнями, які не збігаються зі спостереженнями з навчальної вибірки. Таким чином, мінімізація емпіричної помилки до значень близьких до нуля може позбавити алгоритм можливості до узагальнення. При навчанні на довільній вибірці X^m модель запам'ятовує тільки значен результати з даної

множини прецедентів, що в свою чергу веде до того, що спостереження, які не належать навчальній вибірці, будуть призводити до довільного результату. Для того, щоб уникнути дану проблему необхідно підбирати правильні параметри моделі і використовувати різні підходи мінімізації ризику перенавчання.

1.2 Класифікація

На даний момент існує велика кількість алгоритмів класифікації, що відрізняються один від одного своїми підходами. До них відносяться алгоритми класифікації різних методів, але в рамках даної роботи буде розглянута обробка даних за допомогою алгоритму машини опорних векторів, який відноситься до лінійно поділяючих алгоритмів.

Класифікація являє собою один з розділів машинного навчання та інтелектуального аналізу даних, в ході якого створюється модель f, здатна при надходженні на неї даних $x \in X$ віднести дане спостереження до одного з класів $y \in Y$.

За допомогою класифікації вирішується великий спектр різних завдань: медична діагностика (класифікація видів захворювання, оцінка ризиків, пошук синдромів); розпізнавання об'єктів (рукописні символи, різні об'єкти зовнішнього світу); розпізнавання мови; біологічна діагностика (парсинг ДНК, класифікація біологічних видів) і т.д. Всі ці завдання вимагають навчальної вибірки з відповідями для кожного з спостережень для успішного створення класифікатора.

Правильно створений класифікатор повинен знайти приховані зв'язки між даними і відобразити закономірності між спостереженнями, що належать до одного класу. Класифікація відбувається за типовим планом алгоритмів типу навчання з учителем, але на відміну від регресії, має функцію втрат, засновану на кількості помилок прогнозів, а не від квадрату різниці між передбаченою і правильною відповідями.

1.3 Основні методи класифікації

На даний момент існує велика кількість різних алгоритмів класифікації. В даному розділі будуть поверхнево розглянуті і описані одні з найпопулярніших підходів[3], що використовуються в світі машинного навчання та інтелектуального аналізу даних.

Один з найпопулярніших і інтуїтивно-зрозумілих алгоритмів – метод k-найближчих сусідів представлений на рисунку 1.1. В даному алгоритмі можна виділити такі кроки:

- обчислення відстаней від об'єкта, який класифікується, до всіх інших об'єктів навчальної вибірки;

- з множини всіх відстаней до об'єктів необхідно вибрати k елементів, які є найближчими до об'єкта класифікації;

- присвоїти об'єкту, що класифікується, мітку класу, якому відповідає більшість з k найближчих сусідів.

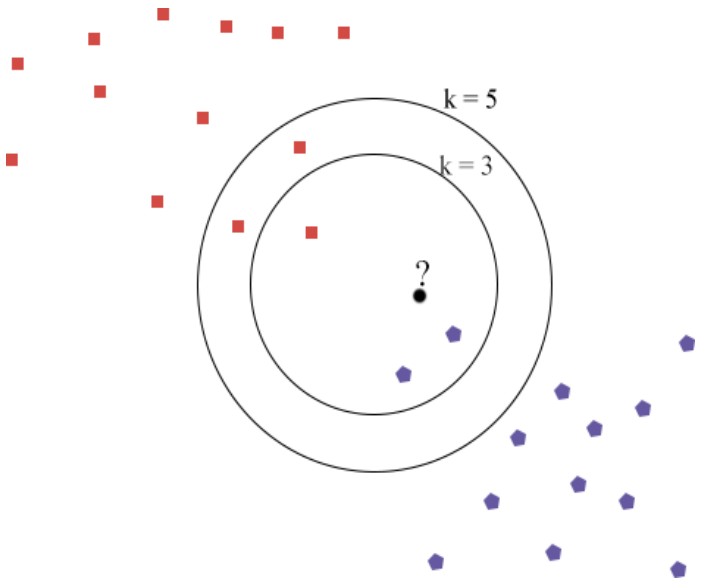


Рисунок 1.1 – Алгоритм kNN

Варто відзначити, що перед використанням даного методу необхідно провести нормалізацію даних, з метою прирівнювання важливості кожного з атрибутів. Класичний варіант використання даного методу базується на обчисленні дистанції в евклідовому просторі.

Наступним важливим напрямком в алгоритмах класифікації є дерево рішень. Дані методи базуються на обчисленні ступеня впливу кожного з атрибутів спостережень на вихідний атрибут. В ході даних обчислень будується дерево рішень, через яке буде проходити об'єкт класифікації. Дерево являє собою набір вузлів, які допомагають визначено описувати кожен з вхідних прикладів. Поділ вузлів описується розподілом вихідного атрибута (класу) об'єкта при дотриманні умови, описаного в даному вузлі.

Спостереження, яке поступає на вхід, повинно пройти через усі вузли, поки не досягне листової вершини. Кожна з листових вершин означає один клас з простору класів. Той клас, якому відповідає дана листова вершина, і буде присвоєний даному об'єкту класифікації. Клас спостереження можна представити у вигляді кон'юнкції набору правил, яким відповідає дане спостереження.

Наступним поширеним алгоритмом класифікації є наївний байесовский класифікатор. Даний метод є відносно простим, але в той же час іноді дуже ефективним. Дана ефективність викликана тим, що наївний байесовский класифікатор передбачає що кожен атрибут є незалежним один від одного.

Даний метод ґрунтується на теоремі Байеса. За допомогою даної теореми обчислюється ймовірність приналежності даного об'єкта до певного класу. Вибирається той клас, ймовірність настання якого є максимальною.

Наступний алгоритм, на базі якого буде розглядатися подальша робота по обробці даних, є алгоритм машини опорних векторів або метод опорних векторів (SVM – Support Vector Machine).

Даний метод є одним з найпопулярніших методів навчання

класифікаторів. Суть даного методу полягає в побудові поділяючої гіперплощини, яка повинна бути максимально і рівно віддалена від усіх опорних векторів обох класів.

У разі лінійної роздільності простору спостережень на два класи, проводиться гіперплощина між ними, що в реальному світі зустрічається вкрай рідко. Найчастіше простір спостережень неможливо лінійно розділити. Для подолання даної проблеми вводиться функція подібності (ядро), тим самим збільшуючи розмірність простору, в якому розглядається дана вибірка. Даний процес буде розглянуто в наступних розділах.

Алгоритм машини опорних векторів є актуальним і часто використовуваним методом класифікації в нинішній час, що викликає великий інтерес до даного алгоритму. Тому для завдання обробки даних був обраний даний алгоритм.

1.4 Постановка задачі

Метою даної роботи є аналіз методу опорних векторів, розбір роботи алгоритму, опис роботи алгоритму в разі лінійної роздільності і в разі неможливості лінійної роздільності простору спостережень і підходи подолання даної проблеми.

Наступним кроком буде опис засобів розробки для створення даного методу, засобів візуалізації та реалізації даного програмного продукту.

Також, буде розроблений класифікатор машини опорних векторів з метою обробки даних і перевірки залежності результатів класифікації моделі від параметрів моделі, і результати роботи будуть представлені у вигляді графічних візуалізацій і результатів класифікації. Так як машина опорних векторів належить до методів штучних нейронних мереж с прямою передачею інформації, необхідно провести аналіз методів з даною архітектурою.

2 ШТУЧНІ НЕЙРОННІ МЕРЕЖІ З ПРЯМОЮ ПЕРЕДАЧЕЮ СИГНАЛІВ

Штучні нейронні мережі – це обчислювальні мережі, що намагаються імітувати роботу мозку живої істоти за допомогою спрощення реальної моделі мозку. Біологічний нейрон мозку являє собою дуже складну систему, що обумовлюється необхідністю не тільки передавати та обробляти сигнал, а ще й підтримувати свою життєздатність.

Математичний нейрон являє собою набір входів, через які надходить вхідний сигнал, набір ваг, які відповідають кожному входу, суматор, який знаходить суму добутків сигналів та ваг, функцію активації та вихід нейрона[4]. На рисунку 2.1 наведена модель нейрону.

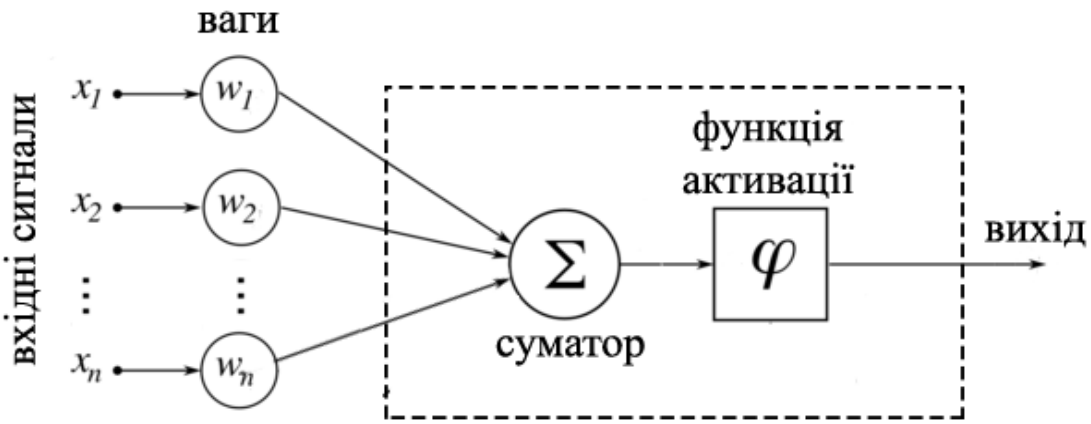


Рисунок 2.1 – Приклад схеми штучного нейрону

У кожного нейрону повинні бути входи X, через які він отримує вхідні сигнали. Після надходження сигналів до входів виконується добуток кожного сигналу з вагою w_i, якій він відповідає. Наступна дія – сума добутків у суматорі. Вона має вигляд:

$$\sum_{i=1}^{n} x_i w_i . \qquad (2.1)$$

Також суматор (2.1) можна представити у вигляді добутку векторів:

$$\sum_{i=1}^{n} x_i w_i = W^T X. \qquad (2.2)$$

Після знаходження суми добутків результат передається до активаційної функції φ. Виходом нейрону буде:

$$out = \varphi(W^T X). \qquad (2.2)$$

Функція активації – це функція, яка обчислює вихідний сигнал штучного нейрону[5]. Таким чином, нейрон описується завдяки його вагам та функцією активації. Вибір активаційних функцій залежить від задачі та підходу. Найпоширеніші функції активації наведені в таблиці 2.1.

Таблиця 2.1 – 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	$\frac{1}{1 - e^{-W^T X}}$	(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	$e^{-(W^T X)^2}$	(0, 1)

Гіперболічний тангенс	$\frac{e^{2(W^T X)} - 1}{e^{2(W^T X)} + 1}$	(0, 1)

Традиційні нейронні мережі з прямою передачею представляють собою систему нейронів, які взаємодіють між собою, кожен з яких виконує функціональне перетворення над сигналами, що поступають до них на вхід, у яких зв’язки йдуть строго з однієї сторони до іншої. Нейрона мережа являє собою набор шарів, які складаються з нейронів. Результат нейронної мережі – це результати функції активації на останньому вихідному шарі нейронної мережі. Мережі прямого поширення – це як правило багатошаровий перцептрон[4].

Ця мережа зазвичай складається з певної кількості входів, вузлів джерела, по іншому – сенсорних елементів, промацують зовнішнє середовище ШНМ. Ця множина входів утворює вхідний шар. Далі вони передають сигнали прихованим шарам, які виробляють обчислювальні операції і ніяк не пов'язані з навколишнім світом. Далі вони передають сигнал вихідному шару, який, в свою чергу, і надає інформацію вихідних нейронах. Як видно в мережах прямого поширення сигнал завжди рухається в одну сторону: від входу до виходу, проходячи всі шари по одному разу. Це зручно використовувати для більшості навчальних алгоритмів. Структура нейронної мережі представлена на рисунку 2.2.

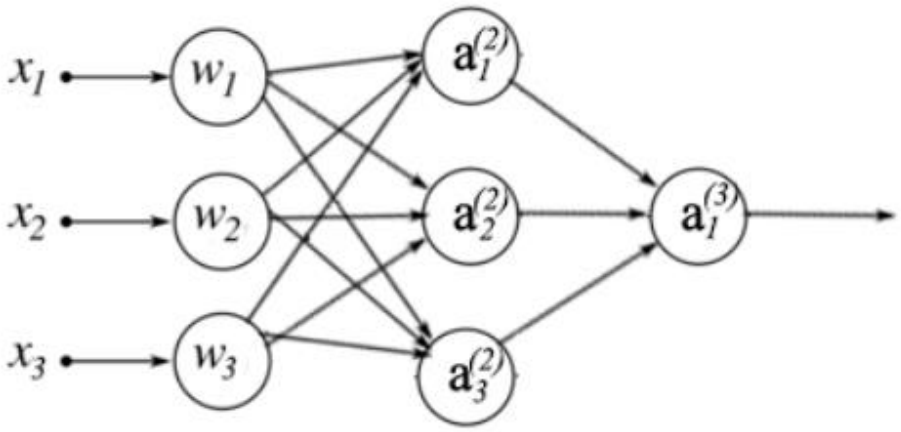


Рисунок 2.2 – Структура нейронної мережі прямого поширення

Результат активації нейрона і у шарі номер j ∈ L має собою вигляд $a_i^{(j)}$, де L – кількість шарів нейронної мережі. А матриця ваг між шаром j та j + 1 має вигляд $\theta^{(j)}$ і містить у собі ваги W для кожного із нейронів на цьому шарі. Таким чином:

$$a_i^{(j)} = \varphi(\sum_{k=0}^{m} \theta_{ik}^{(j-1)} x_k), \tag{2.3}$$

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

Вихід нейронної мережі буде являти собою активації нейронів на вихідному шарі L:

$$h_\theta(x) = a^{(L)}. \tag{2.4}$$

Складність нейронної мережі залежить від кількості внутрішніх шарів та кількості нейронів у кожному із шарів. Додавання всіх цих проміжних шарів в нейронні мережі дозволяє нам більш елегантно виготовляти цікаві та складні нелінійні гіпотези.

У даній роботі розглянуто навчання з учителем, при якому мережі послідовно представляється набір прикладів з навчальної множини. Приклади є собою пари еталонних вхідних впливів і бажаних вихідних сигналів. Процес навчання проходить циклічно, на кожній ітерації виконується розрахунок сигналів при прямому і зворотному поширенню, після чого сигнали помилок використовуються для формування локальних градієнтів векторів параметрів. Обчислені локальні градієнти використовуються для подальшого коректування параметрів. Використовувані режими навчання – це послідовний режим, при якому підстроювання параметрів відбувається після кожного прикладу, і пакетний, при якому підстроювання здійснюється на основі кумулятивного локального градієнта – суми локальних градієнтів по всім ітераціям прикладів з навчальної множини. В обох режимах повний цикл уявлення безлічі шаблонів навчання, що завершується підстроюванням параметрів, називається епохою навчання мережі.

З математичної точки зору процес навчання зводиться до мінімізації критерію якості навчання по вагам θ. В якості цільової функції найбільш часто використовується квадрат значення помилки навчання [6]:

$$E(\theta) = \frac{1}{2N}\sum_{i=1}^{N}\left(h_\theta\left(x^{(i)}\right) - y\left(x^{(i)}\right)\right)^2, \qquad (2.5)$$

де N – кількість представлених пар прецедентів у навчальній вибірці.

Методи навчання для підстроювання вектору параметрів адаптивного елемента використовують тільки ті сигнали, які присутні в самому розглянутому елементі. Коригуюча зміна $\Delta\theta_i$ для вектору параметрів θ_i обчислюється, на основі інформації, отриманої лише з його локального градієнта $\delta\theta_i$ і історії зміни цього локального градієнта з плином епох. Метод градієнтного спуску є найпростішим методом навчання мережі. Параметр коригується на величину відповідно до виразу

$$\theta_{i+1} = \theta_i + \theta_i. \quad (2.6)$$

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

$$\Delta\theta_i = -\varepsilon\delta\theta_i, \quad (2.7)$$

де ε – це коефіцієнт швидкості навчання.

У вираженні (2.7) знак мінус перед коефіцієнтом необхідний для зміни параметра як аргументу функції в сторону зменшення значення останньої. Важливо приділяти особливу увагу вибору коефіцієнта швидкості навчання: маленька величина коефіцієнта призведе до збільшення часу (кількості ітерацій), необхідного для навчання, однак занадто велика величина призведе до дестабілізації навчання через надмірну збільшення параметра так, що він буде «проскакувати» оптимальне значення. Звичайно, значення коефіцієнта приймають з умови:

$$0 < \varepsilon < 1. \quad (2.8)$$

Метод градієнтного спуску є базовим методом, на основі якого будуються інші методи[6].

3 МАШИНА ОПОРНИХ ВЕКТОРІВ

SVM або Машина Опорних векторів – це сімейство алгоритмів, що належать до методів навчання з учителем, які використовуються для регресійного і класифікаційного аналізу[7].

Метод опорних векторів вирішує проблему бінарної класифікації, розбиваючи простір спостережень за допомогою гіперплощини, яка повинна бути максимально віддалена від опорних векторів двох класів. Найпростіший варіант розвитку подій – класи лінійно роздільні і між ними можна провести площину. Приклад такої ситуації представлений на рисунку 3.1.

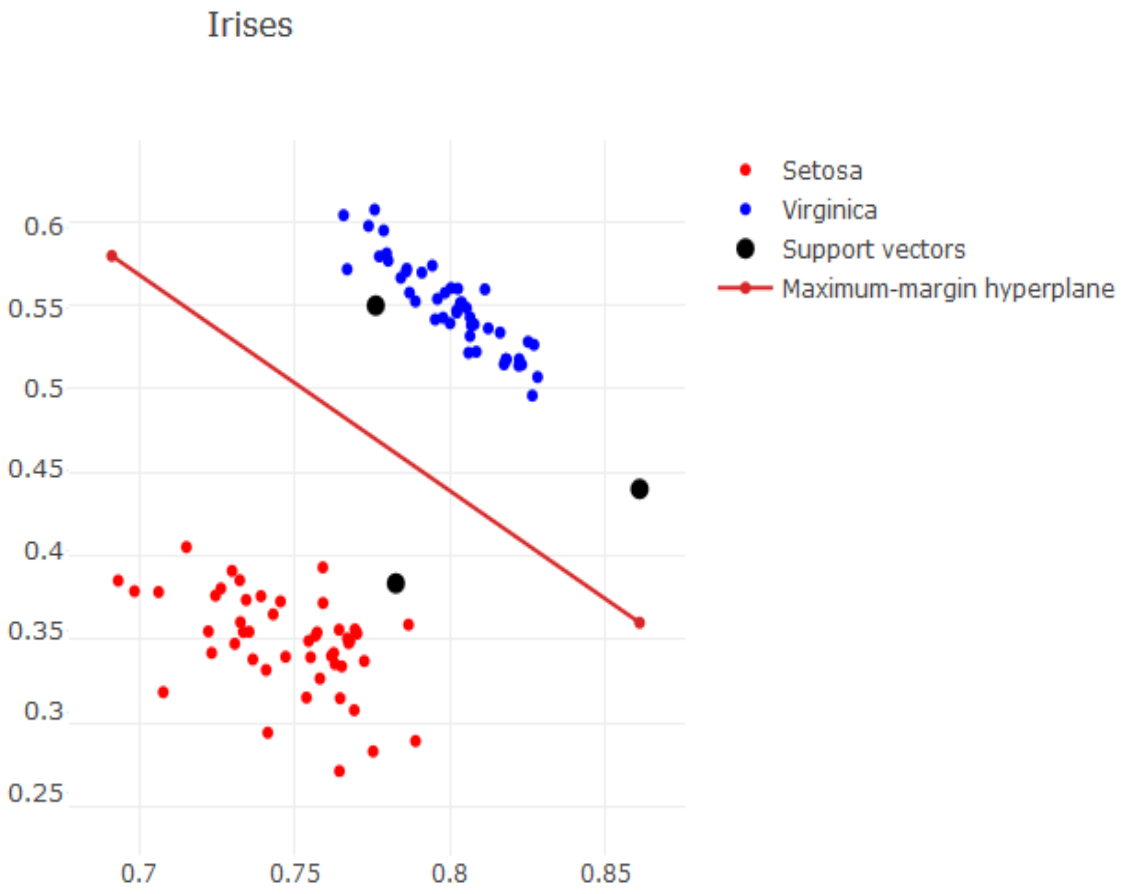


Рисунок 3.1 – Приклад роботи SVM на лінійно роздільній вибірці

На рисунку 3.1 великими точками позначені опорні вектору, за допомогою яких будується гіперплощина, що розділяє простір спостережень на два класи. Всі спостереження, що надходять на вхід, в залежності від розташування від поділяючої гіперплощини, будуть визначені до першого або другого класу.

У разі лінійної нероздільності необхідно перевести простір ознак в простір ознак більшої розмірності. В даному випадку замість скалярних добутків використовуються ядра.

Нейронні мережі опорних векторів, відомі також як машини опорних векторів (Support Vector Machines – SVM) є одним типом ШНМ, що використовують комбіноване навчання. Ці нейромережі відносяться до нейромережей із архітектурою з прямою передачею інформації, в якості активаційних функцій використовуються ядерні конструкції і вони являються узагальненням таких популярних конструкцій, як багатошарові персептрони, радіально базисні і поліноміальні мережі. Ці ШНМ реалізують метод мінімізації емпіричного ризику і знаходять застосування при вирішенні задач ідентифікації, розпізнавання образів, апроксимації, емуляції і т.п.

3.1 Лінійно розділима вибірка

Ключовим поняттям при синтезі цих мереж є опорні вектори (крайні вектори згідно первісної термінології В. Н. Вапника [2]), що представляють собою малу підмножини найбільш інформативних даних з навчальної вибірки і визначаються в процесі навчання. Розгляд методу мінімізації емпіричного ризику почнемо з найпростішої задачі розпізнавання образів, коли задана навчальна вибірка з відомою класифікацією даних $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$, де m – кількість даних, і є два лінійно розділимих класів. При цьому для одного з цих класів навчальний сигнал а для другого – y = -1. Рівняння поділяючої

гіперплощини при цьому має вигляд:

$$w^T x + b = 0, \tag{3.1}$$

де w – (n × 1)-вектор ваги, що підлягають визначенню у [3] узагальненим портретом;

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

$$\begin{cases} w^T x^{(i)} + b = 0 \geq 0, \text{ для } y = +1, \\ w^T x^{(i)} + b = 0 < 0, \text{ для } y = -1, \end{cases} \tag{3.2}$$

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

Для заданої навчальної вибірки, узагальненого портрета та зсувів b в розгляд вводиться область поділу p, визначена відстанню від поділяючої гіперплощини до найближчих до неї точок x^i з різних класів. Завдання мінімізації емпіричного ризику полягає в знаходженні гіперплощини з максимально можливою областю поділу. При цьому точки найближчі до цієї оптимальної гіперплощини і називаються опорними (крайніми) векторами.

Запишемо оптимальну гіперплощину у вигляді:

$$x^T w^* + b^* = 0, \tag{3.3}$$

$$D(x) = x^T w^* + b^*. \tag{3.4}$$

Визначаючим відстань від точки x до гіперплощини (3.3). При цьому сам вектор x може бути представлений у вигляді суми:

$$\mathbf{x} = \hat{\mathbf{x}} + \xi \frac{\mathbf{w}^*}{\|\mathbf{w}^*\|}, \qquad (3.5)$$

де $\hat{x}$ – проекція x на гіперплощину (3.3);

ξ – алгебраїчний параметр відстані, що приймає значення більше нуля на «невід'ємній» стороні гіперплощини и менше нуля на «від'ємній».

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

$$D(x) = x^T w^* + b^* = \xi \|w^*\|, \qquad (3.6)$$

звідки:

$$\xi = \frac{D(x)}{\|w^*\|} \qquad (3.7)$$

Аналіз (3.6) показує, що відстань від початку координат до гіперплощини (3.3) є $b\|w^*\|^{-1}$; якщо зміщення b^* невід'ємне, то початок координат знаходиться на невід'ємній стороні, якщо ж $b^* < 0$ на від'ємній, при $b^* = 0$ поділяюча гіперплощина проходить через початок координат.

Оптимальна поділяюча гіперплощина повинна задовольняти більш жорстким ніж (3.2) нерівностей [8], а саме:

$$\begin{cases} w^T x^{(i)} + b = 0 \geq 1, \text{ для } y = +1, \\ w^T x^{(i)} + b = 0 \leq -1, \text{ для } y = -1, \end{cases} \qquad (3.8)$$

При цьому пари (x, y), звертаючи будь-яке з нерівностей (3.8) в рівність, є опорними векторами і оскільки саме вони розташовані найближче до розділяє гіперплощини, їх найважче класифікувати. Ідея, що лежить в основі SVM, полягає, що визначення опорних векторів на заданій

навчальній вибірці дозволяє класифікувати всі інші дані. З (3.7) випливає, що для будь-якого опорного вектору $x^{(s)}$ відстань до оптимальної гіперплощини обчислюється таким чином [9]:

$$\xi = \frac{D(x^{(s)})}{\|w^*\|} = \begin{cases} \dfrac{1}{\dfrac{D(x^{(s)})}{\|w^*\|}}, \text{ для } y^{(s)} = +1, \\ -\dfrac{1}{\dfrac{D(x^{(s)})}{\|w^*\|}}, \text{ для } y^{(s)} = -1, \end{cases} \tag{3.9}$$

де знак «+» вказує, що $x^{(s)}$ лежить на невід'ємній, а «-» на від'ємній стороні оптимальної гіперплощини. З співвідношень (3.9) видно, що зона поділу визначається виразом:

$$p = 2\xi = \frac{2}{\|w^*\|}, \tag{3.10}$$

а її максимізація еквівалентна мінімізації евклідової норми вектору ваг $\|w\|$.

Таким чином, з формальної точки зору завдання мінімізації емпіричного ризику полягає в тому, що для заданої навчальної вибірки $X = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$, необхідно мінімізувати цільову функцію:

$$E(w) = \frac{1}{2}\|w\|^2, \tag{3.11}$$

при обмеженнях виду (3.7) або, що, те ж саме:

$$y^{(k)}\left(w^T x^{(k)} + b\right) \geq 1, \; k = 1, 2, \ldots, N. \tag{3.12}$$

Нескладно бачити, що вирази (3.10), (3.11) задають стандартне завдання квадратичного програмування [10].

В якості одного з найбільш простих і наочних методів аналізу цього завдання можна використовувати метод невизначених множників Лагранжа, пов'язаний з перебуванням особливих точок спеціально сконструйованої функції (лагранжиана), що має в даному випадку вид:

$$L(w, b, \lambda) = \frac{1}{2} w^T w - \sum_{k=1}^{N} \lambda_k \left(y^{(k)} \left(w^T x^{(k)} + b \right) - 1 \right), \qquad (3.13)$$

де λ_k – невід'ємні невизначені множники Лагранжа.

Знаходження оптимального рішення зводиться до відшукання сідлової точки лагранжиана (3.12), яка визначається системою Куна-Таккера [9]

$$\begin{cases} \dfrac{\delta L(w, b, \lambda)}{\delta w} = 0, \\ \dfrac{\delta L(w, b, \lambda)}{\delta b} = 0, \\ \dfrac{\delta L(w, b, \lambda)}{\delta \lambda_k} \le 0, \ \lambda_k \ge 0, \ k = 1, 2, \ldots, N. \end{cases} \qquad (3.14)$$

З першого рівняння (3.14) випливає, що шуканий вектор ваг визначається виразом:

$$w = \sum_{k=1}^{N} \lambda_k y^{(k)} x^{(k)}, \qquad (3.15)$$

а з другого рівняння видно, що

$$\sum_{k=1}^{N} \lambda_k y^{(k)} = 0. \tag{3.16}$$

Аналіз виразів (3.15), (3.16) показує, що для єдиної оптимальної гіперплощини може існувати безліч множників Лагранжа, визначити які в аналітичній формі неможливо. Можна лише відзначити, що в сідловій точці повинна виконуватися умова:

$$\lambda_k(y^{(k)}\left(w^T x^{(k)} + b\right) - 1) = 0,\ k = 1, 2, \ldots, N. \tag{3.17}$$

Обійти цю складність можна, розглянувши двоїсту задачу квадратичного програмування, рішення якої еквівалентно побудові узагальненого портрета [11]. Коректність такого прийому спирається на фундаментальне положення загальної теорії нелінійного програмування, з якого випливає те, що якщо вихідна задача оптимізації має рішення, то існує і дуальна до неї задача, що також має рішення, причому пряме і двоїсте рішення збігаються.

Для пошуку множників Лагранжа потрібно підставити формули (3.15) та (3.16) до лагранжиану:

$$\begin{aligned} L(w, b, \lambda) &= \frac{1}{2} w^T w - \sum_{k=1}^{N} \lambda_k y^{(k)} \left(w^T x^{(k)} + b\right) - 1) = \\ &= \frac{1}{2} w^T w - (w^T w - \sum_{k=1}^{N} \lambda_k) = \\ &= \sum_{k=1}^{N} \lambda_k - \frac{1}{2} \left\| \sum_{k=1}^{N} \lambda_k y^{(k)} x^{(k)} \right\|^2. \end{aligned} \tag{3.18}$$

Таким чином, задача зводиться до пошуку критичних точок функції

$$\Phi(\lambda)= \sum_{k=1}^{N} \lambda_k - \frac{1}{2} \left\| \sum_{k=1}^{N} \lambda_k y^{(k)} x^{(k)} \right\|^2 . \qquad (3.19)$$

Так як ця функція являє собою різницю лінійної і квадратичної функцій, причому квадратична функція від'ємно визначена, то потрібно знайти найбільше значення функції Ф(λ) при умові (3.16) та невід'ємності параметра λ. Далі проводиться оптимізація одним із алгоритмів оптимізації.

3.2 Лінійно нерозділима вибірка

У випадку лінійно нероздільної вибірки необхідно перевести простір спостережень у простір більшої розмірності.

Як зазначалося вище, машина опорних векторів є узагальненням цілого роду нейромереж з прямою передачею інформації.

В рамках даної задачі в якості таких ядер використовуються ті ж Гауссіани виду:

$$\varphi(x) = \exp\left(- \frac{\left\| x - x^{(s)} \right\|^2}{2\sigma^2} \right), \qquad (3.20)$$

де σ – параметр середньоквадратичного відхилення, який налаштовується вручну.

Також, у якості ядер можуть використовуватися поліноміальні ядра:

$$\varphi(x) = (x^T x^{(s)} + r)^d, \qquad (3.21)$$

де r – довільний параметр;

d – ступінь поліному.

Основною відмінністю мережі опорних векторів від традиційних є те, що розмірність простору ознак, а отже, і кількість нейронів в прихованому шарі може бути визначено заздалегідь і дорівнює числу опорних векторів в навчальній вибірці, тобто векторів, що задовольняють умові:

$$\begin{cases} \varphi(x^{(s)})^T w^* = 1, \text{ для } y = +1 \\ \varphi(x^{(s)})^T w^* = -1, \text{ для } y = -1 \end{cases}. \quad (3.22)$$

Підхід, пов'язаний з використанням опорних векторів, з обчислювальної точки зору громіздкий, процедура оптимізації містить досить багато вільних параметрів, які вибираються користувачем і можуть робити істотний вплив на якість одержуваного рішення, а сама SVM може працювати тільки в пакетному режимі на вибірці фіксованого обсягу.

«Слабким місцем» SVM є громіздкість процедури навчання синаптичних ваг, яка зводиться до вирішення задачі нелінійного програмування з обмеженнями-нерівностями, число яких визначається обсягом навчальної вибірки[4].

3.3 Мультикласова класифікація

Найчастіше, задачі класифікації розглядаються як однокласові проблеми, де кожне спостереження належить тільки одному класу. Але у задачах класифікації даних нерідко виникає необхідність робити мультикласові передбачення, де спостереження може відноситися більш ніж до одного класу.

Машина опорних векторі відноситься до того типу алгоритмів, які можуть вирішувати проблему бінарної класифікації. Тому, алгоритм SVM необхідно адаптувати до мультикласових випадків. Один з поширених методів реалізації мультикласових передбачень – це перетворення

мультикласової задачі до однокласової методом створення нових класів, які є згрупованими варіантами різних класів, але при такому підході кількість класів може бути дуже великою. Але, попре це ми можемо використовувати підхід один проти всіх, де моделі створюються по кількості цільових класів.

У випадку використання машини опорних векторів у нас може виникнути проблема, коли гіперплощини розділять простір на площини, в яких нові спостереження не будуть відноситися до жодного із класів. Двовимірний приклад зображений на рисунку 4.1.

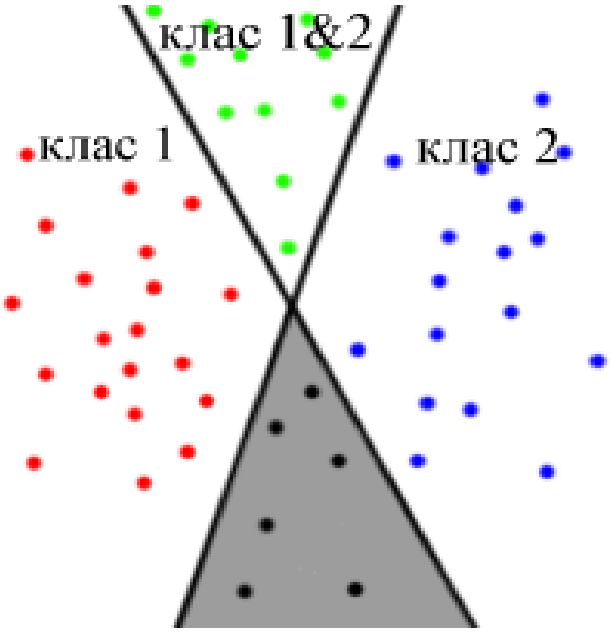


Рисунок 4.1 – Мультикласова класифікація за допомогою SVM

Як можна побачити на рисунку 4.1 спостереження, які знаходяться нижче обох гіперплощин, користуючись формулою 3.22, будуть аутлаєрами та не будуть відноситися до жодного з класів. Таким чином, в задачах мультклассової класифікації за допомогою машини опорних векторів виникає проблема некласифікованих регіонів.

Тому, необхідно розробити механізм, який буде вирішувати проблему некласифікованих регіонів.

3.4 Нечітка машина опорних векторів

Для того, щоб класифікувати спостереження, які не належать до жодної області класифікації, необхідно створити нові класи, які є комбінацією різних класів. Таким чином, n – кількість класів у нашій вибірці, k – комбінації одиночних класів у вибірці. $L = \{1, \dots n, k_1, \dots, k_e\}$, де L – усі можливі варіанти класифікацій, e – кількість можливих мультикласових площин, $L_n = \{1, \dots, n\}$. Перепишемо формули 3.3 та 3.4 для оптимальної гіперплощини у мультикласовому випадку:

$$D_i(x) = 0, \tag{3.23}$$

де i – номер класу в інтервалі від 1 до n.

У цьому випадку створюється $n + e$ регіони, до яких відносяться спостереження:

$$R_k = \{x | D_i(x) > 0 \; for \; i \in L_{ke}, \; D_i < 0 \; for \; i \; L_n - L_{ke}\} \; for \; k = 1, \dots, l, \tag{3.24}$$

де, регіон для класу k містить в собі спостереження, які відносяться до даного класу k. Таким чином, якщо спостереження x належить до регіону R_k, то воно класифікується як k, але при цьому залишаються регіони, які не відносяться до жодного з класів.

Введемо функцію приналежності до кожного з класів. У випадку, коли наше спостереження відноситься до k-го класу, функція приналежності виглядає так:

$$m_{ki}(x) = \min\big(1, \; D_i(x)\big), \; for \; i \in L_k. \tag{3.25}$$

Для гіперплощин, до яких спостереження не належить, функція приналежності виглядає таким чином:

$$m_{ki}(x) = \min\big(1, -D_i(x)\big),\ for\ i \in L_n - L_k. \tag{3.26}$$

Таким чином, належність спостереження x до заданого регіону R_k виглядає так:

$$m_k(x) = \min_{i=1..e} m_{ki}(x)\,. \tag{3.27}$$

Ця формула вказує на те, що функція належності до вибраного регіону вимірюється за допомогою найближчої гіперплощини до заданого спостереження. Якщо значення функцій належності до кожного з регіону являються від'ємними числами, це значить, що спостереження знаходиться в тому регіоні, який не належить до жодного із класів. В такому випадку спостереження відноситься до найближчого класу:

$$arg\max_{i=1..e} m_k(x). \tag{3.28}$$

Виходячи з цього, можна побачити як нечітка машина опорних векторів вирішує проблему мультикласової класифікації, при відсутності поділяючого регіону для нових спостережень.

4 ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ

У цьому розділі буде представлені експериментальні дослідження та результати виконаної роботи. Головною задачею роботи являється продемонструвати можливості машини опорних векторів у обробці та класифікації даних, порівняти результати машини опорних векторів з нечіткою машиною опорних векторів та показати вплив параметрів моделі на результат.

4.1 Використовувані засоби розробки

Для реалізації даної роботи було використано мову програмування Python 3.6. Вибір даної мови програмування обумовлений тим, що на даний момент часу дана мова являється найбільш популярною та затребуваною у світі машинного навчання та інтелектуального аналізу даних. Велика кількість бібліотек, пакетів та фрайморків для машинного навчання реалізовані на Python[12].

Одною із головних переваг даної мови являється те, що Python став загальноприйнятою мовою для багатьох сфер застосування у інтелектуальному аналізі даних. Ця мова поєднує в собі міць мов програмування з простим використанням скриптових мов типу MATLAB та R[13].

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

інструментів типу Jupyter Notebook, який ми розглянемо нижче.

Машинне навчання та аналіз даних – це в основному ітераційні процеси, в яких дані задають хід аналізу. Вкрай важливо для цих процесів мати інструменти, які дозволяють оперативно і легко працювати.

В якості мови програмування загального призначення Python дозволяє створювати складні графічні інтерфейси (GUI) і веб-сервіси, а також легко інтегруватися в уже існуючі системи.

Для середи розробки було використано Jupyter Notebook. Jupyter Notebook є інтерактивною середою для запуску програмного коду в браузері. Це відмінний інструмент для розвідувального аналізу даних і широко використовується фахівцями з аналізу даних[14]. Незважаючи на те що Jupyter Notebook підтримує безліч мов програмування, нам потрібна лише підтримка Python. Jupyter Notebook дозволяє легко інтегрувати програмний код. На рисунку 4.1 представлена сторінка з Jupiter Notebook у браузері.

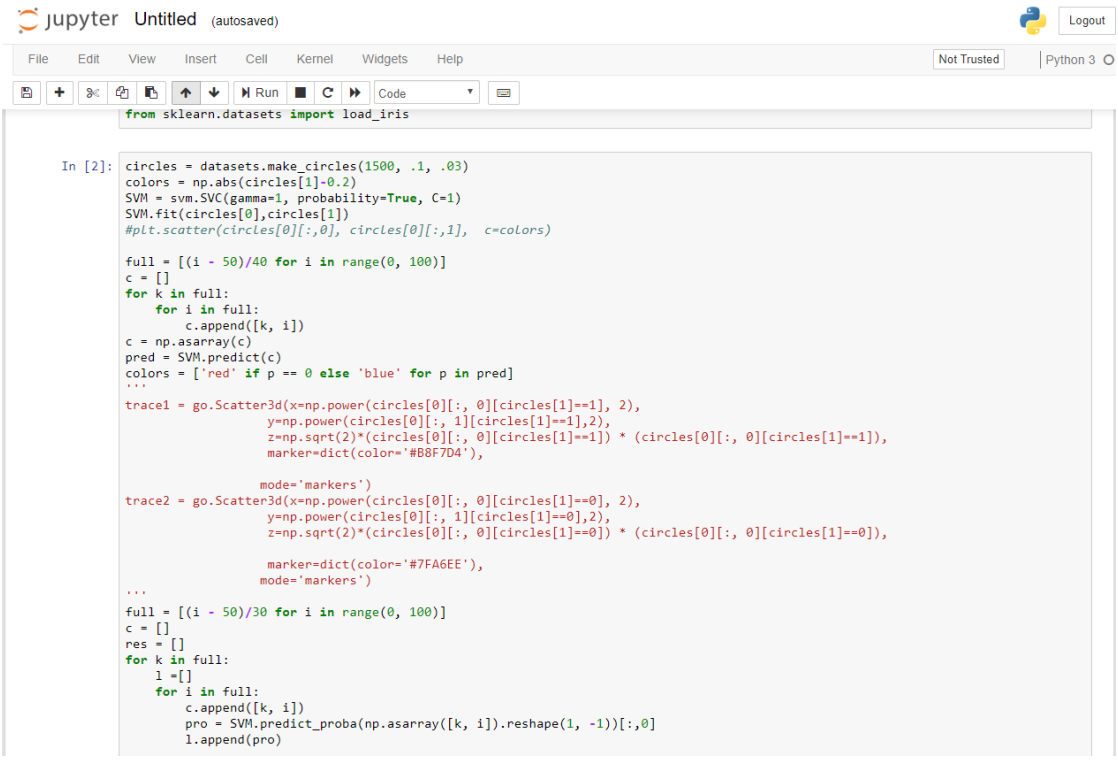


Рисунок 4.1 – Jupiter Notebook

Scikit-learn – проект з відкритим вихідним кодом, це означає, що його можна вільно використовувати і поширювати, і будь-яка людина може легко отримати вихідний код, щоб побачити, що відбувається «за лаштунками»[15]. Проект Scikit-learn постійно розвивається і вдосконалюється, і у нього дуже активна спільнота користувачів. Він містить ряд сучасних алгоритмів машинного навчання, а також повну документацію по кожному алгоритму. Scikit-learn – дуже популярний інструмент і найвідоміша пітоновская бібліотека для машинного навчання. Вона широко використовується в промисловості і науці, а в інтернеті є багатий вибір навчальних матеріалів і прикладів програмного коду. Scikit-learn прекрасно працює з низкою інших наукових інструментів Python.

NumPy – це один з основних пакетів для наукових обчислень в Python. Він містить функціональні можливості для роботи з багатовимірними масивами, високорівневими математичними функціями (операції лінійної алгебри, генератор псевдовипадкових чисел)[16].

У scikit-learn масив NumPy – це основна структура даних. scikit-learn приймає дані у вигляді масивів NumPy. Будь-які дані, які ви використовуєте, повинні бути перетворені в масив NumPy. Базовий функціонал NumPy – це клас ndarray, багатовимірний (n-вимірний) масив. Всі елементи масиву повинні бути одного і того ж типу. Масив NumPy представлено на рисунку 4.2.

```
import numpy as np
array = np.array([[2, 4 , 9], [3, 6, 1]])
print(array)
```

```
[[2 4 9]
 [3 6 1]]
```

Рисунок 4.2 – Масив NumPy

Також, треба згадати о дистрибутиві Anaconda. Дистрибутив Python, призначений для великомасштабної обробки даних, прогнозної аналітики і наукових обчислень. Anaconda вже включає NumPy, Jupyter Notebook і scikit-learn. Є версії для Mac OS, Windows і Linux[17]. Це дуже зручне рішення і це той дистрибутив, який рекомендується користувачам, у яких ще не встановлені пакети Python для наукових обчислень.

Для засобу візуалізації була використана бібліотека Plotly. У python є багато бібліотек для візуалізації: matplotlib, seaborn, портований з R ggplot і інші. Є серед них і ті, які дозволяють будувати інтерактивні графіки, наприклад, bokeh, pygal і plotly, про який власне і піде мова.

Plotly позиціонується як online-платформа, де можна створювати та публікувати свої графіки. Однак, цю бібліотеку можна використовувати і просто в Jupyter Notebook[18]. До того ж у бібліотеки є offline-mode, який дозволяє використовувати її без реєстрації і публікації даних її графіків на сервер Plotly. Є докладна документація з прикладами, підтримані різні типи графіків (scatter plots, box plots, 3D графіки, bar charts, heatmaps, дендрограмми і т.д.) і графіки виходять досить привабливими.

4.2 Експериментальні дослідження

В ході експериментальних досліджень буде розглянуто декілька вибірок даних на яких буде навчена модель SVM. Також у цьому підрозділі буде розглянуто вплив вибору параметрів моделі на результати обробки та класифікації даних.

За допомогою бібліотеки Plotly буде продемонстровано як класифікуються данні за допомогою моделі машини опорних векторів.

Перший приклад буде розглянуто на прикладі вибірки «Double donut», який являє собою дві розряджені окружності одна всередині іншої. Для того, щоб завантажити у пам'ять вибірку необхідно скористуватися функцією «make_circles» з пакету «datasets». В дану функцію ми

передаємо кількість спостережень, рівень шуму та фактор їх масштабу між внутрішнім та зовнішнім колами.

На рисунку 4.3 представлений графік, який відображає як виглядає вибірка і до якого класу відноситься кожна із окружностей.

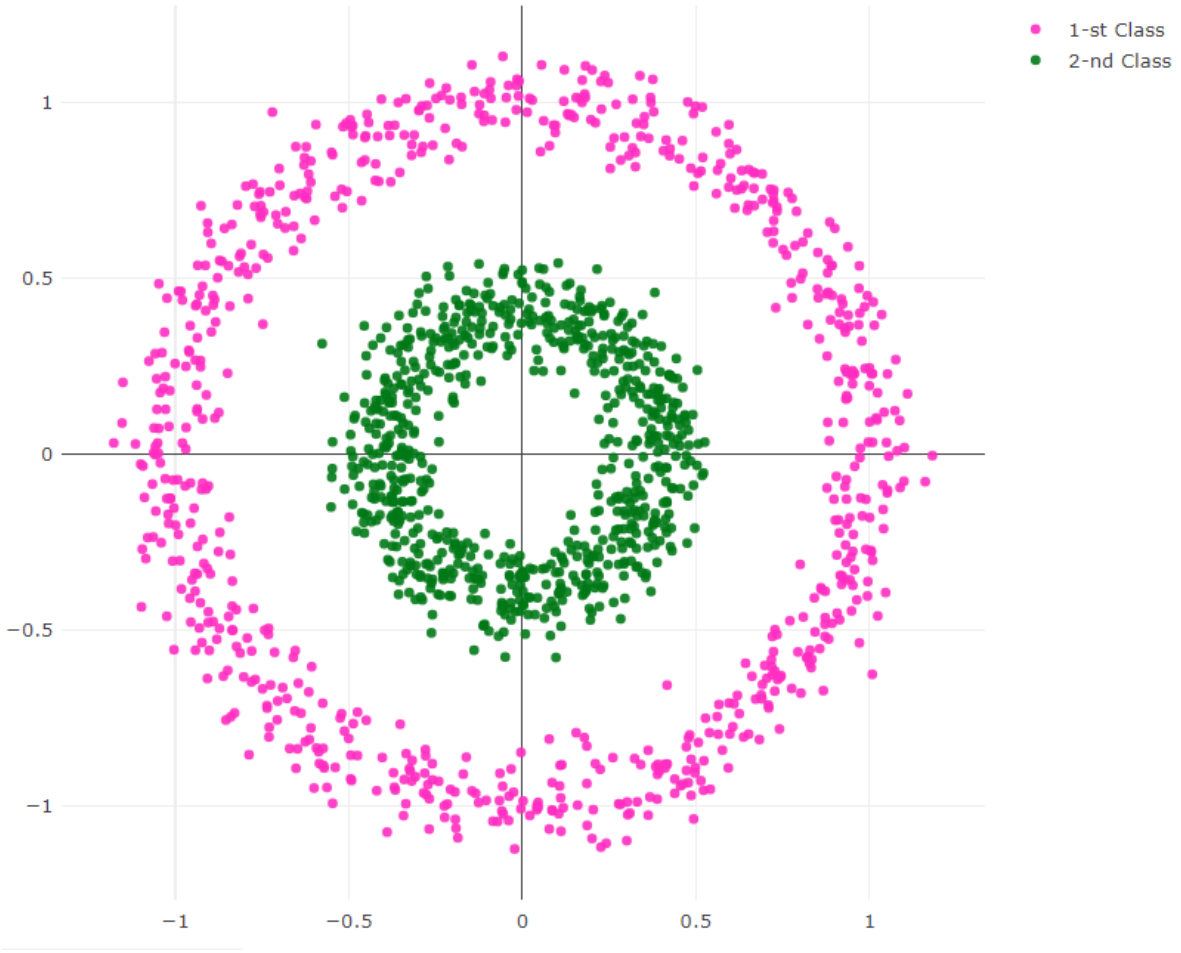


Рисунок 4.3 – Вибірка двох окружностей

Зараз треба обучити модель машини опорних векторів на цих даних та розглянути вплив вибору параметрів на роботу моделі. У моделі машини опорних векторів є два параметри, які впливають на навчання. Також, нам необхідно вибрати ядерну функцію за допомогою якої буде проходити навчання.

Ми можемо вибрати поліноміальне, радіально базисне або лінійне

ядро. Лінійним ядром називається відсутність ядра і використовування даних без обробки.

В даному випадку можна взяти поліноміальне або радіально базисне ядро. Але сперш, треба провірити як буде вести себе лінійне ядро на даній вибірці. На рисунку 4.4 представлені результати простора на які поділяє вибірку лінійне ядро.

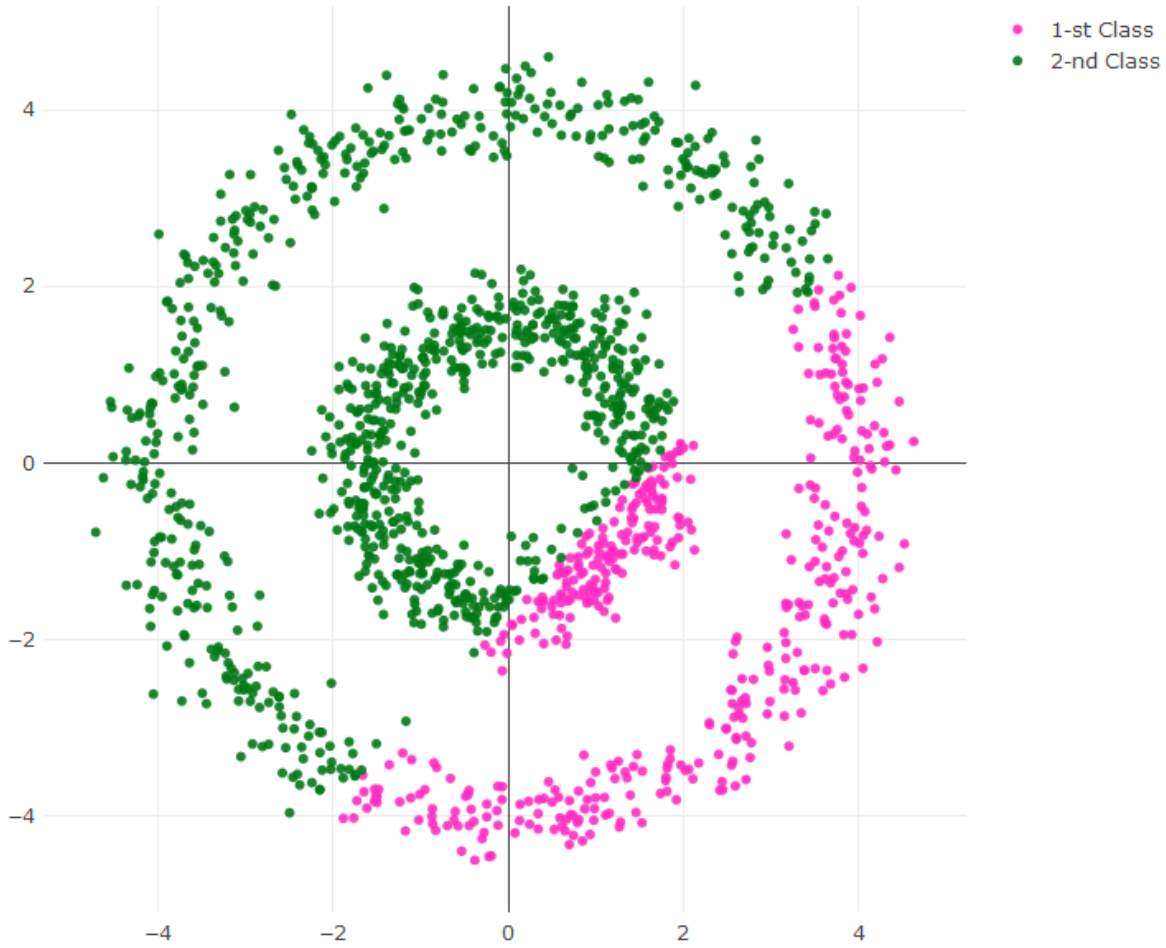


Рисунок 4.4 – Результат класифікації за допомогою SVM моделі з лінійним ядром

Зараз розглянемо класифікацію цієї вибірки за допомогою моделі машини опорних векторів, яка використовує якості ядра радіально базисне

ядро (3.20). В алгоритмі оптимізації за допомогою градієнтного спуску, який оптимізує функцію (3.19), використовується параметр штрафу C за невірну відповідь. Необхідно вручну вибирати параметр штрафу.

Таким чином, для адекватної класифікації необхідно вибрати параметр таким чином, щоб класифікатор буд не дуже загальним, але і не перенавченим.

При модифікації вихідного шару машини опорних векторів ми можемо отримати вірогідність настання кожного з класів. Завдяки ймовірностям приналежності до класу ми можемо побудувати 3D-графік, який буде показувати вірогідність настання заданої точки до одного з класів.

Розглянемо вплив параметру σ з формули (3.20) на результати класифікації. Нам необхідно вибрати параметр:

$$\gamma = \frac{1}{2\sigma^2}. \tag{4.2}$$

Чим більше даний параметр, тим більше модель перенавчається на становиться мнеш загальною. Перший параметр γ нехай прийме значення 0.15. Результати класифікації представлені на рисунку 4.5 та рис. 4.6.

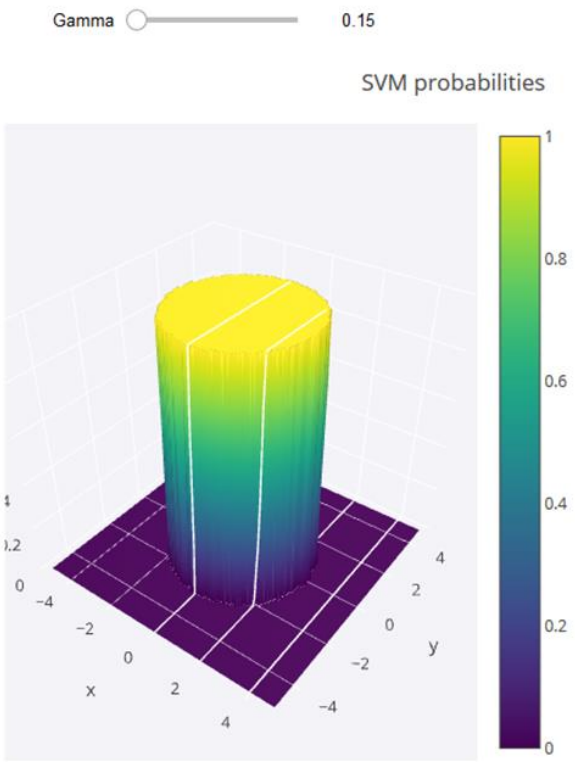


Рисунок 4.5 – Поверхня вірогідності настання 2-го класу

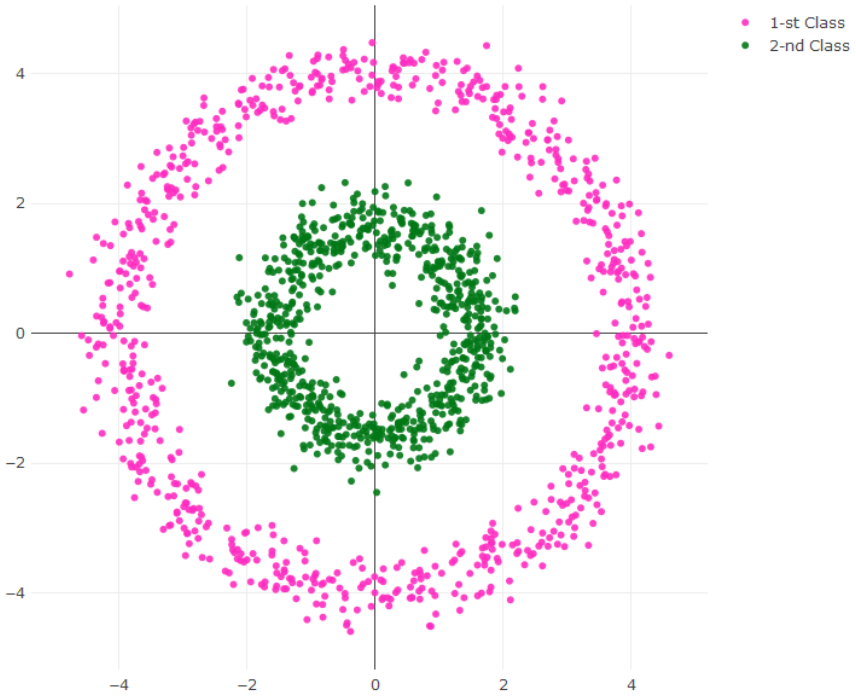


Рисунок 4.6 – Результат класифікації даної моделі

Таким чином, ми бачимо, що вибір радіально базисної функції було вибрано вірно. Та, дивлячись на рисунку 4.6 можна сказати, що поділяюча гіперплощина побудована вірно. Але, також треба зауважити, що всі спостереження, що знаходяться ззовні зовнішнього кола будуть належати до першого класу, а інші – до другого.

Також, необхідно визначити, що буде при збільшенні параметру γ. Подивимося на результуючу поверхню площини ймовірностей при γ = 10. На рисунку 4.7 показано результат роботи цієї моделі.

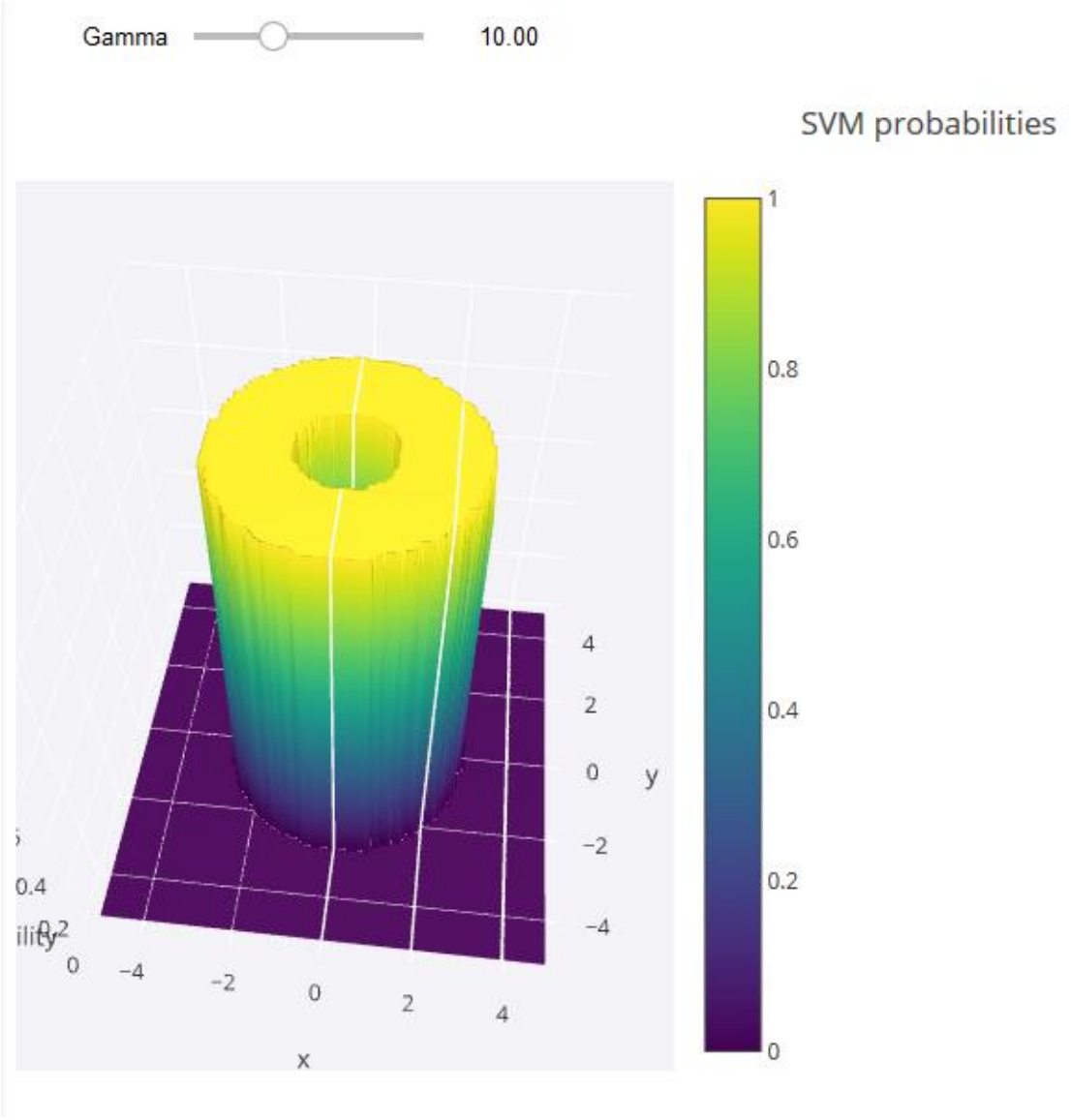


Рисунок 4.7 – Поверхня ймовірності настання 2-го класу при змінених параметрах

На рисунку 4.7 ми бачимо, що внутрішня частина внутрішнього кола буде відноситися до першого класу, у відміні від попередньої моделі. Цей факт говорить нам о том, що при збільшенні значення параметру радіально базисної функції наша модель становиться менш узагальненою та більш заточеною під один з класів.

Наступним шагом буде вибір параметру штрафу C, який повинен штрафувати модель при навчанні за невірну відповідь. На прикладі першої моделі, в якої параметр радіально базисної функції дорівнює 0.15, ми розглянемо вплив параметру На рисунку 4.8 та рисунку 4.9 представлені результати площини результатів при великому на малому значеннях параметру C.

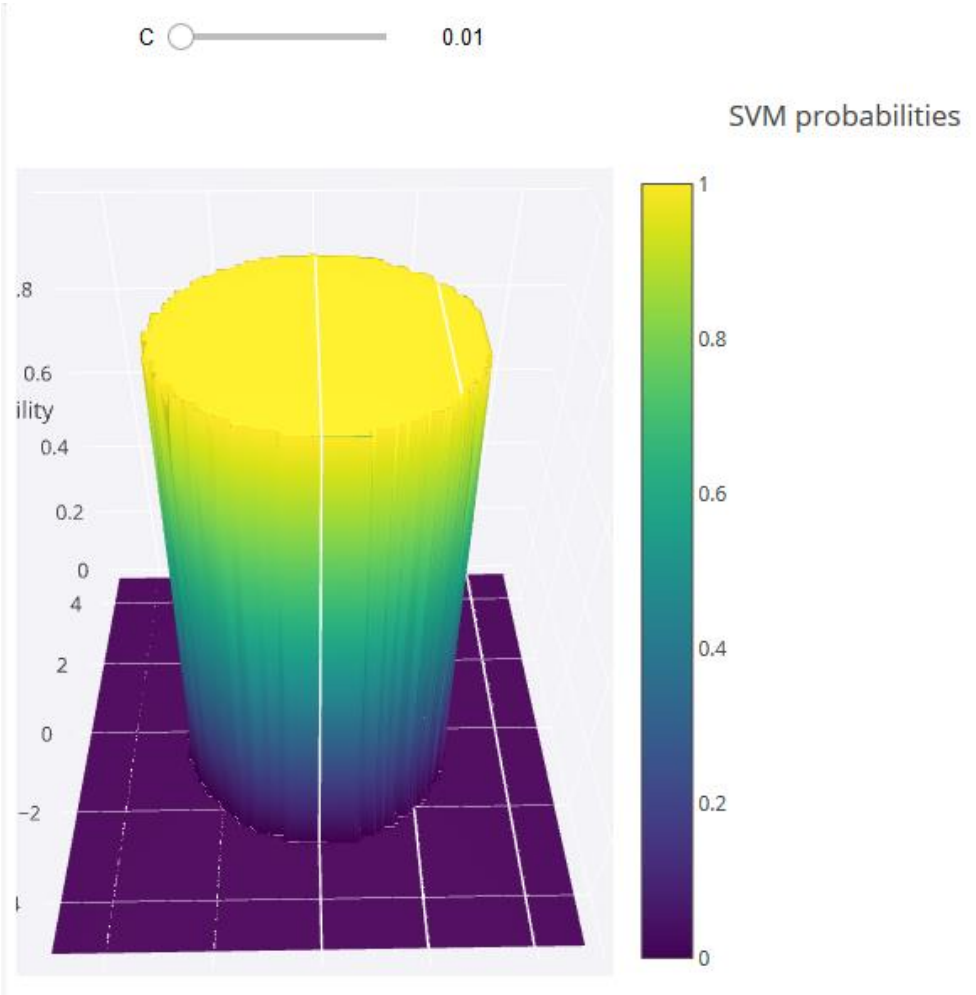


Рисунок 4.8 – Результати при малому значенню параметру C

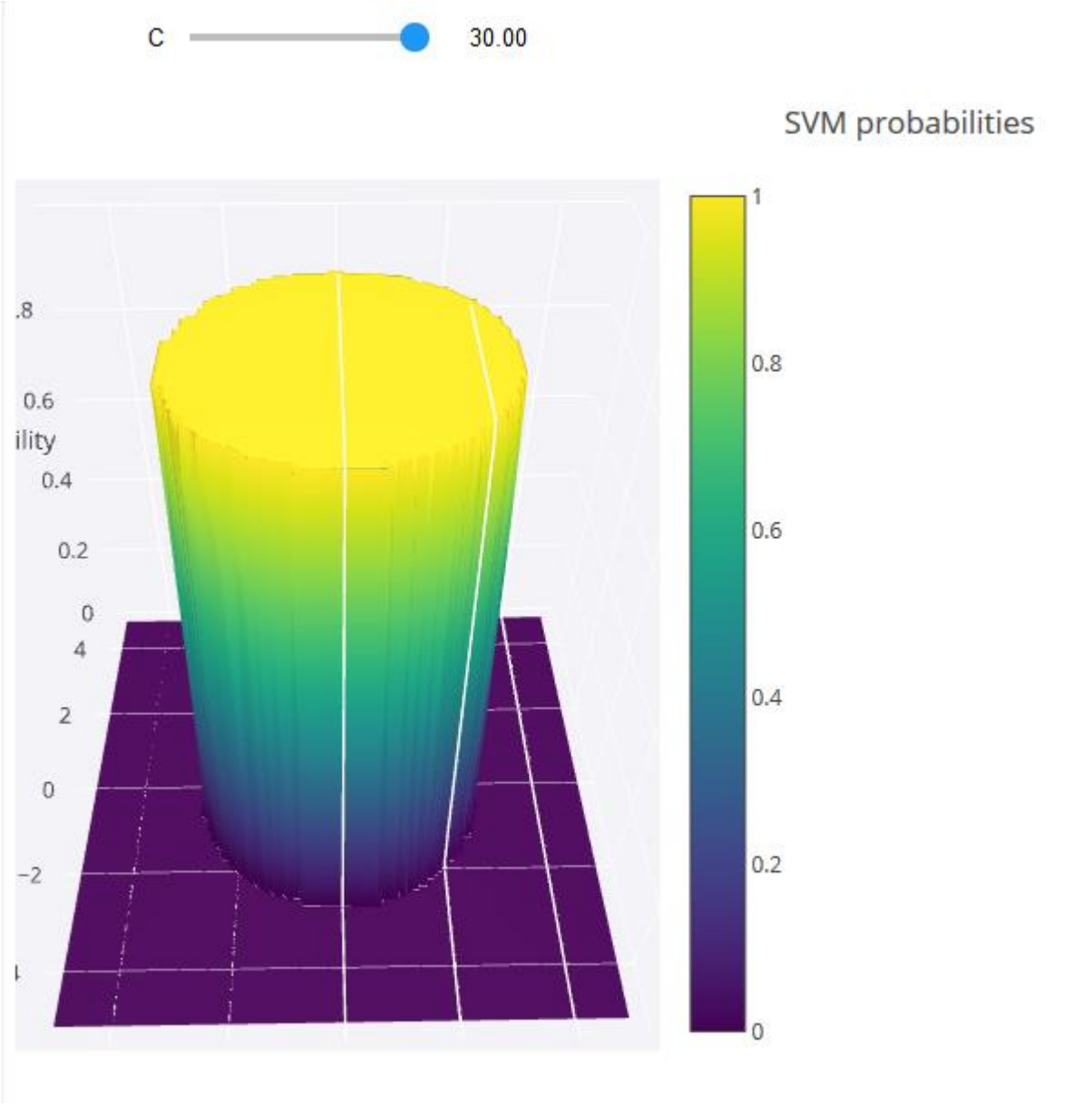


Рисунок 4.9 – Результати при великому значенню параметру C

Виходячи з отриманих результатів на вибірці з двома окружностями, можна зробити висновок, що для цієї вибірки параметр штрафу не має великого впливу. Це може бути пов’язано з тим, що при даних параметрах знаходження поділяюча гіперплощина знаходиться швидко та похибка під час навчання є дуже малим значенням.

Наступний приклад, який необхідно роздивитися – це вибірка «Подвійна спіраль». Цю вибірку можна скачати з відкритих джерел, або зробити власноруч за допомогою функції. Дана вибірка представляє собою

дві спіралі, які знаходяться одна в іншій. На рисунку 4.10 можна побачити як виглядає дана вибірка.

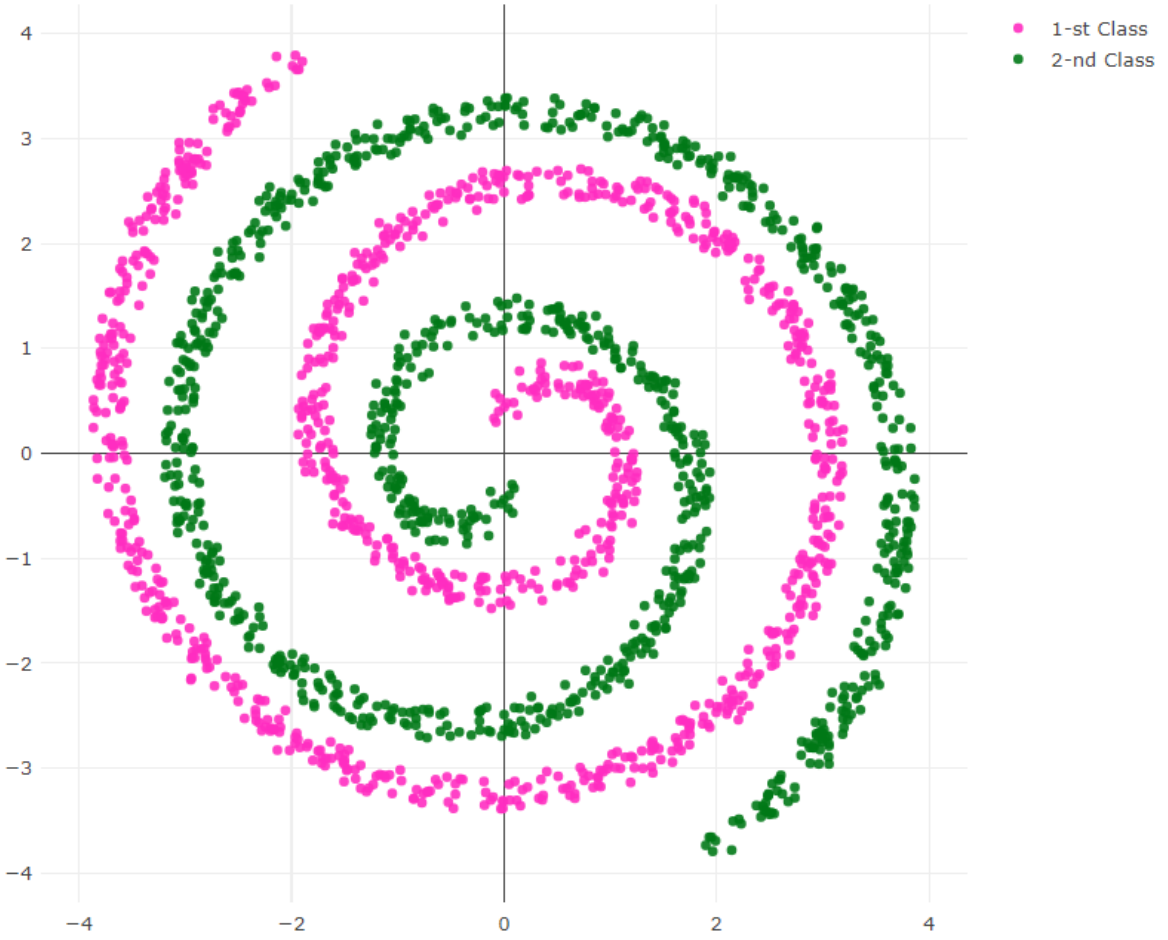


Рисунок 4.10 – Вибірка «Дві спіралі»

Тепер нам необхідно побудувати модель SVM, яка буде класифікувати дані спостереження. Ми можемо сказати, що даний простір спостережень неможливо лінійно розділити. Але, треба перевірити що на виході дає модель машини опорних векторів з лінійним ядром. На рисунку 4.11 представлена класифікація даної вибірки за допомогою моделі машини опорних векторів з лінійним ядром.

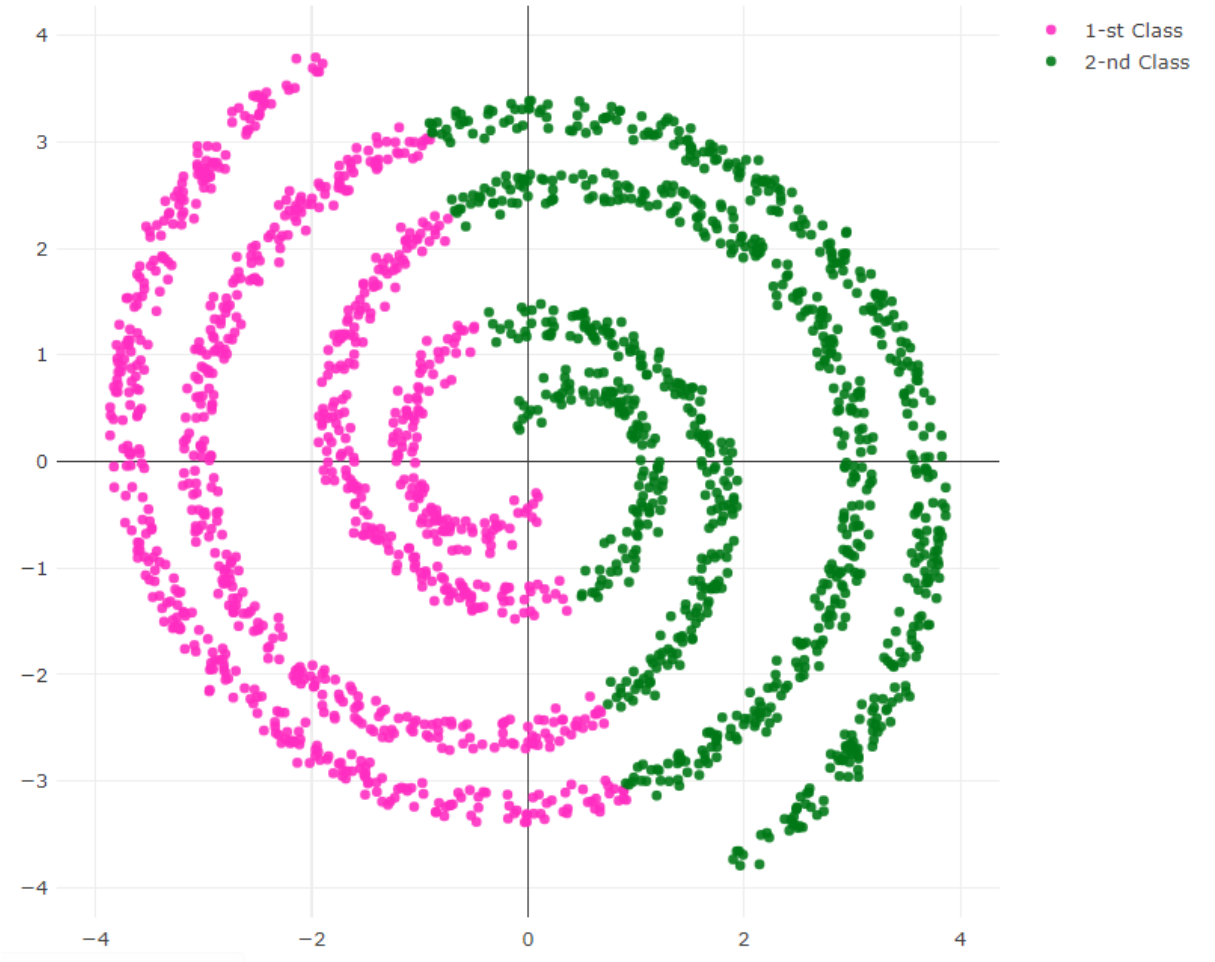


Рисунок 4.11 – Результати класифікації вибірки «Дві спіралі» з лінійним ядром класифікатора

Як ми можемо бачити на рисунку вище, лінійний класифікатор розбиває простір спостережень на дві половини, мінімізуючи, наскільки можливо, похибку класифікації. Так як лінійний класифікатор не дає потрібного результату, необхідно використати радіально базисне ядро.

Спробуємо навчити класифікатор з радіально базисним ядром на даній вибірці. Для того, щоб показати результати навчання класифікатора побудуємо площину ймовірностей настання одного з класів. На рисунку 4.12 представлена площина ймовірностей належності до першого класу при параметрах штрафу та параметру радіально базисного ядра дорівнюючим одиниці.

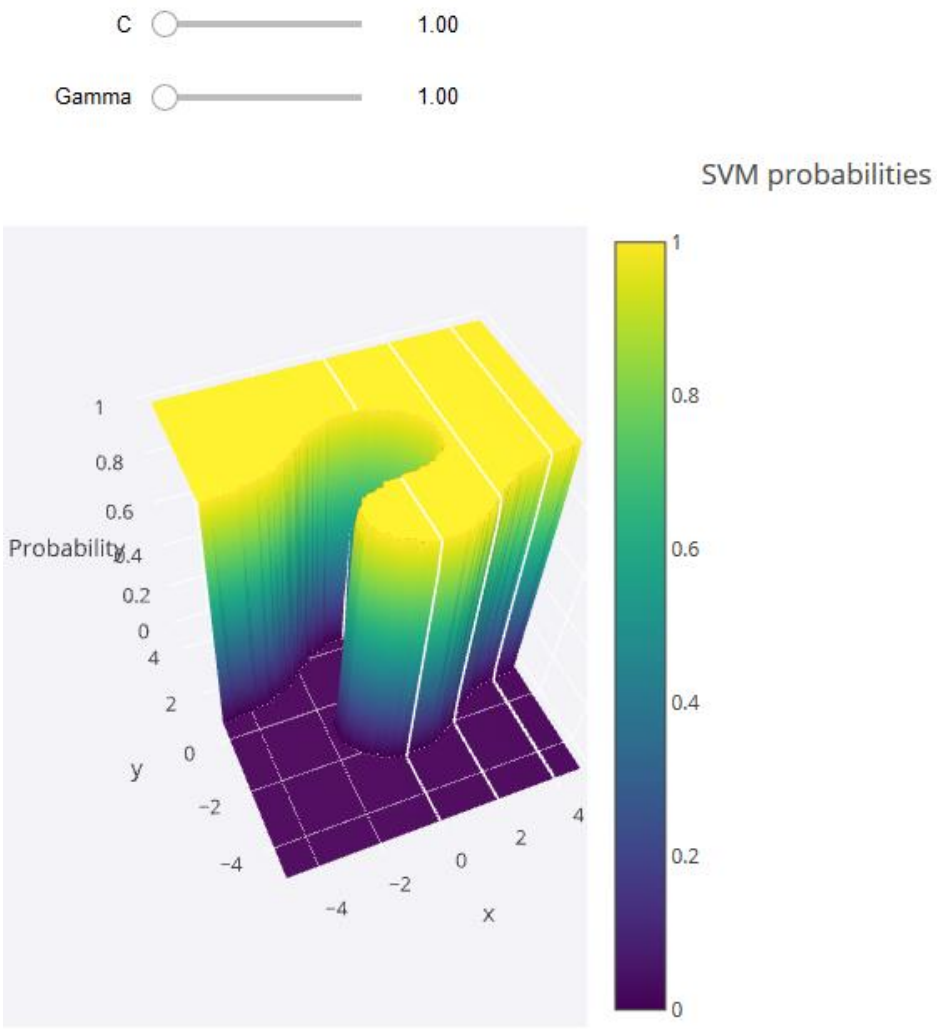


Рисунок 4.12 – Площина ймовірностей настання першого класу на вибірці «Дві спіралі»

Як ми бачимо, площина погано описує дану вибірку, так як параметри моделі вибрані недостатньо вірно для доброї класифікації. Це пов’язано з тим, що малі параметри штрафу та радіально базисної функції для складних моделей створюють недостатньо навчену модель, а узагальнюють її. Однак, при збільшені параметрів, модель стає біль перенавченою.

Результати самої класифікації даної вибірки представлені на рисунку 4.13.

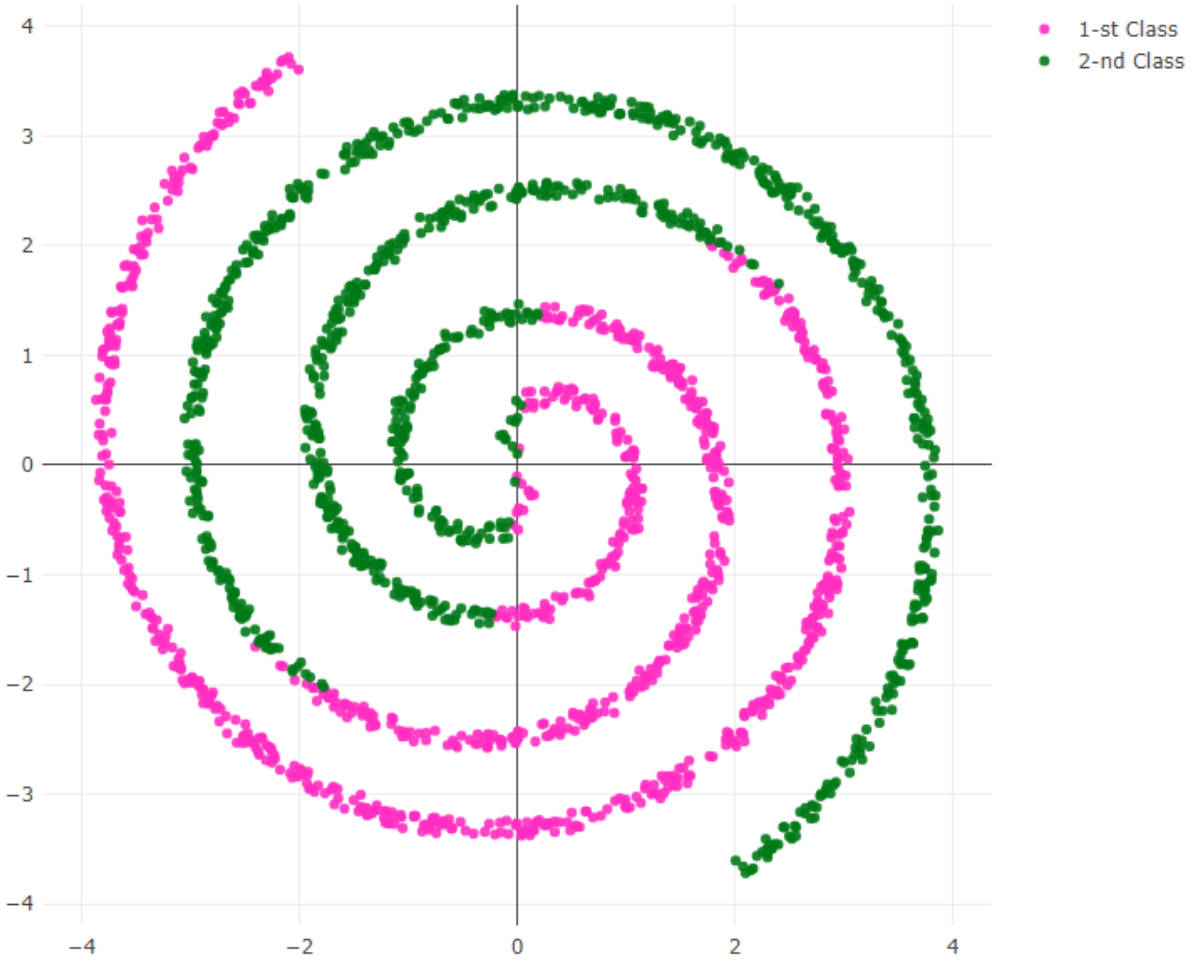


Рисунок 4.13 – Результат класифікації моделі з невеликими значеннями параметрів

Як можна побачити на попередньому рисунку, класифікація класів виконана недостатньо гарно для того, щоб вірно класифікувати навіть ті спостереження, що використовувались у навчанні моделі. Це пояснюється тим, що модель є дуже загальною та не може створити вірну поділяючу гіперплощину, так як параметри моделі не дозволяють створити більш складну модель.

Для вирішення цієї проблеми необхідно збільшити параметр штрафів або збільшити параметр радіально базисної функції. На рисунках 4.14 та 4.15 приведені графіки площин вірогідності настання першого класу при збільшені параметру штрафів та радіально базисної функції відповідно.

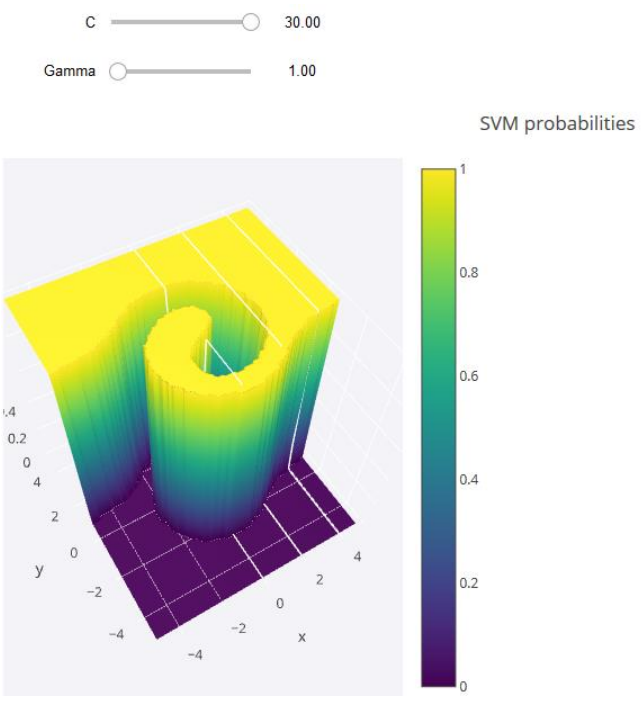


Рисунок 4.14 – Результати при збільшенні параметру штрафів

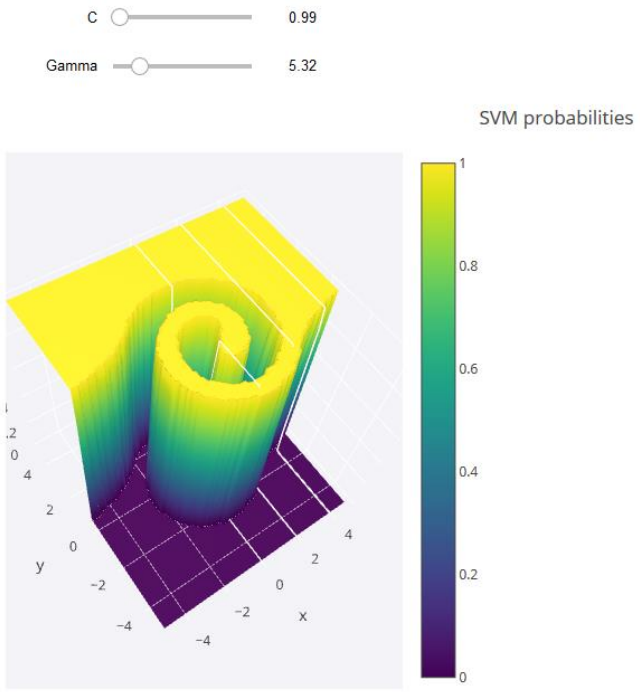


Рисунок 4.15 – Результати при збільшенні параметру радіально базисної функції

Виходячи з результатів, представлених на рисунках вище, можна сказати, що зміна обох з параметрів дуже впливає на результати класифікації. На рисунку 4.15 видно, що при збільшені параметру радіально базисної функції модель сильніше «заточується» під конкретні спостереження вибірки, але при знаходженні оптимального значення параметру, модель становиться більш загальною, що позволяє вірно класифікувати спостереження, які не входили до навчальної вибірки. При дуже великих значеннях параметрів модель утрачає можливість узагальнювати та може класифікувати тільки ті спостереження, що були у навчальній вибірці, що представлено на рисунку 4.16.

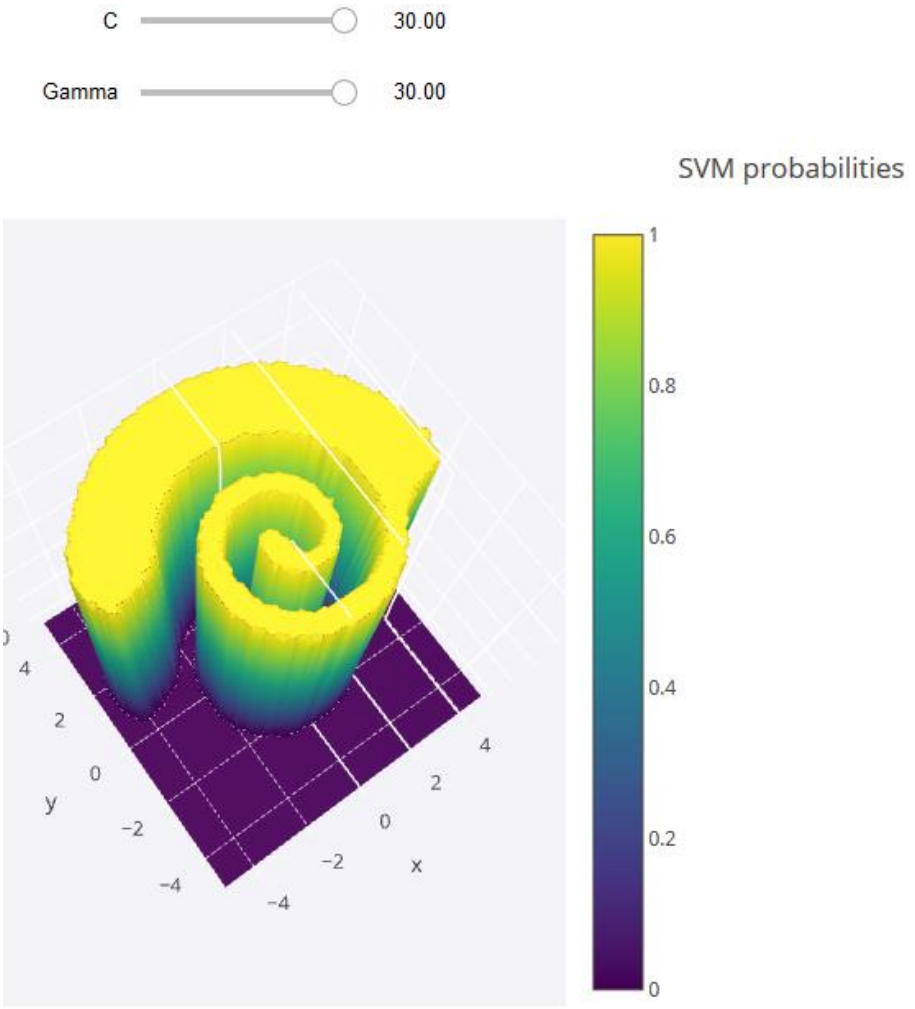


Рисунок 4.16 – Результат моделі при великих значеннях параметрів

Зараз розглянемо роботу алгоритму машини опорних векторів менш тривіальних вибірках. У якості наступного прикладу використовується вибірка «Breast Cancer Wisconsin (Diagnostic) Data Set», яка містить в собі дані про ракові захворювання молочних залоз[19]. Ця вибірка містить в собі риси, які були прораховані на основі оцифрованих результатів тонкоголкової аспіраційної пункційної біопсії взятих з грудної маси.

Для кожного ядра клітини обчислюється різні функції, на основі яких будується вибірка:

- радіус (середній відстань від центру до точок по периметру);
- текстура (стандартне відхилення масштабів);
- периметр;
- область;
- гладкість (локальне відхилення довжин радіуса);
- компактність;
- увігнутість (вираженість увігнутих частин контуру);
- увігнені точки (кількість увігнутих частин контуру);
- симетрія.

На основі цих даних збудована вибірка. По кожній із цих функцій знаходиться середнє значення по клітинам, найгірший показник клітини та стандартна похибка. Також, для кожного з спостережень представлено атрибут відповідей: «M» – злоякісна, «B» - доброякісна. Так як у вибірці багато атрибутів, для візуалізації використаємо метод компресії даних PCA. Але перед компресією необхідно нормалізувати дані. На рисунку 4.17 представлені результати компресії даних за допомогою методу головних компонент.

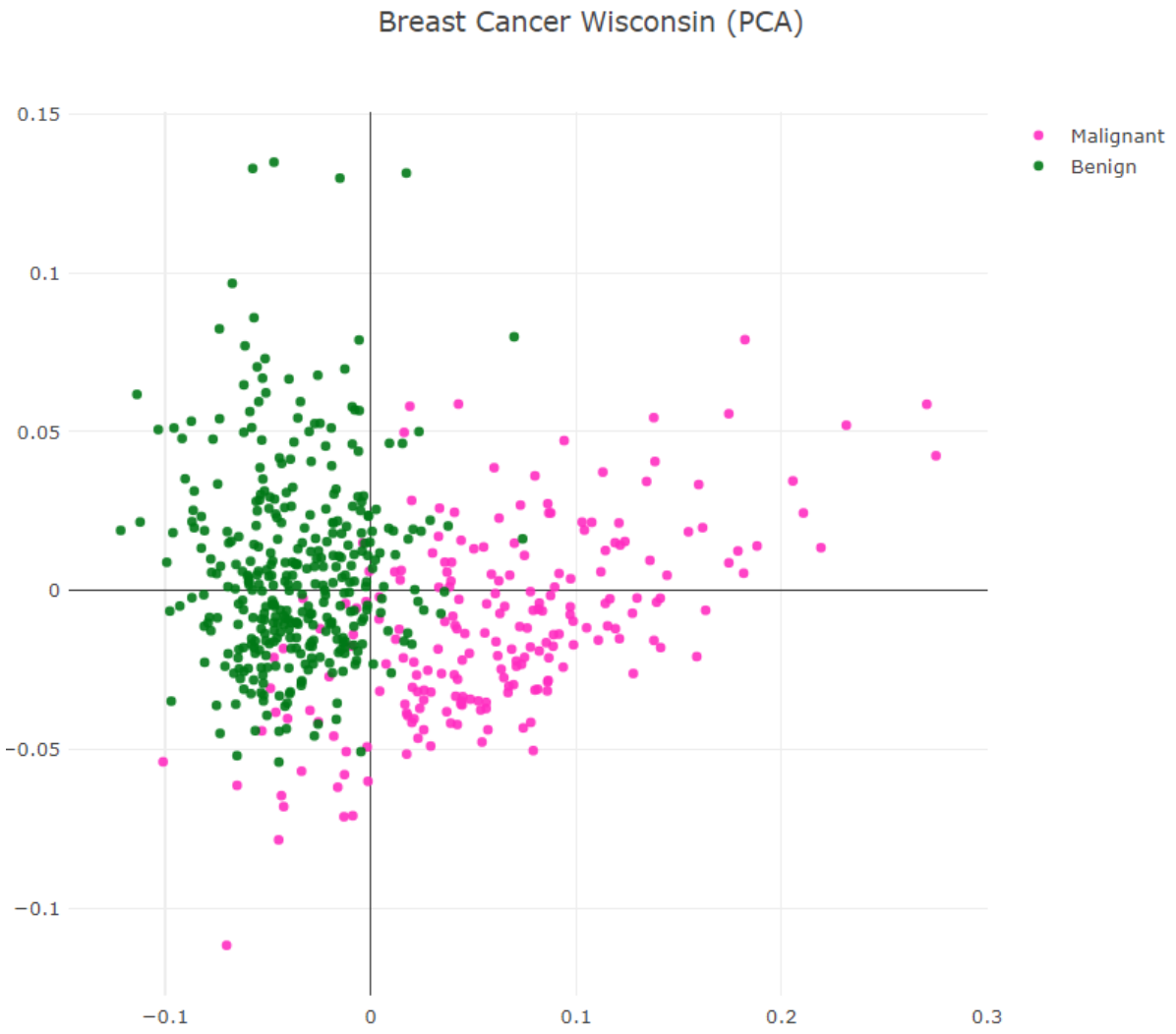
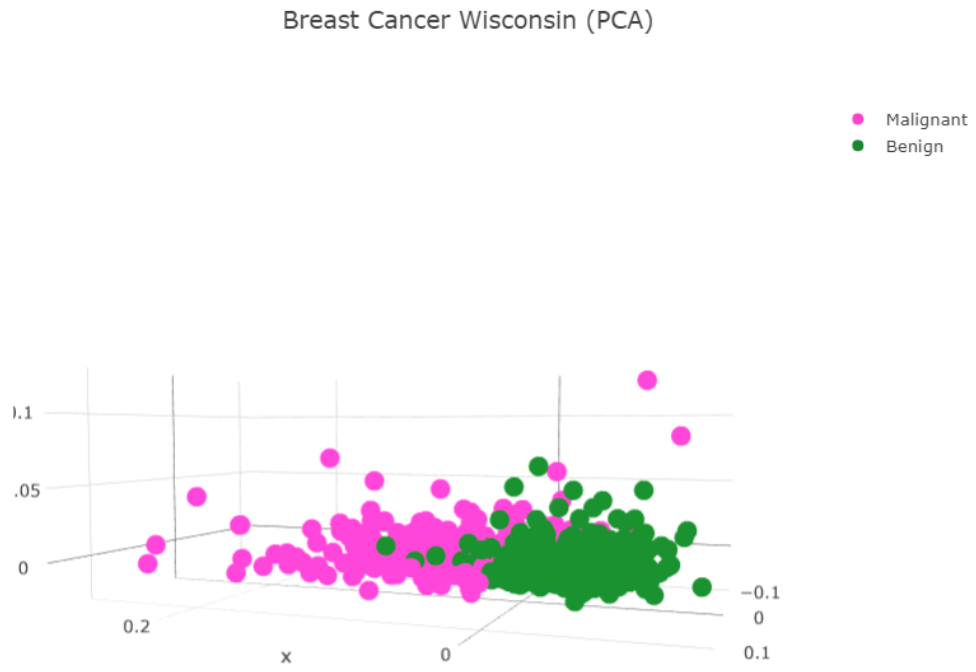


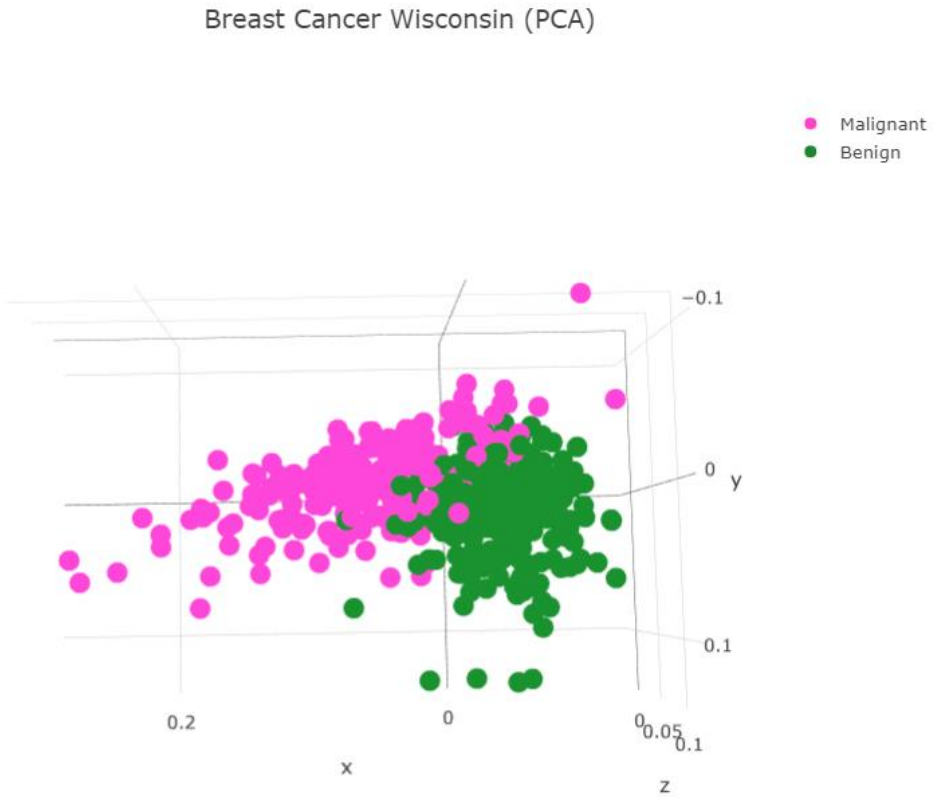
Рисунок 4.17 – Скомпресована вибірка «Breast Cancer Wisconsin»

Як ми можемо бачити, лінійно поділити скомпресовану вибірку неможливо, але ми чітко бачимо, що більшість спостережень з правої сторони належать до злоякісних новоутворень.

Спробуємо зробити компресію даних у тривимірний простір. Це може допомогти нам чи можливо виділити кожен клас та побудувати поділяючу гіперплощину для адекватної класифікації. На рисунках 4.18 представлені результати даної компресії.



а)



б)

Рисунок 4.18 – Скомпресована вибірка у тривимірний простір: а) – вид збоку; б) – вид зверху

Таким чином, на рисунках зверху, можна побачити, що локалізація доброякісних новоутворень знаходиться приблизно в одному місці. Ця закономірність показує те, що існує можливість розділення даної вибірки гіперплощиною для успішного створення моделі класифікації.

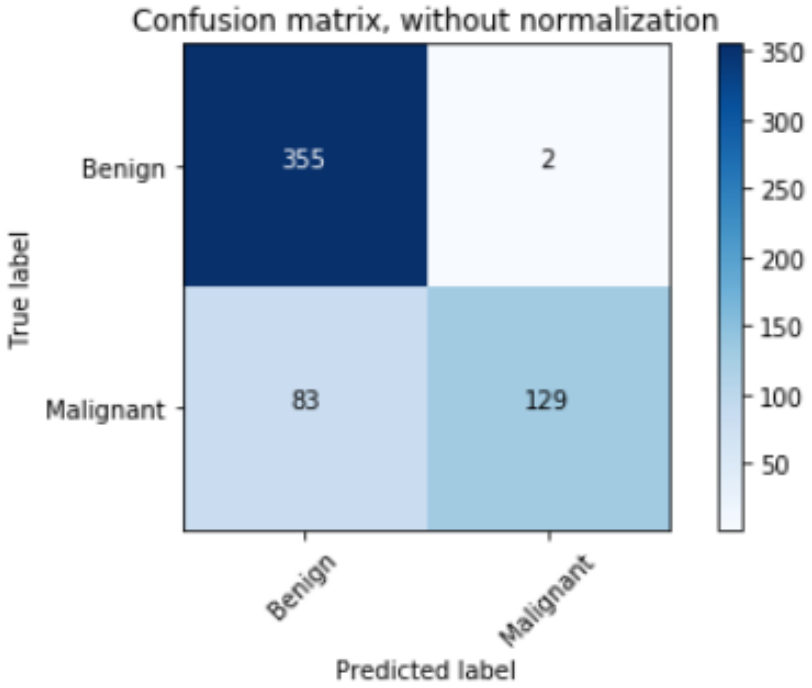
Побудуємо модель машини опорних векторів на даній вибірці без компресії з лінійним ядром. Результати класифікації приведені на рисунку 4.19.

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

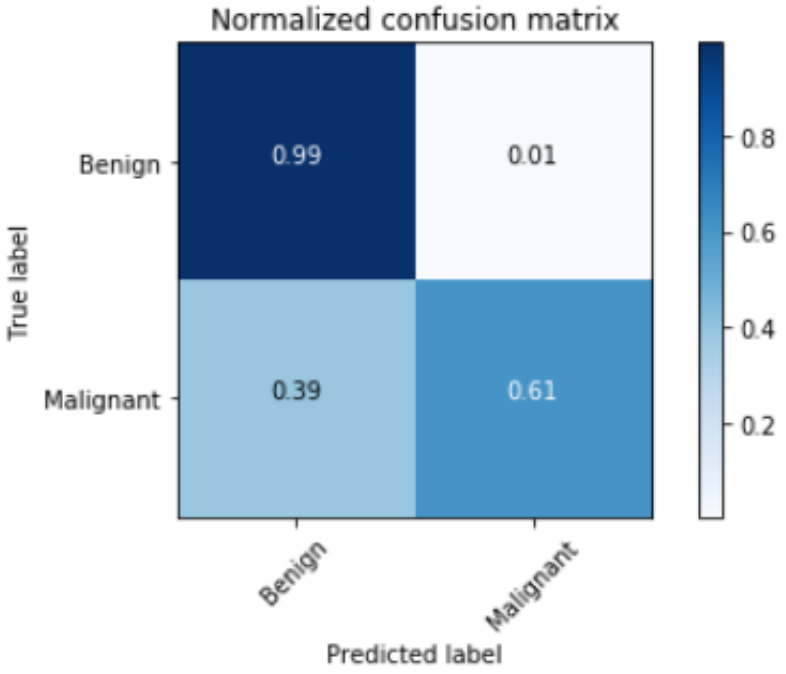
Рисунок 4.19 – Результати класифікації

Як ми бачимо з даних результатів, вірність моделі дорівнює 85%, серед злоякісних новоутворень вірно прокласифіковано близько 60%, а серед доброякісних – близько 99.5%. Виходячи з цих даних, ми можемо сказати, що класифікація злоякісних новоутворень моделлю машини опорних векторів дає недостатньо гарні результати, так як лише на на багато більше половини спостережень були правильно прокласифіковані. Але класифікація злоякісних новоутворень дає відносно гарний результат.

Побудована модель не є достатньо гарною для використання, тому що вірогідність, яка дорівнює 85% відсоткам, не надає достатньо адекватний результат для того, щоб повідомляти його пацієнту, або використовуватися для того, щоб поставити діагноз. Построїмо матрицю похибок на основі результатів даної моделі. На рисунка 4.20 наведена матриця похибок машини опорних векторів з лінійним ядром.



а)



б)

Рисунок 4.20 – Матриці похибки машини опорних векторів з лінійним ядром: а) – кількісна оцінка; б) – оцінка в процентному співвідношенні

На рисунку 4.21 показано як виглядає простір спостережень при класифікації за допомогою машини опорних векторів з лінійним ядром.

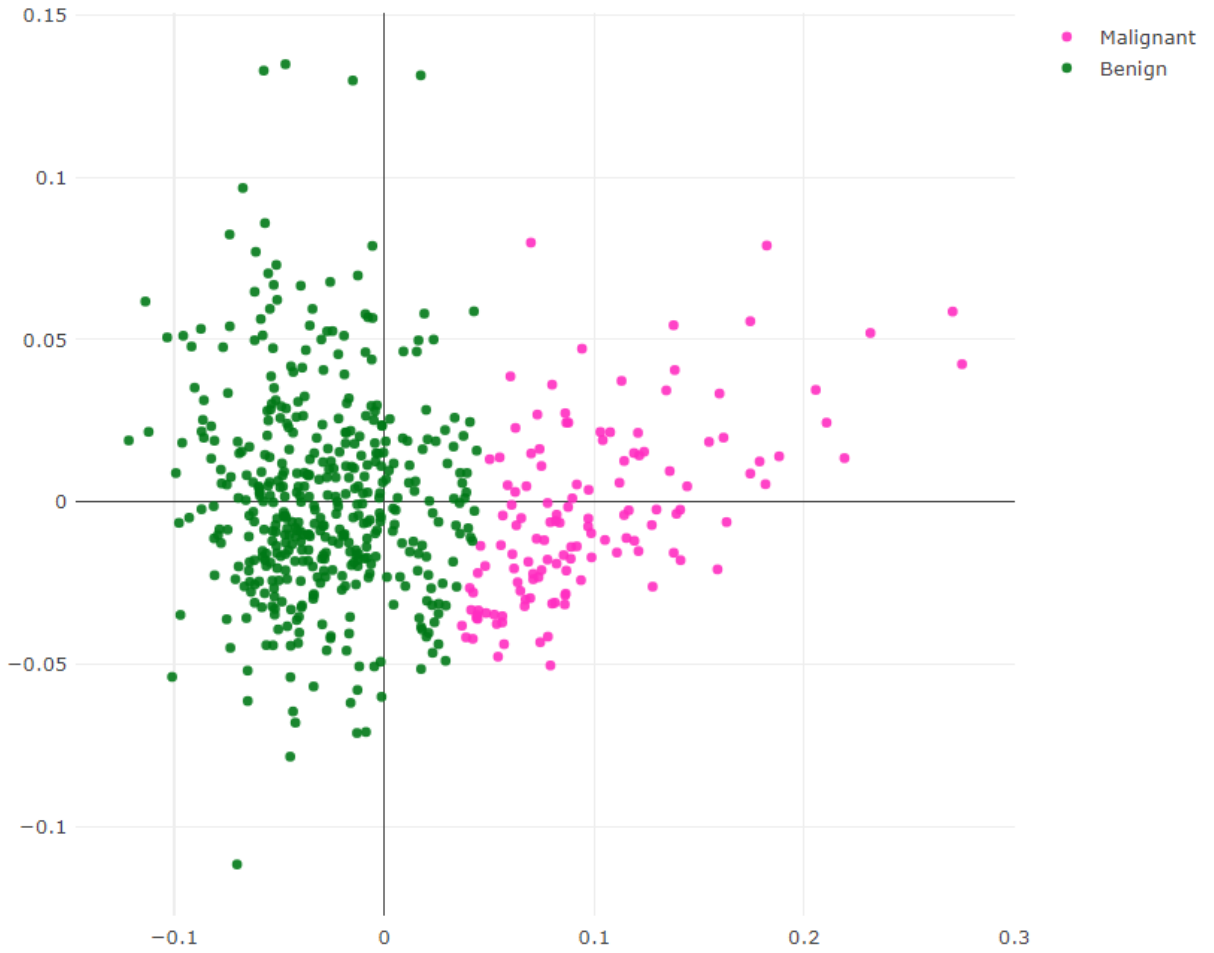


Рисунок 4.21 – Скомпресовані дані з результатами класифікації

Таким чином, проаналізувавши дані результати, можемо сказати, що використовувати лінійне ядро для даної моделі не достатньо гарним рішенням.

Спробуємо використати радіально базисне ядро замість лінійного. Результати класифікації представлені на рисунку 4.22.

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

Рисунок 4.22 – Результат класифікації з радіально базисним ядром

Як ми бачимо з даних результатів, модель з радіально базисним ядром дає

100% правильних класифікацій, що практично неможливо з даними реального світу. Даний факт може вказувати на те, що модель перенавчена на даних та «заточилася» під усі данні. Для перевірки адекватності моделі з радіально базисним ядром, замість використовування усіх даних для навчання, розділимо вибірку на навчальну та тестову. Розмір тестової вибірки – це 20% від усієї вибірки.

При виборі параметру радіально базисної функції, яка дорівнює 0.1, що відповідає попередній моделі, ми отримуємо результати, які представлені на рисунку 4.23.

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

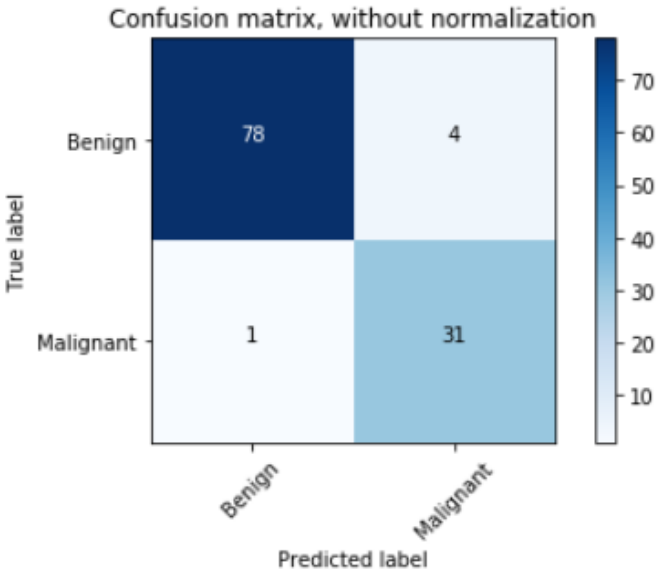
Рисунок 4.23 – Результати класифікації на тестовій вибірці

Результати цієї моделі відносять усі спостереження до класу доброякісних новоутворень. Даний факт означає те, що попередня модель була перенавчена та вона «запам’ятала» усі спостереження класу злоякісних новоутворень, та всі інші автоматично відносило до другого класу. Щоб уникнути такого перенавчання, необхідно зменшити параметр радіально базисної функції, що, як показано на попередніх вибірках, повинно зробити модель більш загальною. Результати моделі з параметром радіально базисної функції, яка дорівнює 0.001, представлені на рисунку 4.24.

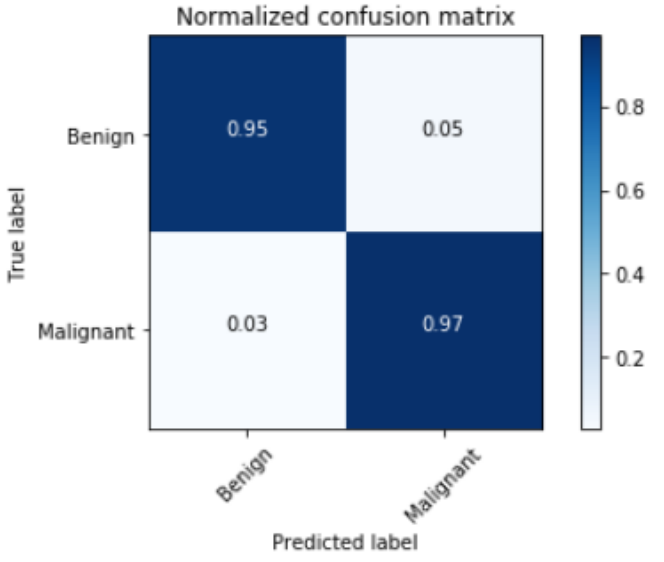
```
Model score: 0.956140350877193
Malignant score: 0.96875 Benign score: 0.9512195121951219
```

Рисунок 4.24 – Результат моделі зі модифікованим параметром радіально базисної функції

Дані результати вказують на те, що при дана модель більш узагальнена, на відміну від попередньої. Точність на тестовій вибірці дорівнює більш ніж 95%, що являється гарним показником. Та майже 97% злоякісних новоутворень було знайдено вірно. Матриці похибок представлені на рисунках 4.25.



а)



б)

Рисунок 4.25 – Матриці похибок моделі з радіально базисним ядром: а) – кількісна оцінка; б) – оцінка в процентному співвідношенні

Графік класифікації даної вибірки представлений на рисунку 4.26.

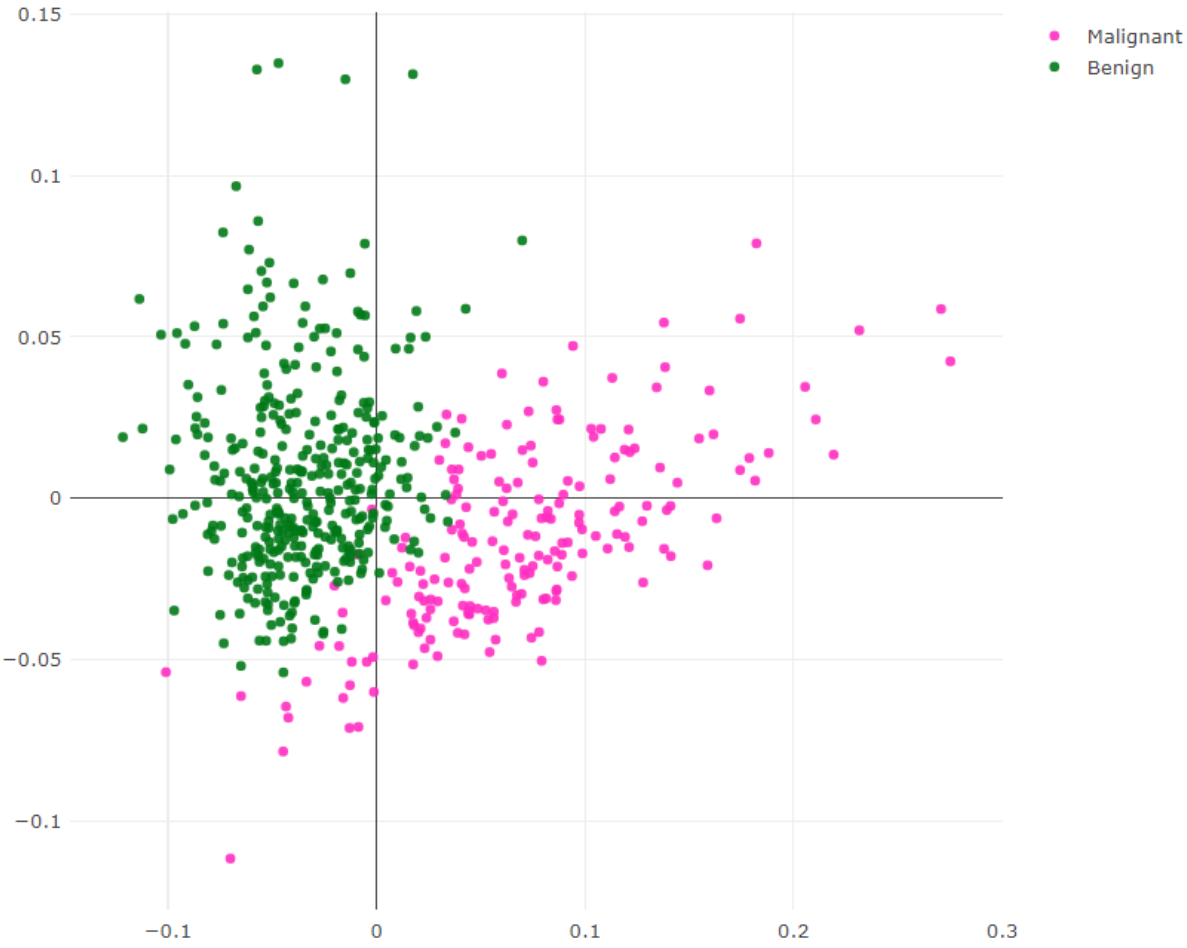


Рисунок 4.26 – Результати класифікації моделі з налаштованим параметром радіально базисної моделі

Таким чином, проаналізувавши результати усіх моделей, які були розглянуті під час експериментальних досліджень, можна сказати, що машина опорних векторів показує адекватні результати при класифікації, але тільки за умови, що параметри моделі повинні бути вірно налаштовані.

4.3 Використання SVM та Fuzzy SVM у задачах мултикласової класифікації

У даному підрозділі ми розглянемо та порівняємо роботу нечіткої машини опорних векторів зі звичайною машиною опорних векторів у задачах мультикласової класифікації. Для вирішення цієї задачі був використаний датасет з ресурсу «Kaggle.com»[20].

Kaggle.com – це платформа для спеціалістів з аналізу даних та експертів з машинного навчання. На базі цієї платформи різні компанії організують змагання по машинному навчанню та аналізу даних. Також, цей сайт використовується для спілкування між учасниками та щоб ділитися досвідом.

Для вирішення задачі мультикласової класифікації було використано датасет зі змагання під назвою «Toxic Comment Classification Challenge»[21]. У даному змаганні користувачам пропонується вирішити задачу мультикласової класифікації коментарів з ресурсу Wikipedia. Головна ціль даного змагання – створити модель NLP для класифікації непристойних коментарів та віднести коментар до 7 різних категорій. Також, треба зауважити, що дана вибірка є дуже небалансованою. Спостереження з даної вибірки продемонстровано на рисунку 4.27.

comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate	none
translation: You are [illegible]	1	1	1	0	1	0	0
Bleh \n\nNahh.. Nothing is relevant anymore.\n...	0	0	0	0	0	0	1
Warning\n\nWhen people have moved on to other ...	0	0	0	0	0	0	1
Why you Little...\nI hope you get banned from ...	1	0	1	0	1	0	0
thrash metal and speed metal there	0	0	0	0	0	0	1

Рисунок 4.27 – Приклади коментарів з вибірки

У даній вибірці знаходиться сім класів, до яких може відноситися коментар:

- токсичний;
- дуже токсичний;
- нецензурний;
- погрожуючий;
- образливий;
- ненависний до ідентичності;
- жоден.

Як було зазначено у розділі 3.4 для використання машини опорних векторів у мультикласовій класифікації необхідно створити декілька поділяючих гіперплощин. Кількість гіперплощин дорівнює кількості класів заданої вибірки. Виходячи с цього, ми створюємо 7 різних моделей машини опорних векторів для кожного з класів при використанні підходу один проти всіх.

Дана задача відноситься до класу задач NLP – Natural Language Processing (Обробка природної мови). Одним із найважливіших кроків в даному класі задач – є вибір представлення слів у векторному просторі. Векторне подання слів у векторному просторі представляє з себе набір дійсних чисел, отриманих за допомогою певної моделі обробки природної мови. Дані вектори використовуються для подальшої обробки та роботи з ними. За допомогою даних векторів вирішуються основні завдання обробки природної мови: аналіз текстів, генерація тексту, переклад текстів на інші мови.

Більшість методів базується на чисельній відстані між векторами, або кутом між ними, наприклад – косинусна відстань між двома векторами. З розвитком методів представлення слів у векторному просторі

приходять нові методи оцінки схожості слів, засновані на відмінності окремих елементів векторів, які є певними смисловими вимірами.

У даній роботі буде використаний один з найпростіших способів представлення текстової інформації у векторному просторі – TF-IDF[22].

TF-IDF (Term frequency, inverse document frequency) – це спосіб представлення текстових речень у векторному просторі, яке представляє собою співвідношення кількості зустрічей слова та загальної кількості слів у документі.

Для навчання та тестування моделі було використано 25000 спостережень для прискорення навчання. Як один з шагів попередньої обробки даних з вибірки були видалені усі символи, які не належать до латинського алфавіту. Також, для створення матриці TF-IDF було використано біграмми. Усього дана матриця має 95867 слів та словосполучень. Опис TF-IDF матриці наведено на рисунку 4.28.

```
tf_idf_df
```

```
<25000x95867 sparse matrix of type '<class 'numpy.float64'>'
        with 2505260 stored elements in Compressed Sparse Row format>
```

Рисунок 4.28 – Опис TF-IDF матриці

Всього в даних використовується 7 класів, в вибірці зустрічається 28 різних комбінацій даних класів. Це означає, що гіперпростір спостережень буде мати 28 регіонів, які утворюються за допомогою комбінацій 7 поділяючих гіперплощин, знайдених за допомогою машини опорних векторів.

У попередньому підрозділі була описана важливість вибору гіпепараметрів моделі машини опорних векторів для класифікації. Тому, за допомогою серій навчань моделі з різними параметрами були вибрані кращі.

Дивлячись на розподіл класів у вибірці, можна сказати, що класи не являються збалансованими. Розподілення класів представлено на рисунку 4.29.

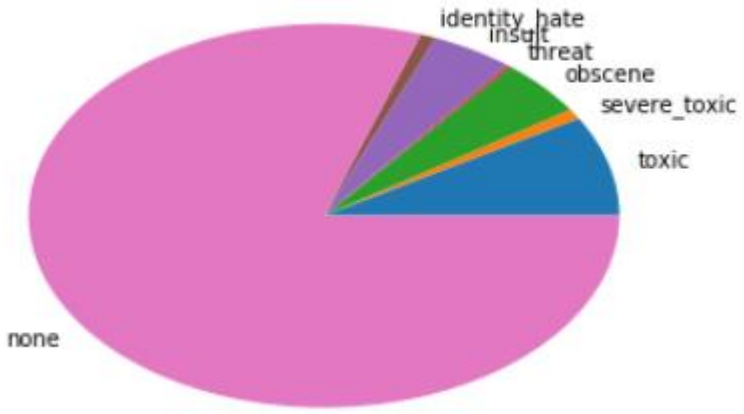


Рисунок 4.29 – Розподілення класів

Виходячи з цього, ми можемо сказати, що використовувати точність в якості метрики не має змісту. Тому, в якості перевірки моделей машини опорних векторів на дієздатність будемо використовувати F1-score. F1-score – це подвійне відношення добутку повноти та точності до їх суми. У цьому випадку ми зможемо адекватно оцінити нашу модель навіть коли вибірка є незбалансованою.

Для тестування результатів вибірка була розділена на навчальну та тестову вибірки по 80% та 20% відповідно. Розглянемо результати моделей для кожного з класів. Результати класифікації приведені на рисунку 4.30.

```
toxic:
Confusion matrix:
[[5608   49]
 [ 215  378]]
F1_Score: 0.7411764705882353

severe_toxic:
Confusion matrix:
[[6170   13]
 [  50   17]]
F1_Score: 0.35051546391752575

obscene:
Confusion matrix:
[[5872   21]
 [ 122  235]]
F1_Score: 0.7667210440456771

threat:
Confusion matrix:
[[6233    0]
 [  16    1]]
F1_Score: 0.1111111111111111

insult:
Confusion matrix:
[[5886   35]
 [ 159  170]]
F1_Score: 0.6367041198501874

identity_hate:
Confusion matrix:
[[6182    7]
 [  52    9]]
F1_Score: 0.23376623376623376

MEAN F1_SCORE: 0.4733324072131617
```

Рисунок 4.30 – Результати класифікації SVM

Як ми можемо бачити, середня F1-score моделей менше 0.5, що э не дуже гарним результатом. Також, для класів «threat», «identity_hate» та «severe_toxic» значення дуже погані. Якщо ми подивимося на рисунок 4.29, то побачимо, що данні класи мають дуже малу кількість зустрічей у датасеті, відносно інших класів. Навіть не дивлячись на те, що вибірка незбалансована, інші класи мають F1-score вище.

Для реалізації нечіткої машини опорних векторів необхідно для кожних спостережень, які не відносяться до жодного із регіонів, знайти відстань до кожного з них. Таким чином, усі спостереження, які знаходяться у некласифікованих регіонах будуть мати відстані до кожного з регіонів. Для цього знайдемо евклідову відстань від кожного з некласифікованих спостережень до усіх регіонів. Після цього необхідно вибрати найближчий регіон та присвоїти клас до цього спостереження. На рисунку 4.31 ми можемо побачити результати після присвоєння значень найближчих регіонів спостереженням без класу.

```
toxic:
F1_score: 0.7711941659070192

severe_toxic:
F1_score: 0.3541666666666667

obscene:
F1_score: 0.7900466562986005

threat:
F1_score: 0.21052631578947367

insult:
F1_score: 0.6666666666666666

identity_hate:
F1_score: 0.32558139534883723

MEAN F1_SCORE: 0.519696977779544
```

Рисунок 4.31 – Результати нечіткої машини опорних векторів

Таким чином, аналізуючи результати машини опорних векторів та нечіткої машини опорних векторів можна сказати, що нечітка машина опорних векторів показує кращі результати навіть на незбалансованих данних.

ВИСНОВКИ

В ході проходження даної атестаційної роботи була розглянута предметна область і визначена формальна постановка задачі. Були розглянуті поняття навчання з учителем і класифікації. Поверхнево були описані основні алгоритми класифікації, які використовуються в даний час.

Був проведений аналіз алгоритму нечіткої машини опорних векторів і описана робота його дії. Головний принцип роботи методу опорних векторів – розбиття простору ознак на дві частини, розділені гіперплощиною. Даний алгоритм працює по принципу бінарної класифікації, а у випадку мультикласової класифікації використовується підхід «один проти всіх». Спостереженню, що надходить на вхід моделі, буде присвоєно клас, в залежності від розташування даного спостереження від поділяючої гіперплощини. У випадку мультикласової класифікації поділяюча гіперплощина ділить простір на дві частини, де до одної відносяться спостереження одного класу, а другої – всі інші класи. Для такої класифікації створюється декілька моделей, кількість яких дорівнює кількості класів.

Метою роботи стало рішення проблеми обробки даних з метою класифікації. Класифікатор на основі нечіткої машини опорних векторів повинен бути навчений на прецедентах, що виявляють собою пару – спостереження-відповідь. Під час проведення експериментальних досліджень було розглянуто вибірку коментарів, на основі яких було продемонстровано як вибір параметрів моделі та нечіткість машини опорних векторів впливає на результати обробки даних та їх класифікацію. За допомогою побудованих графіків були наглядно зображені результати проведеного досліду, з яких ми можемо зробити висновок, що нечітка машина опорних векторів має кращу точність ніж SVM без механизму нечіткості.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Nils J. INTRODUCTION TO MACHINE LEARNING / J. Nilsson Nils. Stanford, 1998. 168 p.

2. Ben-Hur A. A User's Guide to Support Vector Machines / A. Ben-Hur, J. Weston. 2010. 18 p.

3. Kotsiantis S. Supervised Machine Learning: A Review of Classification Techniques / S. B. Kotsiantis // Informatica 31 / S. B. Kotsiantis. Tripolis, 2007. С. 249–268.

4. Бодянский Е. В. Искусственные нейронные сети: архитектуры, обучение, применения / Е. В. Бодянский, О. Г. Руденко. Харьков: Телетех, 2004. 369 с.

5. Mhaskar H. How to Choose an Activation Function / H. Mhaskar, C. Micchelli. 1995. 8 p.

6. Hicken J. Gradient-Based Optimization / J. Hicken, J. Alonso // AA222 - Introduction to Multidisciplinary Design Optimization / J. Hicken, J. Alonso. Stanford, 2012. С. 53–77.

7. Wang L. Support Vector Machines: Theory and Applications / Lipo Wang. Berlin: Springer, 2005. 28 p.

8. Вапник, В.Н. Теория распознавания образов (статистические проблемы обучения) / за ред. В.Н. Вапника, А.Я. Червоненкис М.: Наука. 1974. 416 с.

9. Haykin, S. Neural Networks. A Comprehensive Foundation. / S. Haykin // Upper Saddle River, N.J.: Prentice Hall, Inc. 1999. 842 p;

10. Suykens, J.A.K. Least Squares Support Vector Machines. / J.A.K. Suykens, T.V., Gestel, J.D. Brabanter, B.D. Moor, J. Vandewalle Singapore: World Scientific. 2002. 294 p.

11. Вапник, В.Н. Восстановление зависимостей по эмпирическим данным / за ред. В.Н. Вапника М.: Наука. 1979. 448 с.

12. Improved modules URL: https://ddocs.python.org/3.6/whatsnew/3.6.html#improved-modules (дата звернення: 10.04.2020).

13. Мюллер А. Введение в машинное обучение с помощью Python / А. Мюллер, С. Гвидо. Москва, 2017. 393 с.

14. Jupyter Notebook Documentation URL: http://jupyter.org/documentation (дата звернення: 12.04.2020).

15. Scikit-learn Documentation URL: http://scikit-learn.org/stable/documentation.html (дата звернення: 13.04.2020).

16. Numpy Library Documentation URL: http://www.numpy.org/ (дата звернення: 13.04.2020).

17. Anaconda User Guide URL: https://enterprise-docs.anaconda.com/en/latest/ (дата звертання: 13.04.2020).

18. Plotly python API URL: https://plot.ly/python/ (дата звернення: 15.04.2020).

19. Breast Cancer Dataset URL: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer +Wisconsin+%28Diagnostic%29 (дата звернення: 20.04.2020).

20. Kaggle platform for ML competitions URL: https://www.kaggle.com/ (дата звернення: 25.04.2020)

21. Toxic Comments Dataset URL: https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge (дата звернення: 25.04.2020)

22. Ramos, J. Using TF-IDF to determine word relevance in document queries. / J. Ramos. 2003, 4 p.