Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

за темою *Квантовий метод опорних векторів: розробка варіаційних алгоритмів для реалізації класифікації даних у технічних, природничих і соціально-економічних системах*

Галузь знань *12 – Інформаційні технології*
Шифр і назва галузі знань

Спеціальність *122 – Комп'ютерні науки*
Шифр і назва спеціальності

Освітня програма *Комп'ютерні науки*
Назва освітньої програми

Виконав: *студент 4 курсу, група КН-19-1* _____ *М.А. Казіонов*
Курс, група виконавця — Підпис — Ініціали, прізвище

Керівник: *д.т.н., проф., завідувач кафедри КН* _____ *О.В.Бармак*
Науковий ступінь, посада — Підпис — Ініціали, прізвище

Нормоконтроль: *к.т.н., доцент кафедри КН* _____ *Р.О. Багрій*
Науковий ступінь, посада — Підпис — Ініціали, прізвище

До захисту допускаю:
Зав. кафедри КН, д.т.н., професор _____ *О.В. Бармак*
Підпис — Ініціали, прізвище

01 06 2023 р.

Хмельницький 2023

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет *інформаційних технологій*
Кафедра *комп'ютерних наук*
Освітній ступінь *бакалавр*
Галузь знань *12 – Інформаційні технології*
Спеціальність *122 – Комп'ютерні науки*

ЗАТВЕРДЖУЮ
Завідувач кафедри *комп'ютерних наук*

_____
(підпис)
д.т.н., професор О.В. Бармак
«06» 03 *2023* року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

1. Тема кваліфікаційної роботи бакалавра: *«Квантовий метод опорних векторів: розробка варіаційних алгоритмів для реалізації класифікації даних у технічних, природничих і соціально-економічних системах»*

2. Завдання видано студенту *Казіонову Максиму Андрійовичу*
(прізвище, ім'я, по батькові)

3. Керівник роботи *завідувач кафедри КН, Бармак Олександр Володимирович*
(посада, прізвище, ім'я, по батькові)

4. Затверджено наказом університету від «1» 03 *2023* р. № 5

5. Дата видачі завдання студенту: «3» 03 *2023* р.

6. Строк подання студентом роботи на кафедру: «____» *2023* р.

7. Зміст пояснювальної записки (перелік задач) та вихідні дані: задачі дослідження: огляд методів та засобів машинного навчання; огляд квантових обчислень для великої кількості даних; огляд технічних, природничих та соціально-економічних інформаційних систем, що потребують класифікації та мають великі об'єми даних; запропонувати способи реалізації квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем; провести реалізацію квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем.

8. Календарний план виконання кваліфікаційної роботи бакалавра:

№	Назва етапів (розділів) кваліфікаційної роботи бакалавра	Термін виконання	Примітка
1	*Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи бакалавра з керівником*	*23.12.2022*	*виконано*
2	*Ознайомлення з предметною областю, формулювання мети та задач дослідження, визначення об'єкта та предмета дослідження*	*01.02.2023*	виконано
3	*Робота над розділом 1 – Огляд принципів роботи квантових обчислень для задач класифікації, що мають великий об'єм даних*	*01.02.2023*	виконано
4	*Робота над розділом 2 – Особливості реалізації квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем*	*01.03.2023*	виконано
5	*Робота над розділом 3 – Програмна реалізація з застосуванням квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем*	*30.04.2023*	виконано
6	*Оформлення пояснювальної записки згідно вимог*	*31.05.2023*	виконано
7	*Підготовка статті до журналу, попередній захист кваліфікаційної роботи бакалавра*	*02.05.2023*	виконано
8	*Захист кваліфікаційної роботи бакалавра на засіданні Екзаменаційної комісії*	*червень 2023*	виконано

навець: *студент 4 курсу, група КН-19-1* — Курс, група виконавця — Підпис — *М.А. Казіонов* — Ініціали, прізвище

зник: *д.т.н., проф., завідувач кафедри КН* — Науковий ступінь, посада — Підпис — *О.В. Бармак* — Ініціали, прізвище

Анотація

Тема кваліфікаційної роботи бакалавра: *Квантовий метод опорних векторів: розробка варіаційних алгоритмів для реалізації класифікації даних у технічних, природничих і соціально-економічних системах*

Виконавець кваліфікаційної роботи бакалавра: *студент групи КН-19-1 Казіонов Максим Андрійович*

Керівник кваліфікаційної роботи бакалавра: *д.т.н., професор, завідувач кафедри КН Бармак Олександр Володимирович*

Кваліфікаційна робота бакалавра містить:

Пояснювальна записка				Кількість додатків
Сторінок	Рисунків	Таблиць	Джерел інформації	
74	54	6	52	4

Мета кваліфікаційної роботи полягає в дослідженні особливостей використання квантового методу опорних векторів (з варіаційними алгоритмами) для реалізації класифікації даних у технічних, природничих і соціально-економічних системах з великим об'ємом даних.

Для досягнення поставленої мети визначені наступні завдання дослідження: огляд методів та засобів машинного навчання; огляд квантових обчислень для великої кількості даних; огляд технічних, природничих та соціально-економічних інформаційних систем, що потребують класифікації та мають великі об'єми даних; запропонувати способи реалізації квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем; провести реалізацію квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем.

Ключові слова: штучний інтелект, машинне навчання, квантові обчислення, квантовий метод опорних векторів, технічні інформаційні системи, природничі інформаційні системи, соціально-економічні інформаційні системи.

Виконавець: *студент 4 курсу, група КН-19-1* (Курс, група виконавця) _____ (Підпис) *М.А. Казіонов* (Ініціали, прізвище)

Зміст

Перелік скорочень 3

Вступ 4

Розділ 1 Огляд принципів роботи квантових обчислень для задач класифікації, що мають великий об’єм даних 6

1.1 Огляд методів та засобів машинного навчання 6

1.2 Огляд сучасного стану квантових обчислень 10

1.3 Технічні, природничі та соціально-економічних інформаційні системи, що потребують застосування методів класифікації та мають великі об’єми даних 14

1.4 Мета та завдання дослідження 17

Розділ 2 Особливості реалізації квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем 18

2.1. Класичний метод опорних векторів 18

2.2. Квантовий метод опорних векторів 23

2.3 Оцінка якості класифікації та обробка даних 30

2.4 Особливості використання квантового методу опорних векторів в кібербезпеці 36

2.5 Особливості використання квантового методу опорних векторів в екології 39

2.6 Особливості використання квантового методу опорних векторів для економічних систем та фінансової аналітики 41

2.7 Висновки до розділу 2 43

Розділ 3 Програмна реалізація з застосуванням квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем 44

3.1 Програмна реалізація з застосуванням квантового методу опорних векторів для класифікації DDoS атак 44

3.2 Програмна реалізація з застосуванням квантового методу опорних векторів для класифікації якості води 53

3.3 Програмна реалізація з застосуванням квантового методу опорних векторів для класифікації відтоку клієнтів 59

3.4 Висновки до розділу 3 67

Висновки 69

Перелік посилань 70

Додатки

Перелік скорочень

Скорочення, термін, позначення	Пояснення
КРБ	Кваліфікаційна робота бакалавра
КН	Комп’ютерні науки
ШІ	Штучний інтелект
SVM	Метод опорних векторів
QSVM	Квантовий метод опорних векторів
PCA	Метод головних компонент
DDoS	Distributed Denial of Service

Вступ

Кваліфікаційна робота бакалавра присвячена дослідженню квантового методу опорних векторів, розробці варіаційних алгоритмів та їх особливостям реалізації для класифікації даних у технічних, природничих і соціально-економічних системах.

Актуальність теми. Машинне навчання та квантові обчислення – це дві технології, кожна з яких може змінити спосіб виконання обчислень, щоб вирішити проблеми, які раніше були недосяжними. Такі методи машинного навчання як метод опорних векторів (SVM) є найвідомішими методами для вирішення проблем класифікації. Однак існують обмеження для успішного вирішення задач, коли об’єм даних стає достатньо великим, а функції ядра стають обчислювально складними. Ключовим елементом прискорення обчислень, яке забезпечують квантові алгоритми, є використання експоненціально великого квантового простору станів через контрольоване заплутування та перешкоди. Викликає інтерес дослідження квантовий варіаційний класифікатор який працює за допомогою використання варіаційної квантової схеми для класифікації навчального набору за прямою аналогією до звичайних SVM. Він відноситься до нового класу інструментів для дослідження застосувань квантових комп’ютерів для машинного навчання.

Мета і завдання кваліфікаційної роботи. Мета кваліфікаційної роботи полягає в дослідженні особливостей використання квантового методу опорних векторів (з варіаційними алгоритмами) для реалізації класифікації даних у технічних, природничих і соціально-економічних системах з великим об’ємом даних.

Для досягнення поставленої мети визначені наступні завдання дослідження:

- огляд методів та засобів машинного навчання;
- огляд квантових обчислень для великої кількості даних;
- огляд технічних, природничих та соціально-економічних інформаційних систем, що потребують класифікації та мають великі об’єми даних;
- запропонувати способи реалізації квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем;

- провести реалізацію квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем.

Об’єкт дослідження – процес використання квантових обчислень і квантового машинного навчання для задач класифікації за допомогою метода опорних векторів.

Предмет дослідження – моделі, алгоритми та засоби квантових обчислень для методу опорних векторів для використання у технічних, природничих і соціально-економічних інформаційних системах.

Розділ 1 Огляд принципів роботи квантових обчислень для задач класифікації, що мають великий об’єм даних

1.1 Огляд методів та засобів машинного навчання

Машинне навчання та штучний інтелект (ШІ) - дві технології 21-го століття, що найшвидше розвиваються [1]. Вони змінюють те, як ми взаємодіємо з технологіями, і, в свою чергу, змінюють світ навколо нас. Машинне навчання та штучний інтелект часто використовують як взаємозамінні поняття, але вони стосуються різних аспектів однієї і тієї ж сфери. Машинне навчання - це підрозділ штучного інтелекту, який фокусується на алгоритмах, здатних навчатися і робити прогнози на основі даних.

Історія машинного навчання та штучного інтелекту бере свій початок на початку 20-го століття, з появою перших комп'ютерів [2]. У 1950-60-х роках дослідники почали розробляти алгоритми, які могли б виконувати базові завдання, такі як розпізнавання образів і прийняття рішень. Це призвело до розробки перших експертних систем, які могли імітувати процеси прийняття рішень людьми-експертами в певних галузях.

У 1970-80-х роках дослідники почали розробляти більш досконалі алгоритми машинного навчання, такі як нейронні мережі та дерева рішень. Ці алгоритми можна було використовувати для навчання машин розпізнавати складні закономірності та приймати рішення на основі даних. Однак обмеженість обчислювальних потужностей на той час означала, що ці алгоритми були обмеженими у своїх можливостях.

У 1990-х і 2000-х роках розвиток обчислювальних ресурсів і сховищ даних призвів до відновлення інтересу до машинного навчання та штучного інтелекту. Дослідники розробили нові алгоритми і методи, такі як машини опорних векторів, випадкові ліси і глибоке навчання, які можуть обробляти і аналізувати великі обсяги даних ефективніше, ніж будь-коли раніше. Розвиток цих нових методів призвів до різкого зростання інтересу та інвестицій у машинне навчання та штучний інтелект. У 2011 році комп'ютер Watson від IBM переміг чемпіонів у

вікторині "Jeopardy!", продемонструвавши потенціал алгоритмів машинного навчання перевершувати людей у виконанні певних завдань [3].

У 2015 році програма AlphaGo від Google потрапила в заголовки газет, обігравши чемпіона світу з гри в го, яка вважалася занадто складною для машин [4]. Це стало важливою подією в розвитку штучного інтелекту і продемонструвало потенціал алгоритмів машинного навчання для вирішення складних завдань у широкому діапазоні галузей.

Сьогодні машинне навчання та штучний інтелект використовуються в широкому спектрі областей, від самокерованих автомобілів до медичної діагностики та фінансового трейдингу. Оскільки все більше даних стає доступним, а обчислювальні потужності продовжують зростати, потенційне застосування машинного навчання та штучного інтелекту тільки збільшуватиметься.

На сьогоднішній день існує багато алгоритмів, технік і методів машинного навчання, які широко використовуються в цій галузі. Кожен з цих підходів має свої сильні та слабкі сторони і підходить для різних типів задач. Одним з таких алгоритмів є метод опорних векторів (SVM).

Метод опорних векторів (SVM) - це популярний алгоритм машинного навчання, який особливо добре підходить для задач бінарної класифікації [5]. SVM працюють, знаходячи гіперплощину, яка найкраще розділяє два класи у багатовимірному просторі ознак. Гіперплощина вибирається таким чином, щоб максимізувати відстань між двома класами - тобто відстань між гіперплощиною і найближчими точками даних з кожного класу. Це допомагає забезпечити стійкість SVM до шуму та викидів у даних. На рис 1.1 наведено простий приклад роботи методу опорних векторів.

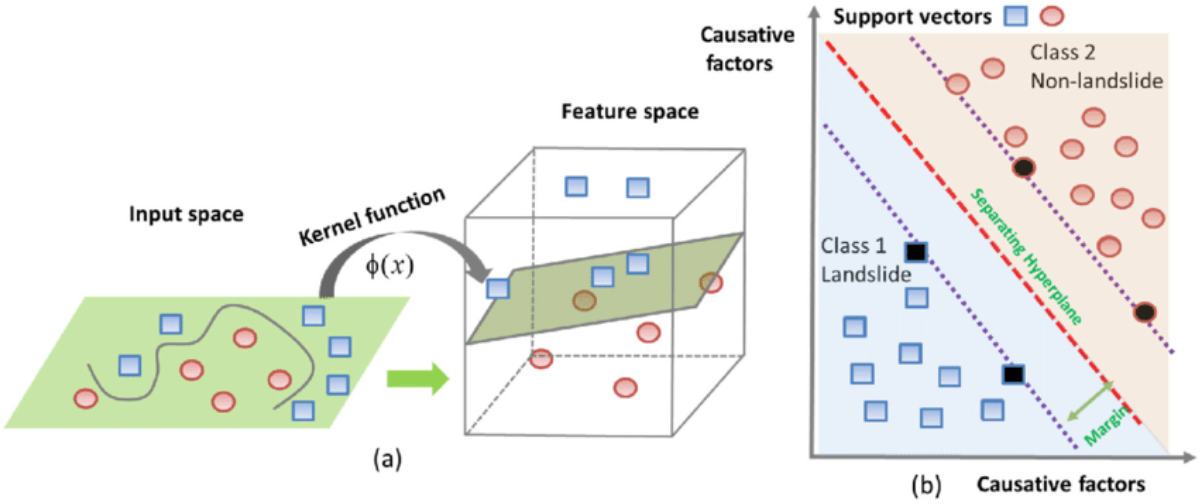


Рисунок 1.1 – Ілюстрація роботи методу опорних векторів [7]

SVM можна використовувати з різними функціями ядра, які перетворюють дані в простір вищої розмірності, де легше розділити класи. Деякі поширені функції ядра включають лінійну, поліноміальну та радіально-базисну функцію (RBF). Вибір функції ядра може мати значний вплив на продуктивність SVM, і, можливо, її доведеться налаштовувати залежно від конкретної задачі та набору даних.

SVM мають кілька переваг над іншими алгоритмами класифікації. Вони ефективніше обробляють дані високої розмірності і можуть використовуватися з нелінійними межами прийняття рішень. Вони також відносно стійкі до перенавчання і можуть добре узагальнювати нові дані.

Однак, одне з поширених питань, яке виникає при роботі з великими даними, чи є SVM відповідним алгоритмом для використання. Метод опорних векторів може бути обчислювально інтенсивними і погано масштабуватися до дуже великих наборів даних [8]. Також цей метод чутливий до вибору функції ядра та параметрів моделі, які може бути важко ефективно налаштувати на великих масивах даних. Це зазвичай призводить до надмірного або недостатнього тренування, що знижує точність моделі. Однак є кілька стратегій, які використовують, щоб зробити SVM більш ефективними при роботі з великими даними.

Одним з підходів є використання методів паралельної обробки для розподілу обчислень між декількома процесорами або вузлами в кластері. Це може значно скоротити час, необхідний для навчання SVM на великому наборі даних, але може вимагати спеціалізованого апаратного та програмного забезпечення.

Інший підхід полягає у використанні методів відбору ознак або зменшення розмірності для зменшення кількості ознак або змінних у наборі даних. Це може допомогти зменшити обчислювальну складність методу опорних векторів та покращити її прогностичні характеристики.

Для вирішення та уникнення даних проблем використовуються квантові комп’ютери.

Квантові обчислення пропонують потенціал для значного прискорення навчання методу опорних векторів [9], використовуючи унікальні властивості квантових систем, такі як суперпозиція та заплутаність. Зокрема, квантові обчислення можуть виконувати певні типи обчислень, які є експоненційно швидшими за класичні обчислення.

Однак важливо зазначити, що квантові обчислення все ще є новою технологією, і існує багато проблем, які необхідно вирішити, перш ніж її можна буде ефективно застосовувати для задач машинного навчання, таких як SVM. Наприклад, розробка квантового обладнання та програмного забезпечення все ще перебуває на ранніх стадіях, і існує багато технічних і практичних проблем, пов'язаних з побудовою та експлуатацією квантових комп'ютерів.

Підсумовуючи, можна сказати, що квантові обчислення мають потенціал значно покращити продуктивність методу опорних векторів при роботі з великими обсягами даних, використовуючи унікальні властивості квантових систем. Однак існує низка проблем, які необхідно вирішити, перш ніж квантові обчислення можна буде ефективно застосовувати для вирішення завдань машинного навчання.

1.2 Огляд сучасного стану квантових обчислень

Квантові обчислення та квантові комп’ютери - це новий тип обчислювальної технології, яка використовує закони квантової механіки для розв’язання складних обчислювальних задач [10]. Квантові комп’ютери обіцяють вирішити проблеми, які зараз вважаються нерозв’язними з використанням традиційних комп’ютерів.

Квантові обчислення відрізняються від класичних обчислень тим, що базуються на принципах квантової механіки. У класичній обчислювальній техніці для збереження та передачі інформації використовуються біти, які можуть приймати значення лише «0» або «1». Однак у квантовій інформатиці використовується квантовий біт або кубіт як головний елемент, який може перебувати у двох базових станах. Квантовий біт може бути в суперпозиції, що означає, що він може бути як «0», так і «1» одночасно [11].

Крім того, квантові біти можуть бути заплутаними, тобто взаємно пов'язаними таким чином, що зміна стану одного кубіту автоматично впливає на стан інших кубітів. На рівні апаратної реалізації квантових комп'ютерів немає потреби у процесорі чи пам'яті, оскільки кубіти є основною та єдиною одиницею в такій технології. Квантові кубіти здатні містити та обробляти значно більше інформації за короткий проміжок часу, що дає їм вищу потужність порівняно з традиційними бітами.

Такі особливості дозволяють квантовим комп’ютерам обробляти великі обсяги даних та розв'язувати складні задачі, які неможливо вирішити за допомогою традиційних комп'ютерів [12].

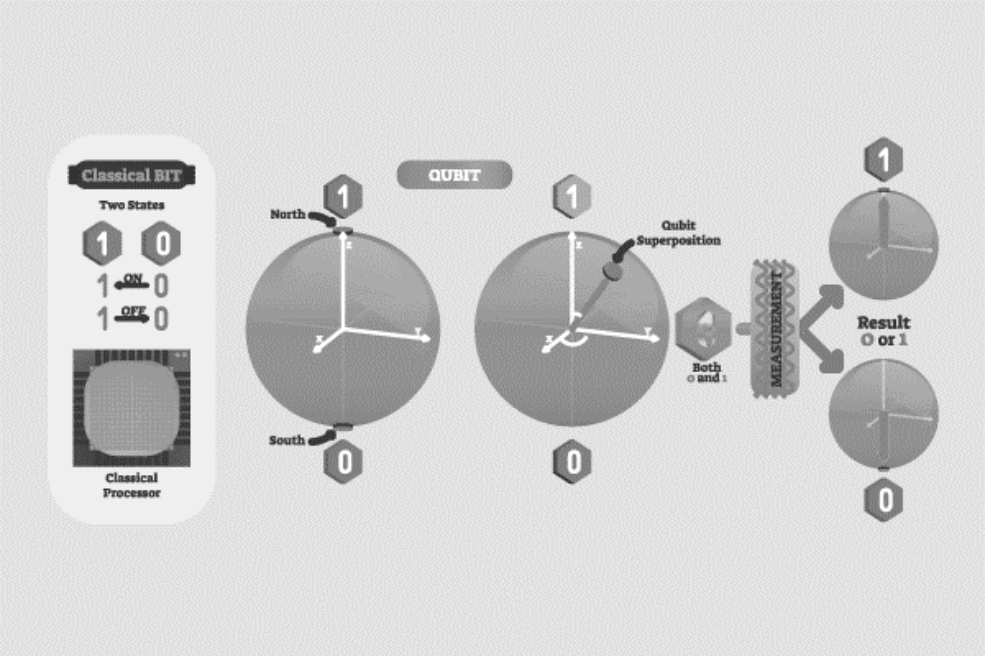


Рисунок 1.2 – Візуалізація принципу роботи квантового комп’ютера [13]

Квантова заплутаність - це фундаментальне явище квантової механіки, яке відрізняє квантові обчислення від класичних. Заплутаність означає, що квантові об'єкти можуть бути взаємно пов'язаними таким чином, що зміна стану одного об'єкту миттєво відображається на стані іншого об'єкту, незалежно від відстані між ними. Це може здатися надзвичайно неінтуїтивним, але саме це явище дозволяє квантовим комп'ютерам працювати з дуже великими об'ємами даних і розв'язувати складні задачі.

Квантова заплутаність [14] була вперше запропонована Ейнштейном, Подольським та Розеном в 1935 році. У той час вони сподівалися показати, що квантова механіка має неповність і вона не може пояснити всі фізичні явища. Проте згодом виявилося, що заплутаність дійсно існує і може бути використана для створення квантових комп'ютерів.

У класичній фізиці, коли два об'єкти взаємодіють між собою, вони можуть мати певний вплив один на одного, але взаємодія залежить від відстані між ними. У квантовій заплутаності цей вплив не залежить від відстані між об'єктами. Це означає, що квантові об'єкти можуть мати зв'язок, який неможливо повністю описати, використовуючи класичні поняття.

Квантова заплутаність має дивовижні властивості, які можуть використовуватися в різних областях науки та техніки. Однією з найбільш цікавих можливостей є використання заплутаних станів для підвищення точності вимірювань. Наприклад, якщо ми знаємо, що два квантових біти заплутані, то зміна стану одного з них негайно відобразиться на стані іншого. Таким чином, ми можемо вимірювати величини з вищою точністю, ніж у класичних системах.

Квантова заплутаність також використовується в квантових обчислювальних системах для розв’язування складних задач. Заплутаність дозволяє взаємодіяти з декількома квантовими бітами одночасно, що дозволяє квантовим комп’ютерам ефективно оброблювати великі об’єми даних та вирішувати задачі, які традиційні комп’ютери не в змозі обробити.

Квантова суперпозиція - це одна з ключових характеристик квантових систем, яка є основою квантових обчислень та квантової механіки загалом. Відмінність від класичної механіки полягає в тому, що в квантовій механіці, квантовий об'єкт може перебувати одночасно в кількох станах, у тому числі й у суперпозиції. У класичній фізиці об'єкти завжди знаходяться у певному стані, який можна точно виміряти. Наприклад, м'яч може знаходитися у стані "рухається вліво" або "рухається вправо", але не може бути в обох станах одночасно. Однак, в квантовій механіці, квантовий об'єкт може бути в суперпозиції кількох станів, тобто одночасно знаходитися у стані "0" та "1".

Експеримент з котом Шредінгера є одним з найбільш відомих ілюстрацій квантової механіки [15]. Цей експеримент був запропонований фізиком Ервіном Шредінгером у 1935 році з метою демонстрації концепції квантової суперпозиції. Ідея експерименту полягає в тому, що радіоактивне ядро, з імовірністю 50/50, може розпастися, випустивши отруйний газ, що вбиває кота. Існує певний момент у часі, коли не можна сказати, чи кіт живий чи мертвий, тобто кіт перебуває в квантовій суперпозиції, яка включає обидва стани - живий і мертвий.

Квантові обчислення здатні досягти значень, які в традиційних обчисленнях вважаються неможливими. В традиційній двійковій системі обчислення значення можуть бути або «1», або «0», і нічого іншого. Проте в квантових обчисленнях можна контролювати значення, які одночасно можуть мати стани як «1», так

і «0» з різними вагами. Наприклад, можливо контролювати значення, які складаються на 30% зі значення «0» та на 70% зі значення «1».

Ця можливість керування значеннями порушує традиційні правила обчислень, що дає можливість розробляти нові алгоритми, розв'язувати проблеми, які раніше вважалися нерозв'язними, і прискорювати процес обчислення.

Хоча квантові обчислення відкривають безліч нових можливостей, їх також відмінює низка недоліків, які потрібно враховувати при їх використанні [16].

Одним з таких недоліків є шум. Шум є одним з головних викликів, з якими стикаються квантові комп'ютери. Це виникає з-за квантових флуктуацій у системі, які можуть викликати помилки у результаті обчислень. Шум є невід'ємною частиною квантових систем і відрізняється від шуму, який ми звикли бачити в класичних системах.

Шум може призвести до неточності результатів та складнощів у виконанні задач. Це особливо важливо в обчислювальних задачах, де кожна помилка може вплинути на відповідь і порушити правильність результатів. Квантові комп'ютери потребують засобів для боротьби з шумом, які можуть включати в себе різноманітні техніки зменшення шуму та корекції помилок.

Щоб зменшити шум, квантові комп'ютери використовують технології, такі як кодування помилок та квантове декодування, які забезпечують корекцію помилок та збільшують точність результатів. Для цього використовуються техніки корекції помилок, такі як квантова корекція помилок (QEC) та динамічне знімання шуму (DDC) [17]. Квантова корекція помилок використовує квантові коди для виявлення та виправлення помилок, які можуть виникнути під час обчислень, що дозволяє досягти більш точних результатів. Динамічне знімання шуму, зі свого боку, застосовується для видалення шуму під час виконання квантових обчислень. Однак, окрім цього, можна зменшити вплив шуму на квантові обчислення за допомогою удосконалення квантових пристроїв та збільшення часу життя кубітів. Наприклад, збільшення часу життя кубітів може бути досягнуте шляхом зменшення взаємодії кубітів з оточуючим середовищем або застосування квантових

кодів для зменшення впливу помилок на квантові обчислення. Всі ці методи допомагають зменшити вплив шуму на квантові обчислення та покращити їх ефективність та точність.

Також одним з негативних факторів, що впливає на роботу квантових комп'ютерів, є температура [16]. Висока температура може призвести до збільшення шуму в системі і зменшення часу життя квантових бітів (кубітів). Квантові обчислення вимагають екстремально низьких температур, близьких до абсолютного нуля (-273,15 °C), щоб забезпечити стійкість і точність обчислень.

Теперішні квантові комп'ютери працюють при надзвичайно низьких температурах, що ускладнює їх розгортання та збільшує витрати на енергію. Також, зберігання квантових систем при таких температурах вимагає спеціальних умов і обладнання, що також збільшує вартість та складність побудови квантових комп'ютерів. Хоча висока температура не є безпосередньою загрозою для квантових комп'ютерів, вона може суттєво підірвати їх ефективність і точність обчислень. Тому, дослідники продовжують шукати способи зменшення впливу температури на квантові обчислення, такі як розробка кращих і більш ефективних квантових систем з меншою тепловою вразливістю.

1.3 Технічні, природничі та соціально-економічних інформаційні системи, що потребують застосування методів класифікації та мають великі об’єми даних

Квантові обчислення відкривають нову перспективу для розв’язання складних завдань у багатьох сферах науки, технологій та соціально-економічного розвитку. Завдяки набагато швидшим та потужнішим обчислювальним можливостям, які квантові комп'ютери надають, їх можна використовувати для обробки та аналізу великих об'ємів даних у таких сферах, як наука про дані, медицина, фінанси, квантова хімія, чиста енергетика, безпечні комунікації та багато інших. Передбачається, що ці нові можливості забезпечать секторам, що мають великі та

складні обчислювальні задачі, можливість вирішувати їх швидше та ефективніше, ніж за допомогою традиційних класичних обчислювальних систем.

Технічні, природничі та соціально-економічні інформаційні системи зазвичай мають великі обсяги даних, які потребують застосування ефективних методів класифікації. Проте, збільшення обсягів даних також може викликати проблеми з обробкою та збереженням цих даних. Одним із способів розв'язання цієї проблеми є застосування технологій штучного інтелекту, таких як машинне навчання та глибинне навчання, що дозволяють автоматизувати процеси обробки та аналізу великих обсягів даних.

Наприклад, в технічних системах можуть бути великі масиви даних про роботу машин та обладнання, які потрібно аналізувати, щоб виявляти проблеми та знижувати витрати на обслуговування [18]. Для цього використовуються різноманітні прилади, сенсори та датчики, що збирають дані про роботу обладнання. Далі ці дані оброблюються та аналізуються за допомогою спеціальних програмних засобів, які можуть використовувати методи класифікації для виявлення аномалій та проблем, що можуть призвести до збоїв в роботі системи. Застосування QSVM для аналізу даних в технічних системах потенційно допоможе виявити відхилення від заданих параметрів та прогнозувати можливі проблеми у виробництві. Крім того, використання квантових методів сприяє зменшенню витрат на обслуговування технічних систем та збільшення їх надійності.

В природничих науках, таких як біологія та екологія, також можуть бути значні обсяги даних, що потребують класифікації для здійснення наукових досліджень та вивчення природних процесів. Застосування QSVM в природничих науках може бути корисним для класифікації даних та дослідження природних процесів. Наприклад, у біології можна використовувати QSVM для аналізу генетичних даних та класифікації біологічних видів на основі їхніх генетичних характеристик [19]. У екології QSVM можна використовувати для класифікації типів екосистем та вивчення взаємодії різних видів у природному середовищі.

Наприклад вчені з університету Сінгапуру використали QSVM для класифікації різних типів екосистем на основі даних, отриманих з супутників, що може

допомогти у вивченні змін клімату та екологічних процесів [20]. Загалом застосування QSVM в природничих науках може сприяти у покращенні точності класифікації та підвищенні ефективності досліджень, що призведе до нових відкриттів та знань у науці.

Також квантові алгоритми можуть бути застосовані в соціально-економічних системах для аналізу великих обсягів даних про споживачів, їх поведінку та інші фактори, що можуть впливати на бізнес-рішення. Одним із прикладів застосування квантових алгоритмів в цій сфері є класифікація даних, що дозволяє розробляти стратегії для підвищення ефективності бізнесу та задоволення потреб клієнтів. Квантові алгоритми здатні виявляти складні залежності та взаємозв'язки між різними параметрами та факторами, що дозволяє розробляти більш точні та ефективні стратегії. Крім того, за допомогою квантових алгоритмів можна здійснювати швидкий та ефективний аналіз великих обсягів даних, що є особливо важливим для бізнес-аналітики та прийняття рішень на основі даних. Квантовий алгоритм SVM може бути використаний для покращення різних аспектів управління ризиками в фінансових установах [21]. Наприклад, він може допомогти відповісти на питання, пов'язані з вибором інвестиційних портфелів та розподілом активів, що максимізують дохід та мінімізують ризик.

Крім того, він може бути використаний для прогнозування фінансових криз, що дозволить установам попереджати можливі проблеми та приймати вчасні рішення. Квантовий SVM також може допомогти управляти кредитним ризиком, знаходячи оптимальний баланс між ризиком та доходом. Всі ці застосування дозволяють фінансовим установам зменшити ризики та максимізувати прибуток, забезпечуючи більш стійке та ефективне функціонування на ринку.

Також квантові обчислення можуть вплинути на криптографічні технології, які є основою для більшості криптовалют [22]. Квантові обчислення матимуть значний вплив на криптографію, оскільки алгоритми Шора, які використовуються на квантових комп'ютерах, здатні розкладати складні числа на прості множники. Це значно зменшить ефективність криптографічних алгоритмів, які використовуються в сучасних криптографічних протоколах.

Однак, з'являються нові криптографічні протоколи, які використовують квантові властивості для створення більш ефективних та надійних систем шифрування. Один з прикладів цього - квантовий ключовий договір (quantum key distribution), який використовує квантову механіку для забезпечення безпеки при обміні ключами.

Крім того, QSVM може бути використаний для аналізу криптографічних протоколів та виявлення потенційних вразливостей, що може допомогти у забезпеченні безпеки криптографічних систем. Застосування QSVM до цих задач може допомогти виявляти складні зв'язки між різними частинами криптографічного протоколу та прогнозувати можливі атаки на систему шифрування.

Отже, квантові обчислення можуть бути як загрозою для криптографії, так і новим інструментом для створення більш ефективних та безпечних систем шифрування.

1.4 Мета та завдання дослідження

Мета кваліфікаційної роботи полягає в дослідженні особливостей використання квантового методу опорних векторів (з варіаційними алгоритмами) для реалізації класифікації даних у технічних, природничих і соціально-економічних системах з великим об’ємом даних.

Для досягнення поставленої мети визначені наступні завдання дослідження:

- огляд методів та засобів машинного навчання;
- огляд квантових обчислень для великої кількості даних;
- огляд технічних, природничих та соціально-економічних інформаційних систем, що потребують класифікації та мають великі об’єми даних;
- запропонувати способи реалізації квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем;
- провести реалізацію квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем.

Розділ 2 Особливості реалізації квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем

2.1. Класичний метод опорних векторів

Метод опорних векторів (SVM) - це алгоритм навчання з учителем, що використовується для класифікації, регресії та виявлення викидів [8]. Основна ідея SVM базується на концепції гіперплощини максимальної ширини (ММШ), яка є лінійною границею рішень і відокремлює дані на різні класи з максимальною шириною між границею рішень і найближчими точками даних. Завдяки трюку ядра, який дозволяє відображати вхідні дані в більш багатовимірний простір, SVM може вирішувати як лінійні, так і нелінійні задачі класифікації.

Основна ідея лінійного класифікатора полягає в тому, що ознаковий простір можна розділити гіперплощиною на два напівпростори, у кожному з яких прогнозується одне з двох значень цільового класу. Якщо це можна зробити без помилок, то навчальна вибірка називається лінійно роздільною.

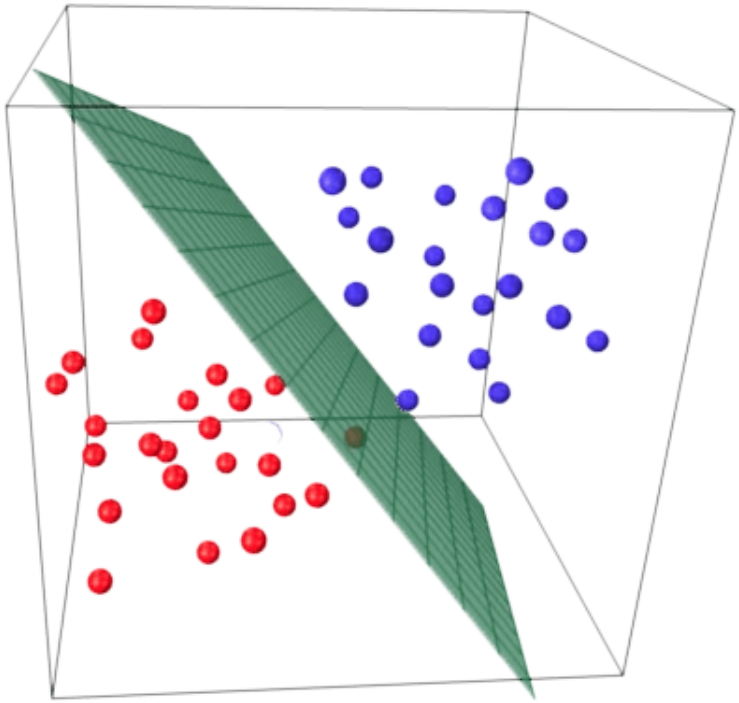


Рисунок 2.1 – Лінійна класифікація [23]

Розглянемо задачу бінарної класифікації, в якій $X = \mathbb{R}^d$ - простір об'єктів, $Y = \{-1, +1\}$ - множина допустимих відповідей (цільова ознака), X - навчальна вибірка.

.

$$X = \{(x_i, y_i)\}_{i=1}^{l} \tag{2.1}$$

Будемо клас $+1$ називати позитивним, а клас -1 - негативним. Тут d - розмірність ознакового простору, l - кількість прикладів у навчальній вибірці.

Лінійна модель класифікації визначається таким чином:

$$a(x) = sign(\langle w, x\rangle + b) = sign\left(\sum_{j=1}^{d} w_j\, x_j + b\right), \tag{2.2}$$

де $w \in \mathbb{R}^d -$ вектор вагів, $b \in \mathbb{R}$ - зсув, $sign(*)$ - функція "сигнум", що повертає знак свого аргументу, $a(x)$ - відповідь класифікатора на прикладі x.

Часто вважають, що серед ознак є константа, $x_{d+1} = 1$. У цьому випадку немає необхідності вводити зсув b, і лінійний класифікатор можна задавати як:

$$a(x) = sign\langle w, x\rangle. \tag{2.3}$$

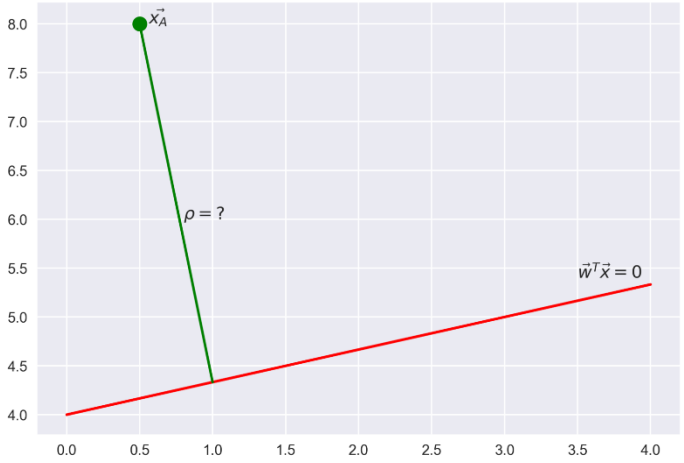


Рисунок 2.2 – Відстань від точки до площини [23]

Геометрично лінійний класифікатор відповідає гіперплощині з вектором нормалі w, яка задається рівнянням $\langle w, x\rangle = 0$. Величина скалярного добутку $\langle w, x\rangle$ пропорційна відстані від гіперплощини до точки x, а його знак показує, з якого боку від гіперплощини розташована ця точка. Якщо бути точним, відстань від точки з радіус-вектором x_A до площини $\langle w, x\rangle = 0$:

$$p(x_A, \langle w, x\rangle = 0) = \frac{\langle w, x\rangle}{||w||} \tag{2.4}$$

Таким чином, лінійний класифікатор розділяє простір на дві частини за допомогою гіперплощини, і при цьому один напівпростір відносить до позитивного класу, а інший - до негативного.

Також варто розглянути вираз $M(x_i, y_i, w) = y_i \cdot \langle w, x\rangle$.

Це відступ класифікації (margin) для об'єкта навчальної вибірки (x_i, y_i). На жаль, його дуже легко переплутати із зазором класифікації, який з'явиться трохи нижче у викладі інтуїції методу опорних векторів. Щоб не плутати: відступ визначено на конкретному об'єкті навчальної вибірки.

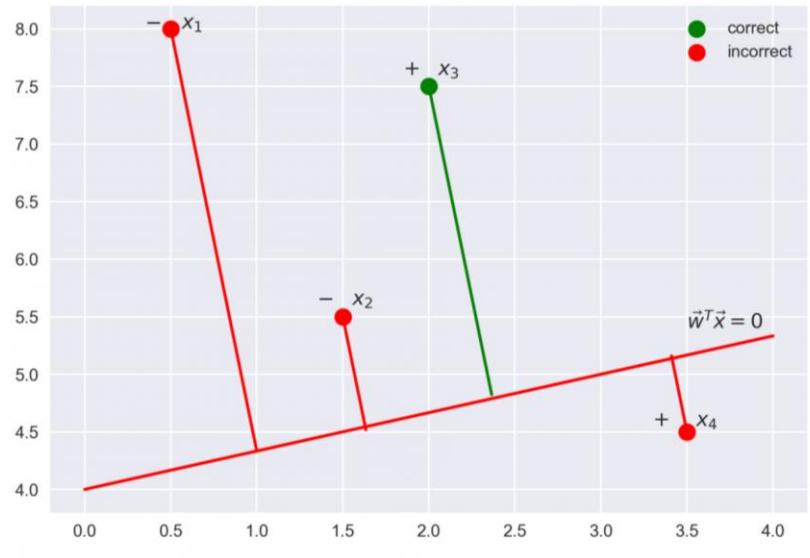


Рисунок 2.3 – Ілюстрація для поняття відступу класифікації [23]

Відступ - це свого роду "впевненість" моделі в класифікації об'єкта [8] (x_i, y_i):

– якщо відступ великий (за модулем) і позитивний, це означає, що мітку класу поставлено правильно, а об'єкт перебуває далеко від гіперплощини, що розділяє (такий об'єкт класифікують упевнено). На рисунку 2.3 - x_3;

– якщо відступ великий (за модулем) і від'ємний, отже, мітку класу поставлено неправильно, а об'єкт перебуває далеко від гіперплощини, що розділяє (найімовірніше такий об'єкт - аномалія, наприклад, його мітку в навчальній вибірці поставлено неправильно). На рисунку 2.3 - x_1;

– якщо відступ малий (за модулем), то об'єкт перебуває близько до гіперплощини, що розділяє, а знак відступу визначає, чи правильно об'єкт класифіковано. На рисунку 2.3 - x_2 і x_4.

Поняття відступу класифікації - частина функції втрат, яка оптимізується в методі опорних векторів.

Задачу методу опорних векторів можна звести до задачі безумовної оптимізації функціоналу, який має вигляд верхньої оцінки на частку неправильних відповідей. Якщо подати цю задачу за допомогою такої умови:

$$\begin{cases} \xi_i \geq 1 - y_i(\langle w, x_i \rangle + b) \\ \xi_i \geq 0 \end{cases} \tag{2.5}$$

Оскільки при цьому у функціоналі потрібно, щоб штрафи ξ_i були якомога меншими, то можна отримати таку явну формулу для них:

$$\xi_i = \max\big(0, 1 - y_i(\langle w, x_i \rangle + b)\big). \tag{2.6}$$

Цей вираз для ξ уже враховує в собі всі обмеження задачі, описаної вище. Отже, виходячи з цього ми отримаємо безумовну задачу оптимізації:

$$\frac{1}{2}\,||w||^2 + C\sum_{i=1}^{l}\max\big(0, 1 - y_i(\langle w, x_i\rangle + b)\big) \to \min_{w,b} \qquad (2.7)$$

Це завдання є негладким, тому розв'язувати його може бути досить важко. Проте вона показує, що метод опорних векторів, по суті, як і логістична регресія, будує верхню оцінку на частку помилок і додає до неї стандартну квадратичну регуляризацію. Ця функція втрат називається кусочно-лінійної (hinge-loss).

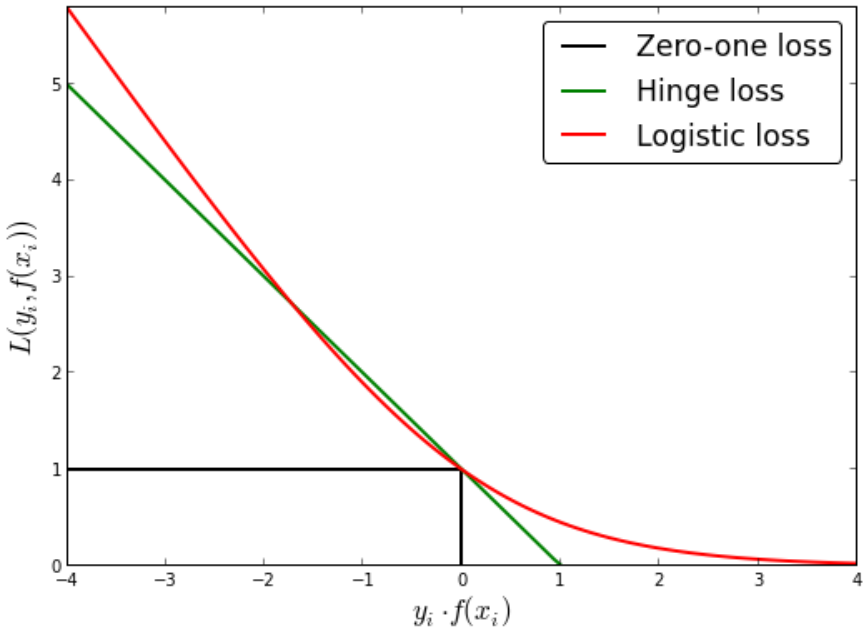


Рисунок 2.4 – Порогова, кусочно-лінійна та логістична функції втрат [24]

В ідеалі ми хотіли б штрафувати класифікатор за помилку на прикладі: $L_{1/0}\,(M_i) = [M_i < 0]$. Це порогова функція втрат (zero-one loss), її графік зображений чорним на рисунку 2.4. Безпосередньо ми не можемо ефективно оптимізувати таку функцію градієнтними методами через розрив у нулі, тому оптимізується верхня оцінка zero-one loss. У разі логістичної регресії - логістична функція втрат $L(M_i) = \log(1 + e^{M_i})$ її графік зображений червоним на рисунку 2.4. У випадку методу опорних векторів - кусочно-лінійна функція $L(M_i) = \max(0, 1 - M_i)$ її графік зображений зеленим на рисунку 2.4.

Алгоритм SVM, використовуваний для лінійної класифікації, здатний до жорсткої та м'якої розділки даних. При наявності чіткої роздільної границі, жорстке розділення є оптимальним вибором, оскільки воно забезпечує найбільшу точність класифікації. У випадках, коли дані не мають чіткої роздільної границі, м'яке розділення може забезпечити більш гнучку модель. М'яке розділення дозволяє приймати деякі помилки в класифікації, що дозволяє досягнути більшої точності на складних даних.

У методі опорних векторів для розв'язання нелінійних задач класифікації необхідне використання ядер [25]. Ядро представляє собою функцію, яка визначається наступним способом:

$$K(x, y) = \big(\phi(x), \phi(y)\big), \tag{2.8}$$

де $\phi(x) -$ вектор вагів, перетворення, яке здійснює перехід з простору $\mathbb{R}^d$ в простір H, x, y – вектори.

Ядра, які використовуються найчастіше [26]:

– Лінійне $K(x, y) = (x, y)^d$;

– Поліноміальне ядро $K(x, y) = (\langle x, y\rangle + c)^d$, визначене для степеня ядра d і параметра нормалізації c;

– Гаусове ядро, також відоме як RBF (radial-basis functions) $K(x, y) = \exp(-\gamma\big||x - y|\big|^2)$ з параметром ядра γ який відповідає за ступінь кривизни.

2.2. Квантовий метод опорних векторів

Квантовий варіаційний класифікатор (QVC) - це квантовий алгоритм машинного навчання, який використовується для класифікації даних на основі квантових обчислень [27]. Цей алгоритм використовується для розв'язання задач, пов'язаних з машинним навчанням, таких як класифікація зображень, відео, аудіо, текстових даних та інших видів даних. Існує дві версії QSVM, які надають окремо квадратичне та експоненціальне прискорення. Перша зосереджена на розв'язанні

неопуклих оптимізаційних задач, використовуючи алгоритм Гровера як підпрограму. Друга спрямована на аналіз апроксимації SVM методом найменших квадратів.

Основними компонентами QVC є квантові біти та квантові гейти, які використовуються для здійснення певних операцій з даними.

Подібно до класичної бітової інформації, квантова бітова інформація позначається як кубіт і зазвичай представляється за допомогою нотації Дірака:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \tag{2.9}$$

та

$$|1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{2.10}$$

Розглядаючи квантову критерію суперпозиції:

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \tag{2.11}$$

де $\alpha, \beta \in \mathbb{C}$ і $|\alpha|^2 + |\beta|^2 = 1$; З цього рівняння слідує що $|\varphi\rangle$ може перебувати як у стані $|0\rangle$ так і у стані $|1\rangle$ з ймовірністю $|\alpha|^2$ і $|\beta|^2$, відповідно.

Квантові гейти є основними будівельними блоками для будь-яких квантових схем, зокрема й тих, що застосовуються для машинного навчання. Можна сказати, що це своєрідний алфавіт квантових обчислень. Він необхідний, щоб відразу розуміти, наприклад, що зображено на подібних схемах:

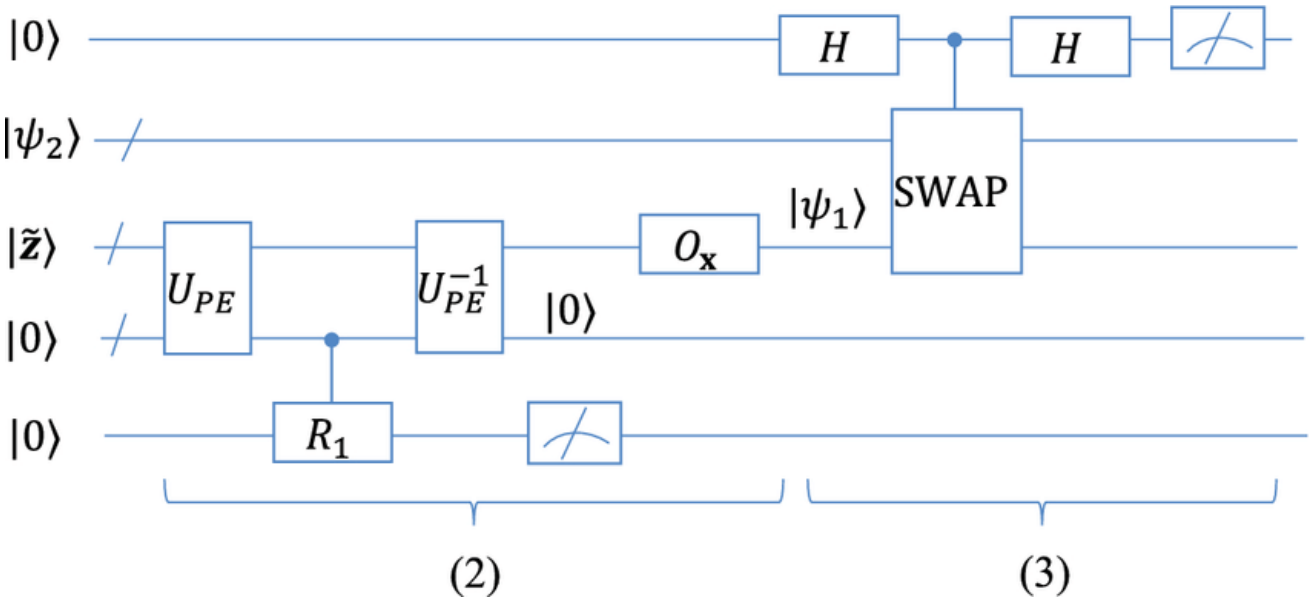


Рисунок 2.5 – Приклад схеми QSVM [28]

До елементарних однокубітних квантових гейтів належать Полі-X (Pauli-X), Полі-Y (Pauli-Y), Полі-Z (Pauli-Z), Гадамар (Hadamard) та Зсув фази (Phase Shift) [29]. Нижче приведені формули для цих квантових гейтів:

$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$: він перемикає $|0\rangle$ на $|1\rangle$ і $|1\rangle$ на $|0\rangle$;

$Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$: він перемикає $|0\rangle$ на $i|1\rangle$ і $|1\rangle$ на $-i|0\rangle$;

$z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$: він залишає $|0\rangle$ незмінним і перемикає $|1\rangle$ на $-|1\rangle$;

$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$: він створює суперпозицію двох станів, та перемикає $|0\rangle$ на $\frac{|0\rangle + |1\rangle}{\sqrt{2}}$ та перемикає $|1\rangle$ на $\frac{|0\rangle - |1\rangle}{\sqrt{2}}$;

$\widehat{U_1} = \begin{pmatrix} 1 & 0 \\ 0 & e^{-\iota\sigma} \end{pmatrix}$: він залишає $|0\rangle$ незмінним і перемикає $|1\rangle$ на $e^{-\iota\sigma}|1\rangle$;

Також є поворотні гейти, які відіграють центральну роль у квантовому машинному навчанні. Ось який вигляд мають наші однокубітні стани на сфері Блоха:

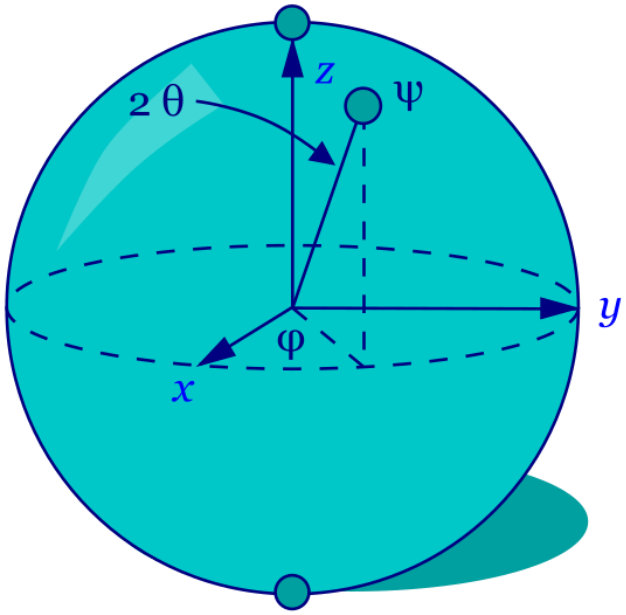


Рисунок 2.6 – Сфера Блоха [29]

Будь-який однокубітний гейт можна представити як обертання вектора стану $|\psi\rangle$ на деякий кут навколо деякої осі, що проходить через центр сфери Блоха.

Гейти $\widehat{RX}(\phi)$, $\widehat{RY}(\phi)$, $\widehat{RZ}(\phi)$ здійснюють поворот на певний кут ϕ навколо відповідної осі на сфері Блоха. Нижче приведені формули для цих квантових гейтів:

$$\widehat{RX}(\phi) = \begin{bmatrix} COS\left(\frac{\phi}{2}\right) & -i\sin\left(\frac{\phi}{2}\right) \\ -i\sin\left(\frac{\phi}{2}\right) & COS\left(\frac{\phi}{2}\right) \end{bmatrix}, \tag{2.12}$$

$$\widehat{RY}(\phi) = \begin{bmatrix} COS\left(\frac{\phi}{2}\right) & -\sin\left(\frac{\phi}{2}\right) \\ \sin\left(\frac{\phi}{2}\right) & COS\left(\frac{\phi}{2}\right) \end{bmatrix}, \tag{2.13}$$

$$\widehat{RZ}(\phi) = \begin{bmatrix} e^{-\frac{i\phi}{2}} & 0 \\ 0 & e^{\frac{i\phi}{2}} \end{bmatrix}, \tag{2.14}$$

Для двокубітної конфігурації гейтів використовуються Controlled NOT (CNOT) та SWAP.

Квантовий гейт контрольованого інвертування (Controlled Not) - це гейт, який діє на два кубіти: робочий і контрольний. Залежно від того, чи має контрольний кубіт значення 1 або 0, цей гейт інвертує або не інвертує робочий кубіт.

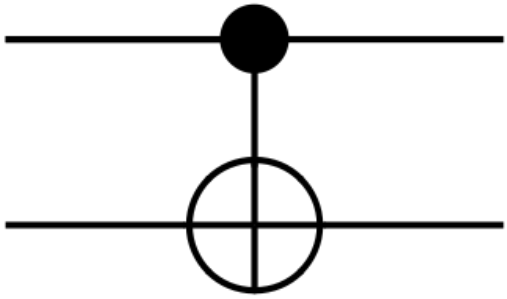


Рисунок 2.7 – Схематичне позначення гейту CNOT [30]

$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$: він використовується для 2 кубітів і виконує операцію NOT, коли керуючий кубіт дорівнює $|1\rangle$. Це призводить до перемикання між двома останніми рядками матриці тотожності;

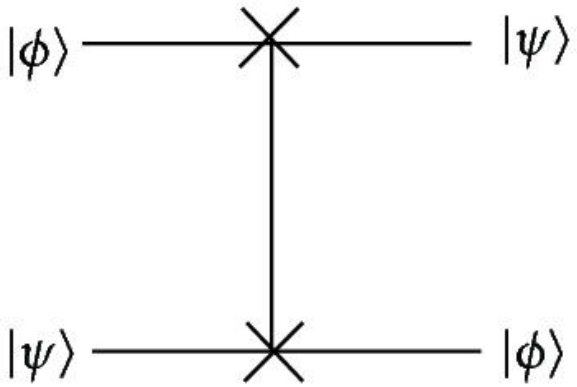


Рисунок 2.8 – Схематичне позначення гейту SWAP [30]

$SWAP = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$: призводить до переходу з рядка 2 до рядка 3 у матриці тотожності.

Квантова варіаційна схема - це квантова схема, яка може налаштовуватись за допомогою параметричних вентилів, щоб досягти оптимальних результатів. Для здійснення класифікації спочатку потрібно розробити варіаційний алгоритм, який може використовувати Гільбертів простір квантового процесора, щоб знайти оптимальну роздільну гіперплощину, як це робить метод опорних векторів. Алгоритм складається з двох основних частин: етапу навчання та етапу тестування. На етапі навчання алгоритм тренується на наборі даних з мітками. На етапі тестування оптимізовану квантову схему класифікації запускають на іншому наборі точок даних без введення міток. Після того, як ми отримали коефіцієнт успіху для набору даних, можна приступити до доповнення процесу навчання та тестування. Одним із способів покращення точності класифікатора є використання карти характеристик. Ця процедура дозволяє перетворити класичні значення на квантовий стан. Після цього можна використати варіаційну оптимізацію, яка визначає оптимальні значення параметрів квантової схеми. На завершення, проводиться вимірювання, яке дозволяє зчитати кінцевий результат класифікації. Важливо зазначити, що точність класифікації залежить від якості карт характеристик та варіаційної оптимізації. Тому важливо здійснювати їх досконале налаштування та оптимізацію для отримання максимально точного результату. Квантова схема за рисунком 2.9

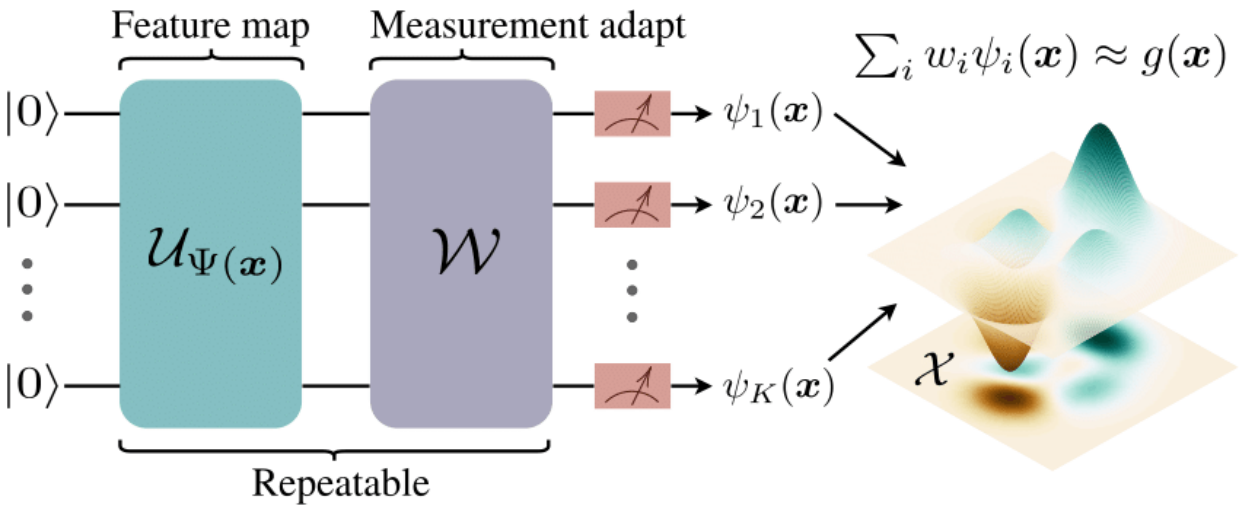


Рисунок 2.9 – Квантова схема варіаційного класифікатора [27]

Квантова оцінка ядра - це метод, який використовується для розв'язання задачі класифікації шляхом використання квантових обчислень. Задача полягає в знаходженні границі прийняття рішень на основі відомих даних. Ядро в цьому випадку відображає взаємодію між даними, які використовуються для розв'язання задачі класифікації.

Квантова оцінка ядра використовує квантові вимірювання, щоб знайти відстань між векторами даних в квантовому просторі, яка може бути використана для розрахунку ядра. Цей метод дозволяє знаходити коефіцієнти ядра, які можуть бути використані для класифікації нових даних.

Для обмеження розгляду в цій частині ми будемо використовувати двійкову мітку, а саме множину C, що складається з елементів "+1" та "-1". Також застосовується квантовий комп'ютер для оцінки ядра в методі опорних векторів.

Ядро обраховується наступним чином:

$$K(x, y) = |\langle \Phi(x) | \Phi(y) \rangle|^2, \tag{2.15}$$

де $\Phi(x)$ - функція, яка переводить з одного простору в інший; x , y – вектори.

Схема оцінки ядра на рисунку 2.10

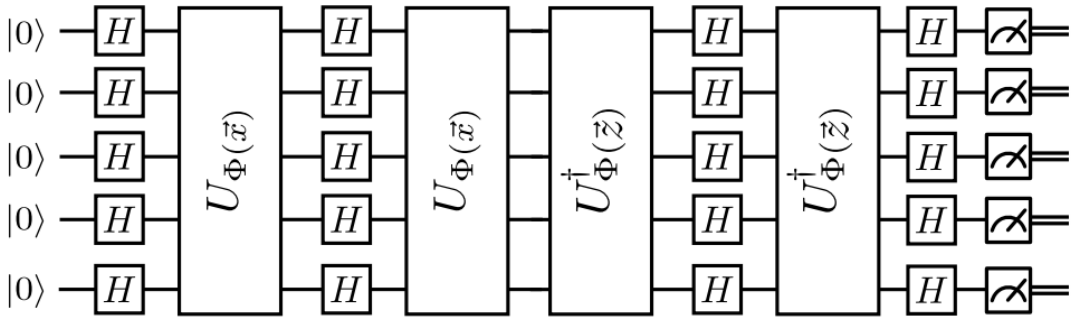


Рисунок 2.10 – Схема оцінки квантового ядра [31]

Після створення ядра, ми можемо використовувати звичайний класичний метод опорних векторів. Щоб визначити мітку для іншої точки, потрібно обчислити ядро на квантовому комп'ютері і використати результат обчислення для присвоєння нової мітки.

2.3 Оцінка якості класифікації та обробка даних

Для вимірювання ефективності класифікації розглянемо розбиття даних X, заданих у рівнянні (2.1), на дві незв'язні підмножини $X^{(train)}$, $X^{(test)}$. Навчальні дані $X^{(train)}$ використовуються для навчання як класичного SVM, так і для квантової варіації цього методу. В обох випадках результатом навчання є набір коефіцієнтів $\{a_n\}$, які можуть бути використані для прогнозування класів за допомогою функції прийняття рішень, яка представлена у рівнянні (2.2). Після цього класифікатор оцінюється на тестових даних $X^{(test)}$ порівнюючи передбачення класів $\widetilde{y_n} = sign(f(x_n))$ зі справжньою міткою y_n для кожного набору $(x_n, y_n) \in X^{(test)}$.

Простим методом оцінки ефективності класифікатора є підрахунок кількості правильних прогнозів, тобто кількості правильно визначених істинно-позитивних результатів (True Positives) для кожного $\widetilde{y_n} = y_n = 1$. Для отримання точності класифікації, необхідно розділити кількість істинно-позитивних результатів (True Positives) на загальну кількість точок у тестових даних - $|X^{(test)}|$. Іншими словами, точність класифікації (accuracy) - це відношення кількості правильно класифікованих точок до загальної кількості точок у тестових даних. Бінарні задачі класифікації часто вимагають більш точної метрики ефективності, ніж проста точність (accuracy) [32], тому що, висока точність не завжди гарантує, що класифікатор є ефективним. Якщо взяти простий приклад датасету, де 80% даних належать до класу негативних, то простий класифікатор, який завжди повертає -1, буде мати точність 80%. Проте цей класифікатор, практично не корисний. У таких випадках нас більше цікавить здатність відрізняти позитивні зразки від негативних, особливо якщо датасет містить багато негативних зразків.

Щоб отримати більш стійку метрику ефективності, ми спочатку визначаємо кількість усіх можливих випадків, які можуть виникнути при передбаченні класу $\widetilde{y_n} = sign(f(x_n))$, кількість істинно-позитивних результатів, де $\widetilde{y_n} = y_n = 1$, кількість хибно-позитивних результатів (False Positives), де $\widetilde{y_n} = 1$, але $y_n = -1$, кількість істинно-негативних результатів (True Negatives), де $\widetilde{y_n} = y_n = -1$, і кількість хибно-негативних результатів (False Negatives), де $\widetilde{y_n} = -1$, але $y_n = 1$ (Варто зазначити, що сума кількості істинно-позитивних, хибно-позитивних, істинно-негативних та хибно-негативних результатів дорівнює загальній кількості точок тестових даних $|X^{(test)}|$).

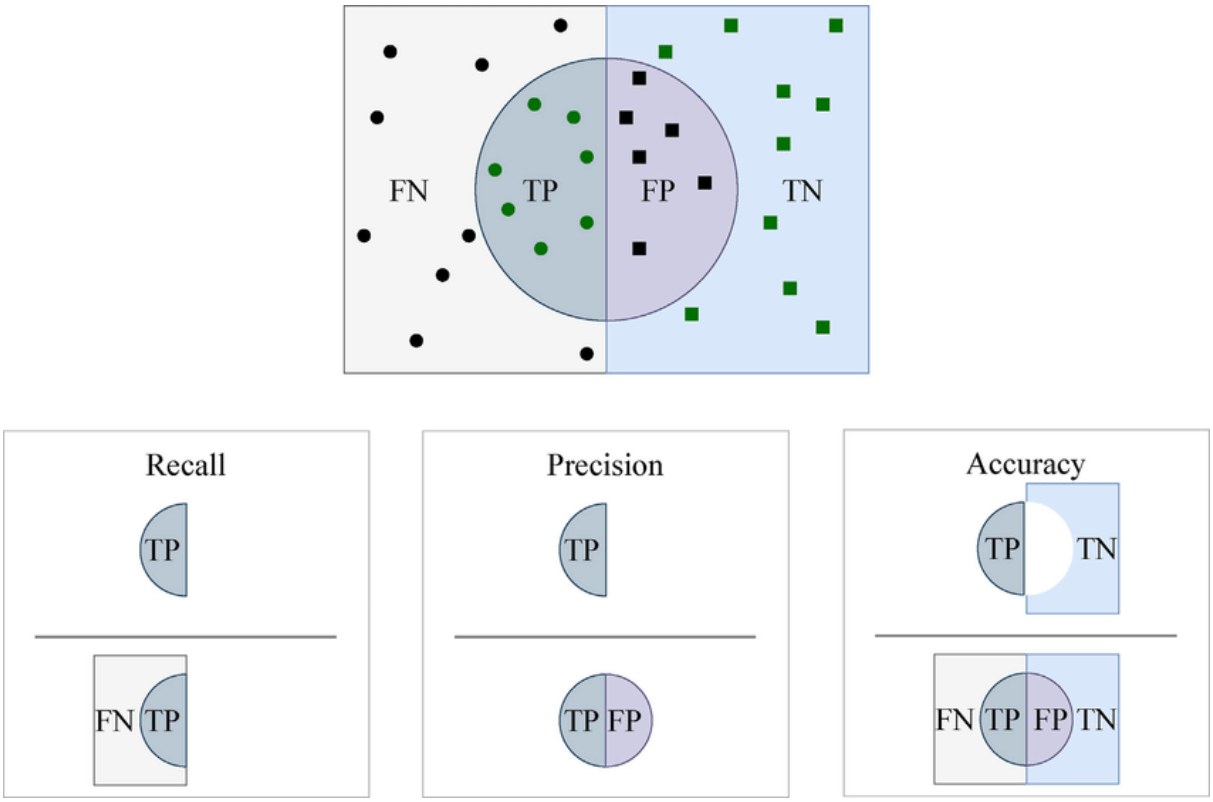


Рисунок 2.11 – Ілюстрація розрахунку метрик оцінки якості [33]

З використанням вищезгаданих кількостей можна обчислити такі метрики, як чутливість або повноту (recall)

$$Recall = \frac{TP}{TP + FN} \tag{2.16}$$

та влучність (precision)

$$Precision = \frac{TP}{TP + FP} \tag{2.17}$$

У наступних застосуваннях ми використовуємо метрики точність, повноту та влучність для порівняння класифікаторів та вимірювання їх продуктивності.

Для зменшення розмірності даних застосуємо метод головних компонент (PCA). PCA - це метод зменшення розмірності даних, який зазвичай використовується для обробки великих наборів даних [34]. Він дозволяє зменшити кількість змінних, що входять до набору даних, при цьому зберігаючи більшість інформації вихідного набору. Зменшення кількості змінних дозволяє спростити аналіз та візуалізацію даних, а також зменшити час, необхідний для обробки даних машинними алгоритмами навчання.

Приклад роботи методу головних компонент зображений на рисунку:

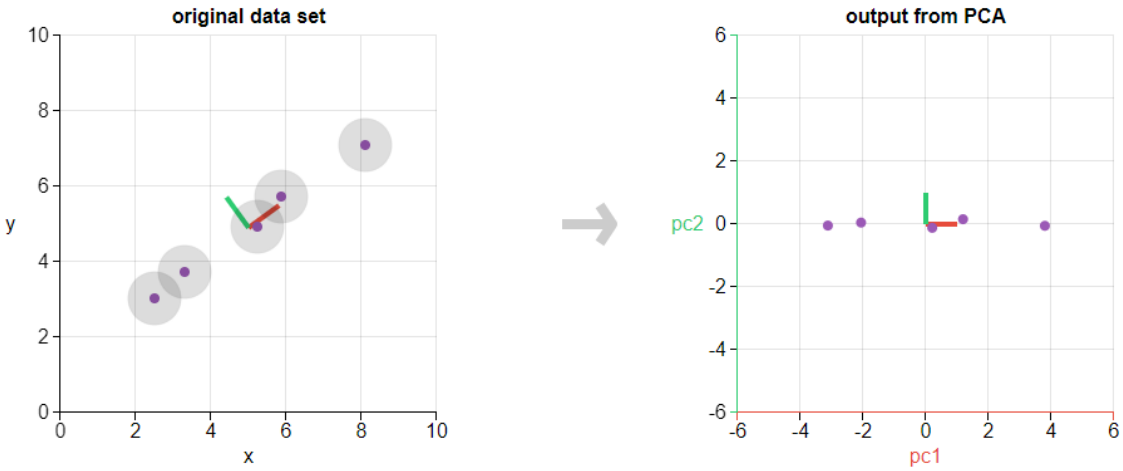


Рисунок 2.12 – Ілюстрація роботи методу головних компонент [35]

Давайте розглянемо математичне формулювання цього процесу:

Щоб зменшити розмірність наших даних з n до k з кількістю вимірів $k \leq n$, ми сортуємо наш список осей в порядку спадання дисперсії та беремо перші-k

з них. Ми починаємо з обчислення дисперсії та коваріації початкових ознак. Зазвичай це робиться за допомогою матриці коваріації. Згідно з визначенням коваріації, коваріація двох ознак обчислюється наступним чином:

$$cov(X_i, X_j) = E[(X_i - \mu_i)(X_j - \mu_j)] = E[X_i, X_j] - \mu_i \mu_j \quad (2.18)$$

де μ_i є математичним сподіванням i-ї ознаки. Варто зауважити, що коваріація симетрична, а коваріація вектора з самим собою дорівнює його дисперсії. Не діагональні значення є коваріаціями відповідної пари ознак. У термінах матриць, де X є матрицею спостережень, матриця коваріації виглядає наступним чином:

$$\Sigma = E[(X - E[X])(X - E[X])^T \quad (2.19)$$

Коваріаційну матрицю для нашої вибірки X можна представити як добуток $X^T X$. За допомогою відношення Релея можна довести, що максимальна варіація нашого набору даних досягається вздовж власного вектора цієї матриці, який відповідає максимальному власному значенню. Таким чином, головні компоненти, на які ми хочемо спроектувати наші дані, є просто власними векторами відповідних топ-k власних значень цієї матриці. Далі процес надзвичайно простий - просто потрібно помножити нашу матрицю даних на ці головні компоненти, і ми отримаємо проекцію наших даних в ортогональному базисі цих компонент. Тепер, якщо ми транспонуємо нашу матрицю даних і матрицю векторів головних компонент, ми відновимо початкову вибірку в тому просторі, з якого ми робили проекцію на компоненти. Якщо кількість компонент була меншою, ніж розмірність вихідного простору, ми втратимо частину інформації при такому перетворенні.

Також для максимально ефективного навчання нашого класифікатору, ми застосуємо метод перехресної перевірки. Навчання параметрів функції прогнозу та тестування її на тих же даних є методологічною помилкою: модель, яка просто повторює мітки зразків, які вона тільки що бачила, матиме ідеальний результат, але не зможе передбачити нічого корисного на ще невидимих даних. Ця ситуація

називається перенавчанням (overfitting) [36]. Щоб уникнути перенавчання моделі, досить поширеною практикою при проведенні машинного навчання на основі навчальних даних є виділення частини доступних даних як тестовий набір X^{Test}, y^{Test}. Важливо зазначити, що тут слово "експеримент" не обов'язково відноситься до наукового дослідження, оскільки навіть у комерційному середовищі машинне навчання зазвичай починається з експериментальних досліджень. Нижче наведено блок-схему типового процесу перехресної перевірки при тренуванні моделі.

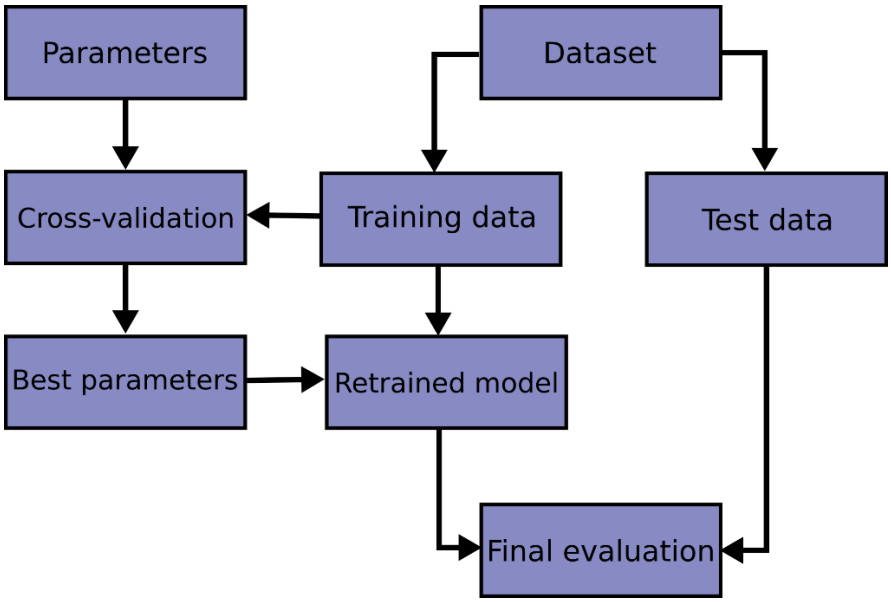


Рисунок 2.13 – Блок-схема типового процесу перехресної перевірки [37]

При оцінці різних налаштувань ("гіперпараметрів") для оцінювачів, наприклад, параметра C, який потрібно задати вручну для SVM, існує ризик перенавчання на тестовому наборі даних. У такому випадку знання про тестовий набір даних може "витікати" в модель, і метрики оцінки не відображатимуть загальної продуктивності. Для вирішення цієї проблеми можна виділити ще одну частину набору даних як "набір даних для перевірки": спочатку проводиться навчання на навчальному наборі, потім оцінка проводиться на наборі даних для перевірки, і коли експеримент виявляється успішним, остаточна оцінка може бути зроблена на тестовому наборі. Проте, розділяючи доступні дані на три набори, ми значно зменшуємо кількість вибірок, які можна використовувати для навчання моделі, і

результати можуть залежати від конкретного випадкового вибору пари (навчання, перевірка) наборів.

Рішенням цієї проблеми є процедура, яку називають перехресною перевіркою (cross-validation) [38]. Тестовий набір даних все ще повинен бути збережений для остаточної оцінки, але набір даних для перевірки більше не потрібен під час перехресної перевірки. У базовому підході, який називається k-fold cross-validation, навчальний набір даних розбивається на k менших наборів. Для кожного з k наборів застосовується така процедура:

– Модель навчається, використовуючи один з $k - 1$ наборів в якості навчальних даних;

– Отриману модель перевіряють на залишковій частині даних (тобто використовують як тестовий набір для обчислення метрики продуктивності, такої як точність).

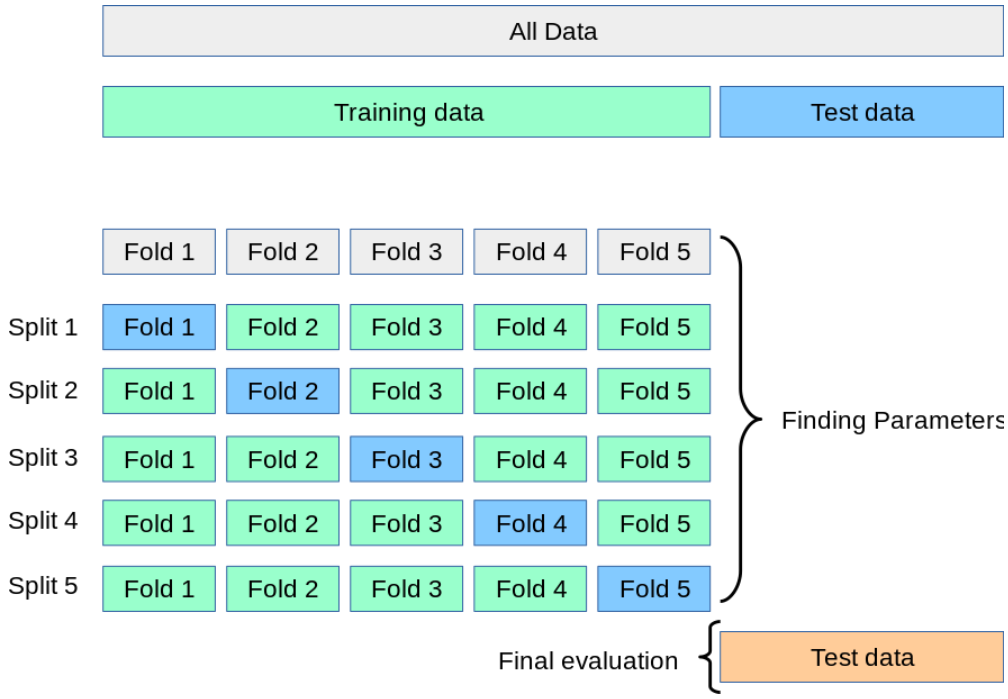


Рисунок 2.14 – Ілюстрація процесу перехресної перевірки [37]

Після цього середнє значення показників, обчислених в кожній ітерації, використовується як показник продуктивності. Хоча цей підхід може бути обчислювально витратним, він не втрачає значну кількість даних (відмінно від фіксації

довільного набору даних для перевірки), що є його головною перевагою в задачах, де кількість вибірок дуже мала.

2.4 Особливості використання квантового методу опорних векторів в кібербезпеці

У сучасному цифровому світі кібербезпека стає все більш важливою та актуальною проблемою. Із зростанням кількості даних та залежності від інформаційних технологій, кіберзлочинці стають більш винахідливими та продуктивними у своїх атаках, тож застосування машинного навчання є необхідним для підвищення рівня кібербезпеки [38].

Однією з ключових проблем у кібербезпеці є виявлення аномальної поведінки в мережі та раннє виявлення загроз [39]. Квантові методи в кібербезпеці знаходять все більше застосування [40]. Одним з таких методів є квантовий метод опорних векторів (QSVM). Наприклад, можна створити модель, яка навчиться розпізнавати небезпечний трафік, такий як DDoS атаки, та автоматично блокувати його.

DDoS (Distributed Denial of Service) вид кібератаки, який спрямований на перевантаження сервера або мережі штучним великим потоком запитів [41]. У результаті цього легітимні користувачі не можуть отримати доступ до ресурсів, які на цьому сервері знаходяться. Для проведення атаки використовуються ботнети, тобто багато комп'ютерів, які знаходяться під контролем зловмисників. Ці комп'ютери відправляють великі об'єми запитів на цільовий сервер або мережу, що призводить до перевантаження та відмови у обслуговуванні. DDoS-атаки можуть бути направлені на будь-яку мережу, що забезпечує доступ до Інтернету, включаючи компанії, установи, урядові організації, медіа-ресурси та індивідуальних користувачів. За 2022 кількість DDoS атак збільшилось на 150% відносно 2021 року [42].

Для того, щоб розробники та дослідники могли ефективно експериментувати з квантовими моделями машинного навчання, доступні бібліотеки та платформи. Зокрема, існують спеціалізовані інструменти для класифікації табличних даних, які підтримують квантові методи опорних векторів та інші квантові моделі.

Ці бібліотеки та платформи дозволяють розробникам та дослідникам зосередитися на розробці та тестуванні квантових алгоритмів, не витрачаючи значну кількість часу на інфраструктурні питання. Більшість з цих бібліотек та платформ є відкритими джерелами та надаються спільнотою розробників безкоштовно. Ось деякі з них:

– Cirq - фреймворк от компанії Google, що надає розробникам можливість писати код для квантових обчислень [43]. Код можна запускати на симуляторах, які можуть працювати локально або у хмарі, а також використовуючи спеціально розроблені компанією Google симулятори на базі Tensor Processing Units (TPU) за допомогою Floq API. Для квантового машинного навчання можна використовувати Tensorflow Quantum - розширення бібліотеки Tensorflow, яке дозволяє комбінувати класичні та квантові нейронні мережі. Крім того, до екосистеми Cirq належить бібліотека для квантової хімії OpenFermion;

– Qiskit - це фреймворк для квантового програмування та обчислень від IBM [44]. Цей інструментарій дозволяє дослідникам та розробникам працювати з квантовими алгоритмами та квантовими обчислювальними системами. Зокрема, для реалізації квантового машинного навчання Qiskit надає бібліотеку Qiskit Machine Learning, яка містить алгоритми квантового машинного навчання, такі як квантові варіаційні алгоритми та квантові алгоритми для глибинного навчання. Крім того, Qiskit надає користувачам можливість взаємодіяти з квантовими комп'ютерами IBM через хмарний сервіс IBM Quantum. Квантові пристрої можна контролювати та запускати задачі через API Qiskit. Отже, Qiskit є потужною платформою для розробки квантових обчислювальних програм та застосування їх у задачах машинного навчання;

– PennyLane - це бібліотека для квантового обчислення та машинного навчання, розроблена з використанням мови Python [45]. Вона дає можливість створювати та тренувати квантові нейронні мережі на різних квантових платформах, включаючи симулятори та реальні квантові комп'ютери. Бібліотека PennyLane та-

кож містить інструменти для автоматичного диференціювання квантових функцій, що дозволяє автоматично обчислювати градієнти для квантових алгоритмів, що є важливим для багатьох задач машинного навчання.

Ці фреймворки та бібліотеки є потужними інструментами для дослідження та розробки методів машинного навчання, зокрема, застосування методу квантових опорних векторів для класифікації даних кібератак.

Поетапну реалізацію задачі класифікації кібератак зображено на схемі:

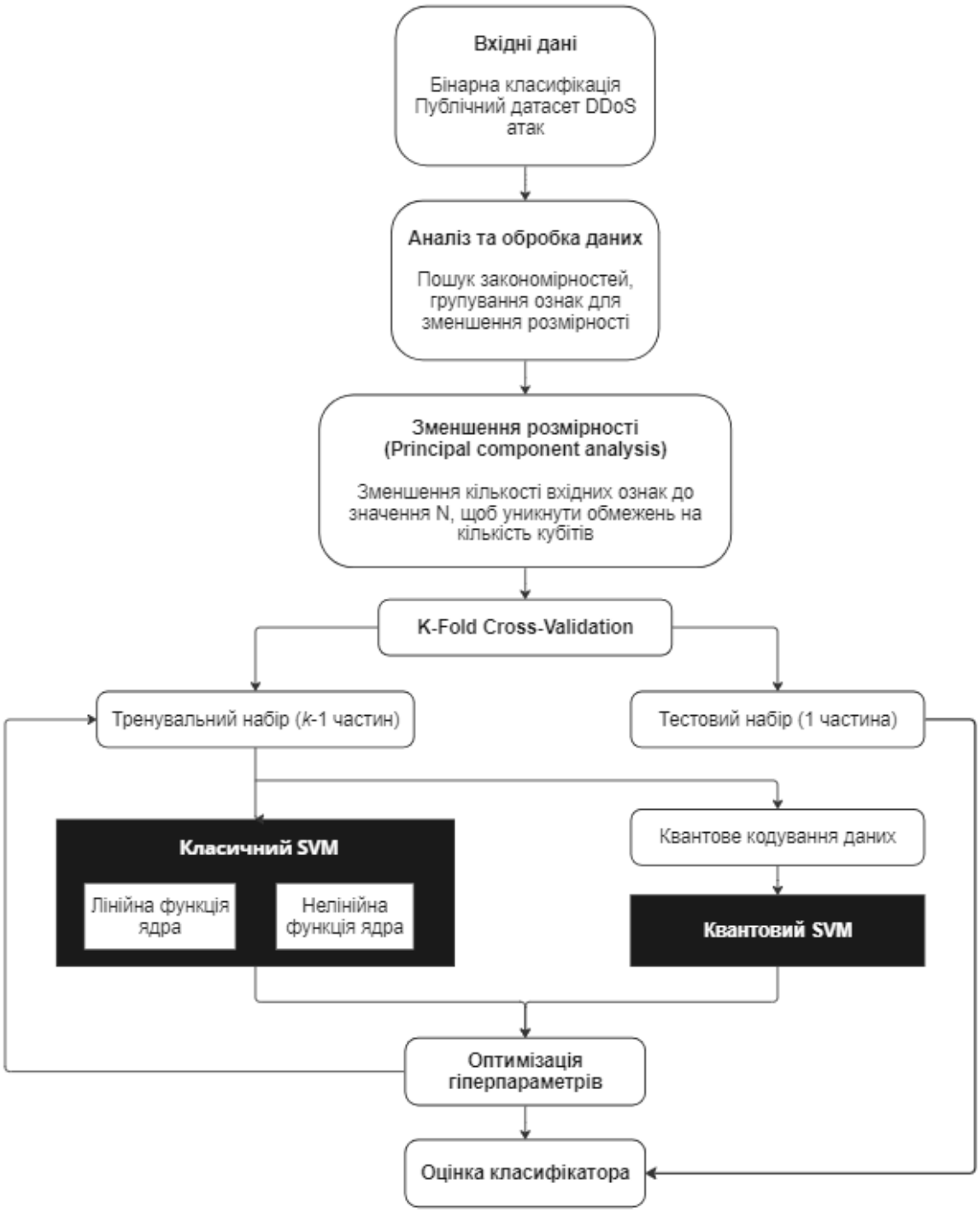


Рисунок 2.15 – Поетапна схема реалізації алгоритму класифікації DDoS атак

2.5 Особливості використання квантового методу опорних векторів в екології

Одним з головних принципів екології є збереження природних ресурсів та зменшення впливу людської діяльності на довкілля [46]. Застосування методів квантового машинного навчання може допомогти вирішувати складні екологічні проблеми та збільшувати ефективність процесів збереження природних ресурсів.

Наприклад, квантові алгоритми можуть бути використані для прогнозування зміни клімату, аналізу використання ґрунту та водних ресурсів, моніторингу відходів та управління енергоефективністю [47]. Квантові алгоритми можуть також використовуватися для розробки більш точних та ефективних методів контролю за забрудненням повітря та води.

Квантове машинне навчання дозволяє обробляти великі обсяги даних, знаходити приховані залежності та прогнозувати майбутні події, що дозволяє екологам та науковцям діяти більш точно та швидко. Крім того, квантові алгоритми можуть допомогти вирішувати проблеми з обробкою даних, пов'язаних з недостатньою точністю та неповнотою даних.

Застосування квантових алгоритмів у екології може допомогти створити більш точні та ефективні методи збереження природних ресурсів та зменшення впливу людської діяльності.

Проблема забруднення води є однією з найбільших екологічних проблем нашого часу [48]. Забруднення води може стати причиною поширення водних захворювань та смертності серед людей та тварин, а також призвести до виснаження природних ресурсів води.

QSVM може бути використаний для класифікації водних даних, включаючи дані про рівень забруднення води. Алгоритм QSVM може бути навчений на основі вхідних даних про якість води, включаючи параметри, такі як рівень pH, розчинені речовини, концентрація кисню та інші.

Після навчання QSVM може бути використаний для класифікації якості води на категорії, такі як безпечна, або небезпечна якість води. Ця класифікація

допоможе визначити рівень забруднення води та вжити необхідні заходи для його зменшення.

Одним з переваг використання QSVM є можливість обробки великих обсягів даних за короткий час. Крім того, QSVM може бути використаний для класифікації даних з високою точністю, що робить його ефективним інструментом для визначення якості води.

Поетапну реалізацію задачі класифікації якості води зображено на схемі:

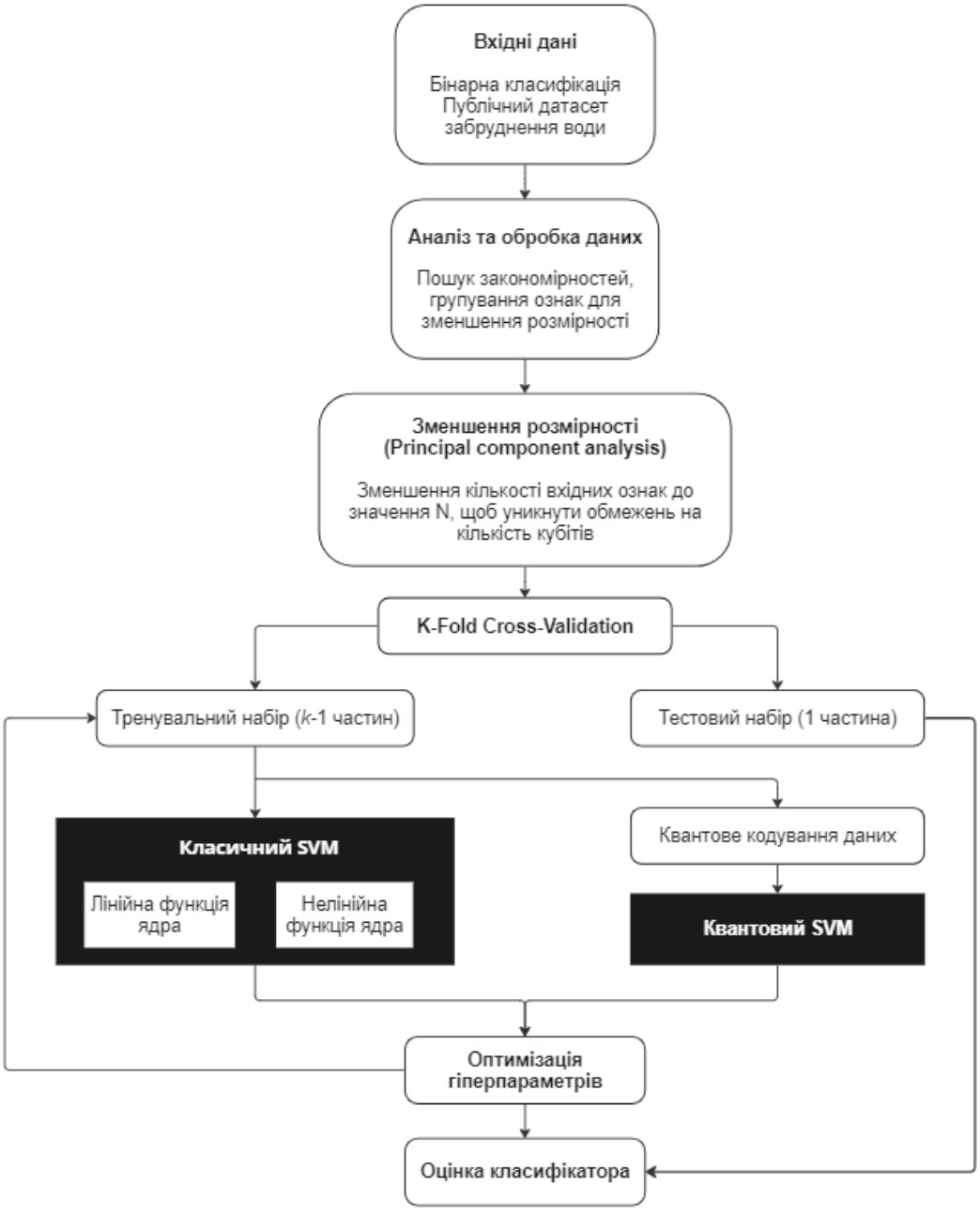


Рисунок 2.16 – Поетапна схема реалізації алгоритму класифікації якості води

2.6 Особливості використання квантового методу опорних векторів для економічних систем та фінансової аналітики

Методи квантового машинного навчання вже давно зацікавили фінансовий сектор, оскільки вони можуть значно покращити ефективність і точність фінансових прогнозів та аналітики [49]. Квантові алгоритми машинного навчання дозволяють обробляти величезні об'єми фінансових даних та знаходити складні зв'язки між різними факторами, що впливають на ринки.

Одним з головних застосувань методів квантового машинного навчання в фінансах є передбачення цін на акції та інші фінансові інструменти. Квантові алгоритми можуть аналізувати величезні об'єми даних, включаючи новини, статистику та соціальні мережі, щоб зрозуміти, які фактори впливають на ринок та які фактори можуть призвести до зміни цін.

Квантові алгоритми також можуть бути використані для визначення ризиків у фінансових ринках та розробки стратегій ризик-менеджменту. Вони можуть аналізувати історичні дані та прогнозувати можливі ризики на ринках, щоб допомогти інвесторам уникнути можливих втрат. Наприклад прогнозувати проблему відтоку клієнтів.

Проблема відтоку клієнтів є серйозною проблемою для бізнесів у всіх галузях. Втратити клієнта може коштувати компанії значну кількість грошей та ресурсів, що було витрачено на залучення та утримання клієнта. Тому, класифікація відтоку клієнтів є дуже важливою задачею для бізнесу.

Квантовий метод опорних векторів (QSVM) є одним з методів квантового машинного навчання, який може бути застосований для класифікації відтоку клієнтів. QSVM працює на основі аналізу даних клієнтів, що зберігаються та тих, які відходять. Дані про клієнтів можуть включати такі параметри, як кількість покупок, витрати, тривалість взаємодії з компанією та багато іншого. QSVM може використовувати ці параметри для того, щоб побудувати модель, яка буде передбачати, чи залишиться клієнт з компанією чи ні.

QSVM може бути використаний для класифікації відтоку клієнтів у режимі реального часу, що дозволяє компаніям вчасно приймати необхідні заходи для

того, щоб утримати клієнта. Компанії можуть використовувати цю інформацію для покращення сервісу та продуктів, що надаються клієнтам.

У підсумку, QSVM є корисним інструментом для бізнесу, який може допомогти в зменшенні втрат клієнтів та збільшенні прибутку.

Поетапну реалізацію задачі класифікації відтоку клієнтів зображено на схемі:

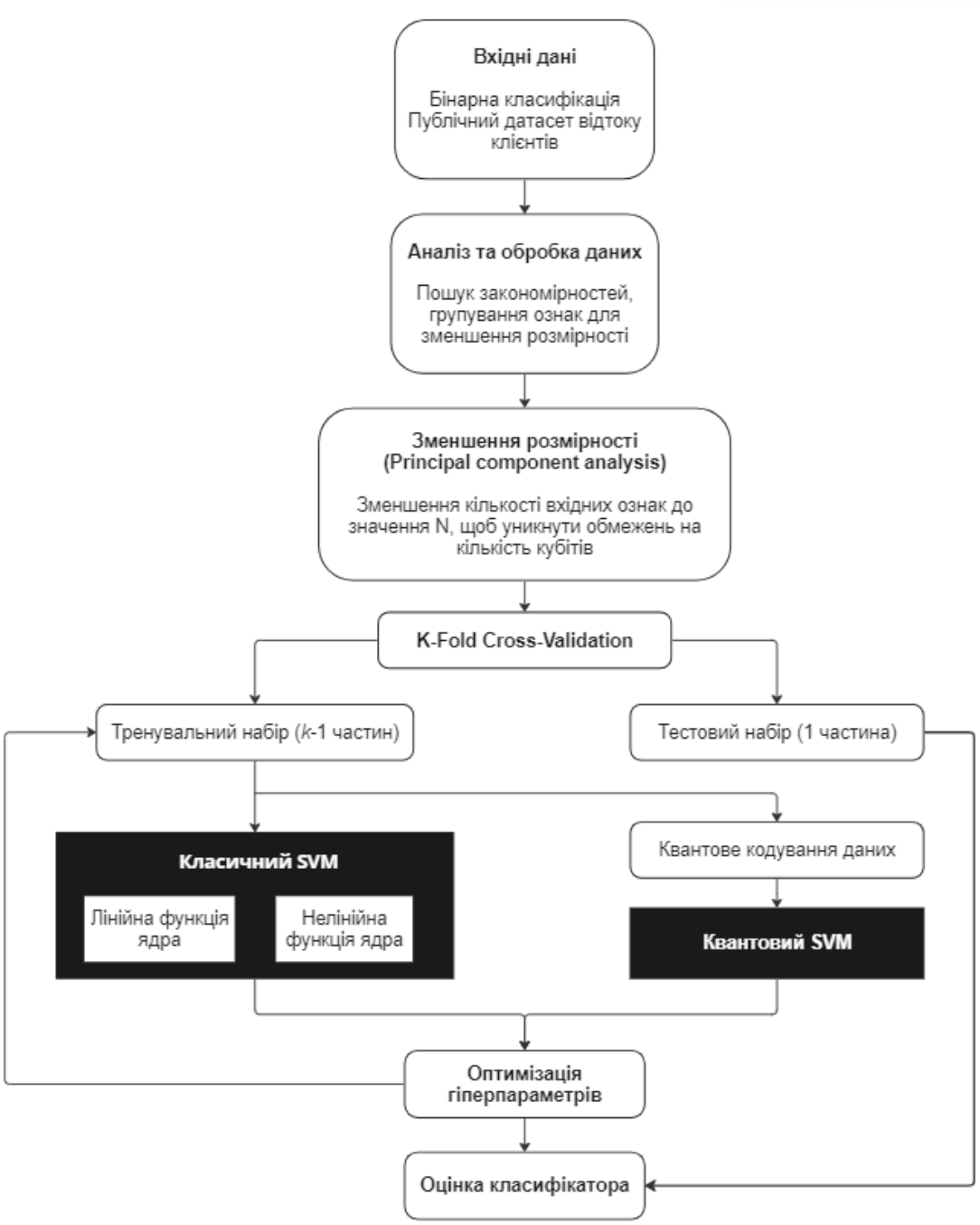


Рисунок 2.17 – Поетапна схема реалізації алгоритму класифікації відтоку клієнтів

2.7 Висновки до розділу 2

Метод квантових опорних векторів має широкий потенціал застосування в різних галузях, проте для його успішного використання необхідно мати достатню кількість квантових бітів та ефективні методи навчання та оптимізації моделей. У кібербезпеці важливо усвідомлювати складність та потребу у великих ресурсах при розробці та використанні цього методу. Дальші дослідження та розробки можуть сприяти вирішенню складних проблем в кібербезпеці та допомагати запобігати кіберзлочинності.

У дослідженнях екологічних проблем, метод квантових опорних векторів може також допомогти в розробці більш ефективних та точних систем для визначення рівня забруднення повітря та ґрунту. Такі системи можуть допомогти зберегти навколишнє середовище та покращити якість життя людей. Крім того, метод квантових опорних векторів може бути використаний для підвищення точності та ефективності прогнозування погодних умов та стихійних лих, що може допомогти зменшити ризиків для населення та економіки.

Метод опорних векторів в прогнозуванні відтоку клієнтів в фінансових задачах має великий потенціал, і може стати конкурентом для класичних методів класифікації в найближчому майбутньому. Для подальшого розвитку цих застосувань потрібно вдосконалювати алгоритми, збільшувати кількість квантових ресурсів, а також активно використовувати нові підходи до навчання та оптимізації моделей. Крім того, ефективність методу опорних векторів може бути покращена шляхом використання великих наборів даних та покращенням якості вхідних даних. Також можливо використовувати інші методи машинного навчання разом з методом опорних векторів, щоб отримати ще більш точні та надійні прогнози. Для успішного використання методу опорних векторів в фінансових задачах також необхідно враховувати ризики та використовувати надійні та безпечні методи зберігання та обробки фінансових даних.

Розділ 3 Програмна реалізація з застосуванням квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем

3.1 Програмна реалізація з застосуванням квантового методу опорних векторів для класифікації DDoS атак

Першим етапом реалізації алгоритму класифікації DDoS атак є знаходження набору даних, який буде використовуватись для тренування моделі класифікації. Цей набір даних повинен містити достатньо інформації про характеристики атак, щоб забезпечити точність та ефективність класифікації.

Для знаходження набору даних можна використовувати різні джерела, наприклад, журнали серверів, мережеві протоколи, відгуки користувачів та інші джерела інформації. Важливо відібрати тільки значущі дані та виключити шум.

Для цього алгоритму було використано набір даних “SDN” [50], який містить інформацію про мережевий трафік та характеристики пакетів. Структура даних описана за допомогою таблиці, наведеної нижче:

Таблиця 3.1 – Атрибути таблиці DDoS атак

№ з/п	Назва атрибуту	Тип даних	Опис
1.	dt	INT	Дата в секундах
2.	switch	INT	Ідентифікатор мережевого комутатора
3	src	STRING	IP-адреса джерела
4.	dst	STRING	IP-адреса призначення
5.	pktcount	INT	Кількість переданих пакетів

6	bytecount	INT	Кількість переданих байтів
7.	dur	INT	Тривалість запитів
8.	flows	INT	Кількість потоків
9.	packetins	INT	Кількість пакетів
10.	pktperflow	INT	Кількість пакетів, що надсилаються на один потік
11.	byteperflow	INT	Кількість байтів, що надсилаються на один потік
12.	pktrate	INT	Кількість пакетів, що надсилаються за секунду
13.	Pairflow	INT	Кількість пакетів на потік
14.	Protocol	STRING	Мережевий протокол
15.	port_no	INT	Кількість портів
16.	tx_bytes	INT	Швидкість передачі байтів
17.	rx_bytes	INT	Швидкість отримання байтів
18.	tx_kbps	INT	Швидкість передачі пакетів
19.	rx_kbps	FLOAT	Швидкість отримання пакетів
20.	tot_kbps	FLOAT	Загальна кількість пакетів

21.	label	INT	Мітка, яка вказує на те, чи є трафік безпечним або зловмисним

Наступним важливим етапом є аналіз даних. Після збору набору даних, їх потрібно обробити та візуалізувати для пошуку корисної інформації та закономірностей, які можуть бути використані для покращення алгоритмів класифікації DDoS атак.

На першому етапі аналізу даних було проведено вивчення розподілу міток, що вказують на кількість зловмисних та безпечних запитів в датасеті.

За результатами аналізу було встановлено, що кількість безпечних запитів перевищує кількість зловмисних запитів на 22%. Однак, різниця в кількості міток не є дуже великою, тому можна припустити, що класи достатньо збалансовані для подальшої обробки та моделювання. Результати аналізу було проілюстровано на рисунку 3.1.

Рисунок 3.1 – Розподіл трафіку за типом

Для отримання додаткової інформації про дані, було проаналізувано розподіл кількості запитів по портах. З графіку можна зрозуміти, що більшість запитів розподілені рівномірно. Але порти 10.0.0.10 та 10.0.0.4 мають значно більшу

кількість запитів порівняно з іншими портами, що може свідчити про їхню важливість для мережі. Результати аналізу було проілюстровано на рисунку 3.2.

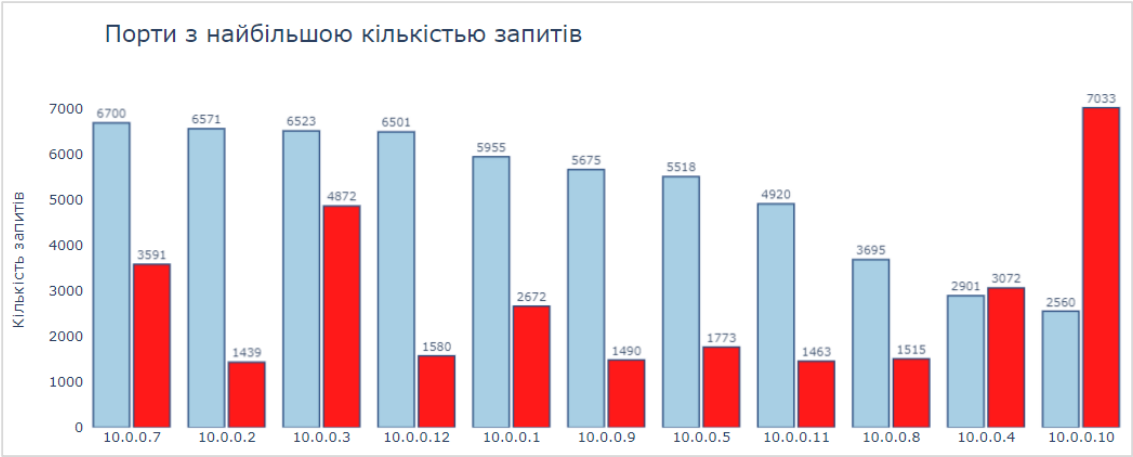


Рисунок 3.2 – Порти з найбільшою кількістю запитів

Також важливим етапом є аналіз розподілу кількості запитів за різними протоколами, який може допомогти виявити можливі аномалії в мережі та порівняти частоту використання протоколів. Для цього було побудовано стовпчасту діаграму, яка ясно показує, що протокол UDP має значно більше зловмисних запитів, ніж протокол TCP та ICMP, що може вказувати на його слабкість у захисті від атак. Протокол TCP має майже рівномірний розподіл, а протокол ICMP має низьку кількість зловмисних запитів. Ці дані можуть допомогти зрозуміти, які протоколи використовуються найчастіше та знайти можливі вразливості мережі, які потребують уваги та виправлення. Результати аналізу було проілюстровано на рисунку 3.3.

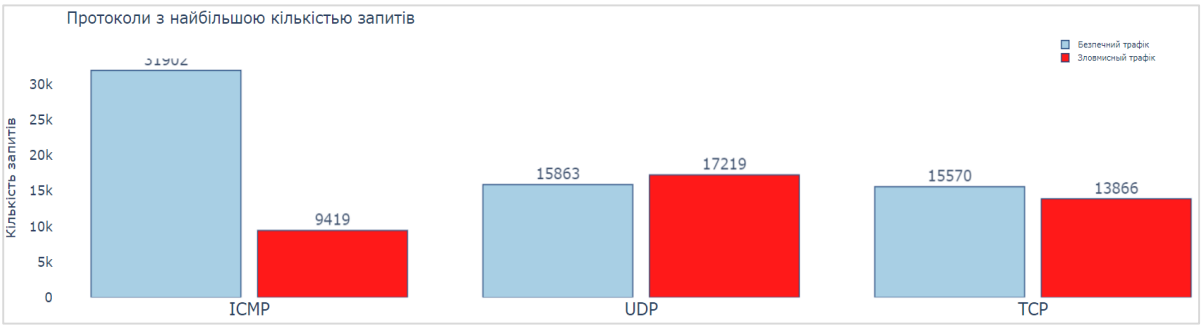


Рисунок 3.3 – Протоколи з найбільшою кількістю запитів

Для аналізу розподілу кількості запитів по комутаторам, було побудовано стовпчасту діаграму, на якій можна побачити, що більшість запитів було здійснено на комутаторах 4, 3 та 5. Зокрема, на комутаторах 4 та 3 було понад 8 тисяч зловмисних запитів. Найменшу кількість запитів мають комутатори 8, 9 та 10. Варто зазначити, що у комутаторів 8, 9 та 10 розподіл безпечних та зловмисних запитів майже порівну. Це може вказувати на їхню відносну безпеку та потенційну можливість для покращення захисту мережі. Детальний аналіз розподілу кількості запитів по кожному комутатору може допомогти знайти можливі вразливості та уразливі місця в мережі, а також прийняти відповідні заходи для забезпечення безпеки. Результати аналізу було проілюстровано на рисунку 3.4.

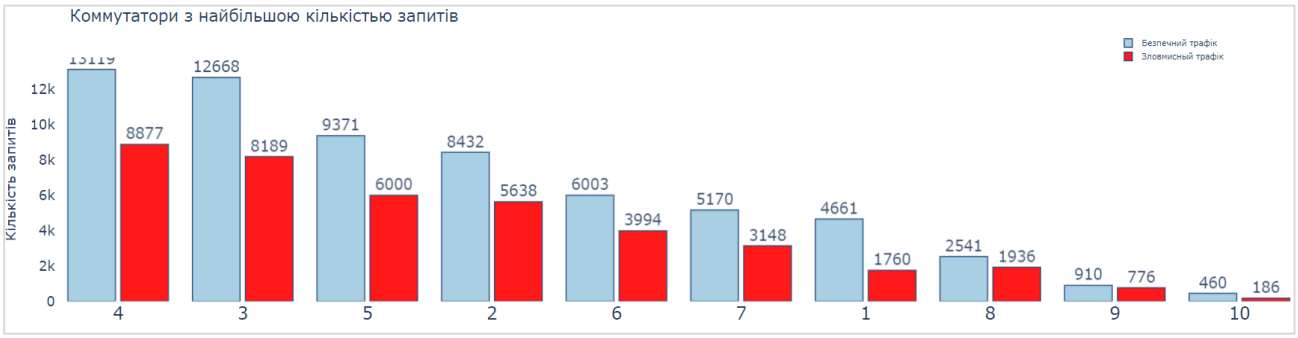


Рисунок 3.4 – Комутатори з найбільшою кількістю запитів

Був проведений аналіз розподілу тривалості запитів, згідно з яким можна зробити висновок, що більшість запитів здійснювалась протягом інтервалу від 10 до 500 секунд. Кількість безпечних та зловмисних запитів була відносно рівна, але в інтервалі від 300 до 500 секунд кількість зловмисних атак була більшою. Після 600 секунд кількість зловмисних запитів зменшилась до мінімуму. Результати аналізу було проілюстровано на рисунку 3.5.

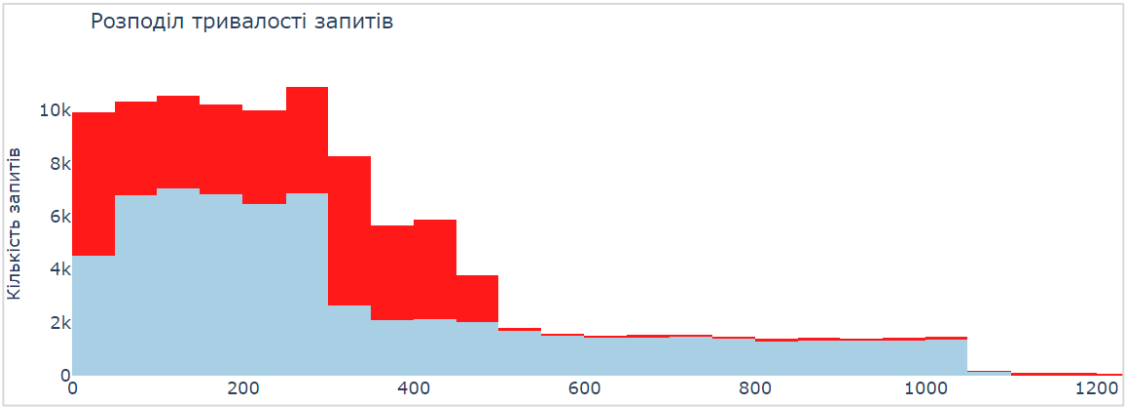


Рисунок 3.5 – Розподіл тривалості запитів

Також був проведений аналіз матриці кореляції ознак. Виявлено, що ознаки, які відповідають за кількість передачі даних, мають високу кореляцію у 0.81. Також варто зазначити, що згідно з аналізом матриці кореляції, інші ознаки, такі як час, тривалість запитів та їх тип, мають слабку кореляцію одна з одною, тобто ці ознаки мало пов'язані між собою. Результати аналізу було проілюстровано на рисунку 3.6.

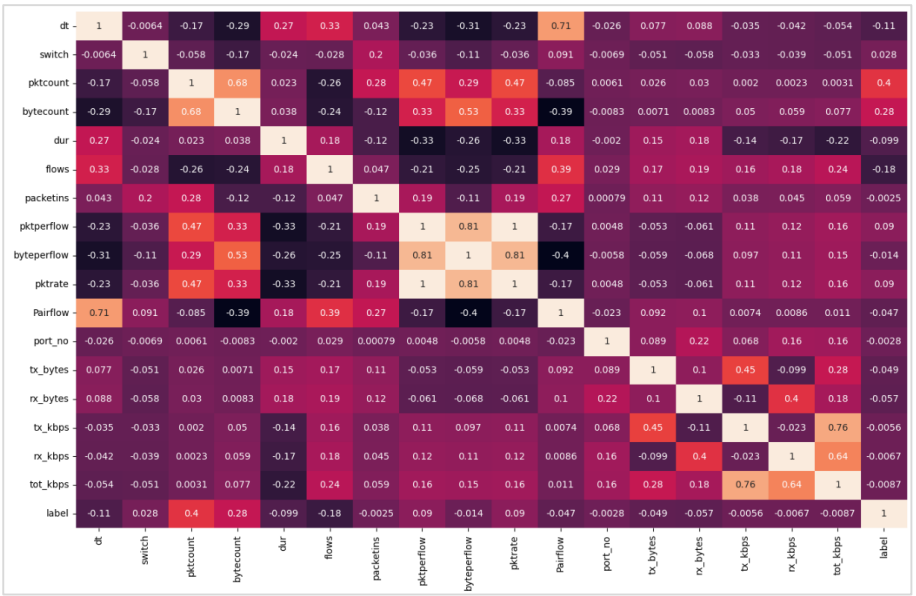


Рисунок 3.6 – Матриця кореляції ознак

На наступному етапі проводиться обробка даних з метою підготовки їх для використання в моделях класифікації. Перший крок полягає у перетворенні категоріальних змінних на числові, щоб класифікатор міг ефективно опрацьовувати ці дані.

В наявному датасеті представлено три категоріальні змінні: src, dst, Protocol. Приклад кодування проілюстровано на рисунку 3.7.

а)

10.0.0.3
10.0.0.10
10.0.0.7
10.0.0.1
10.0.0.2
10.0.0.12
10.0.0.9

б)

12
1
16
0
10
3
18

Рисунок 3.7 – а) Початкові значення змінної src, б) Значення змінної src після кодування

Після цього йде етап шкалювання даних, з метою приведення їх до інтервалу від 0 до 1. Таке приведення даних до єдиної шкали допомагає моделі краще розпізнавати та узагальнювати патерни, забезпечуючи більш точні та надійні результати класифікації. Поетапний процес шкалювання наведений на рисунку 3.8.

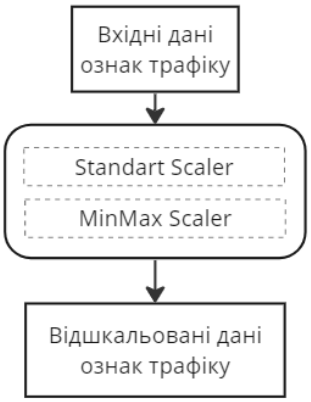


Рисунок 3.8 – Поетапний процес шкалювання даних ознак трафіку

Наступним кроком є операція зменшення розмірності даних. Ця операція дозволяє зменшити кількість ознак у наборі даних, зберігаючи при цьому якомога більше інформації. Зменшення розмірності буде досягнуто шляхом використання методів, таких як аналіз головних компонентів (PCA) та вибір k-кращих (SelectKBest). Поетапний процес наведений на рисунку 3.9, ілюстрація методу головних компонент на рисунку 3.10.

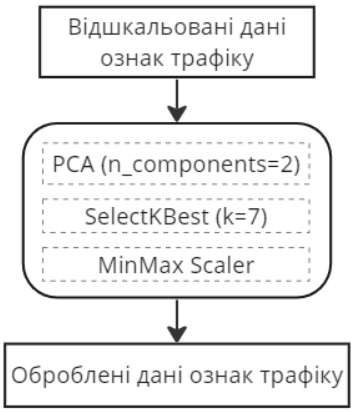


Рисунок 3.9 – Поетапний процес зменшення розмірності ознак трафіку

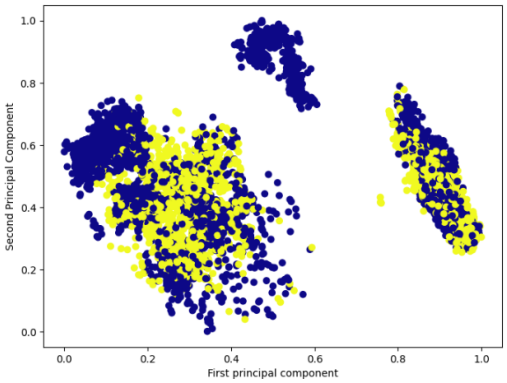


Рисунок 3.10 – Ілюстрація розподілу ознак трафіку після зменшення розмірності

Наступним є етап моделювання, де використовуються різні методи для класифікації та використовувати метод перехресної валідації для тренування. Першим методом буде класичний метод опорних векторів з лінійною функцією ядра. Поетапний процес побудови лінійної моделі наведений на рисунку 3.11.

Рисунок 3.11 – Поетапний процес побудови лінійної моделі класифікації зловмисного трафіку

Другим методом буде класичний метод опорних векторів з поліноміальною функцією ядра. Поетапний процес побудови поліноміальної моделі наведений на рисунку 3.12.

Рисунок 3.12 – Поетапний процес побудови поліноміальної моделі класифікації зловмисного трафіку

Також було застосовувано квантовий метод опорних векторів. Спочатку побудовано схему оцінки квантового ядра. Ілюстрація квантового ядра для квантового моделі класифікації зловмисного трафіку наведена на рисунку 3.13.

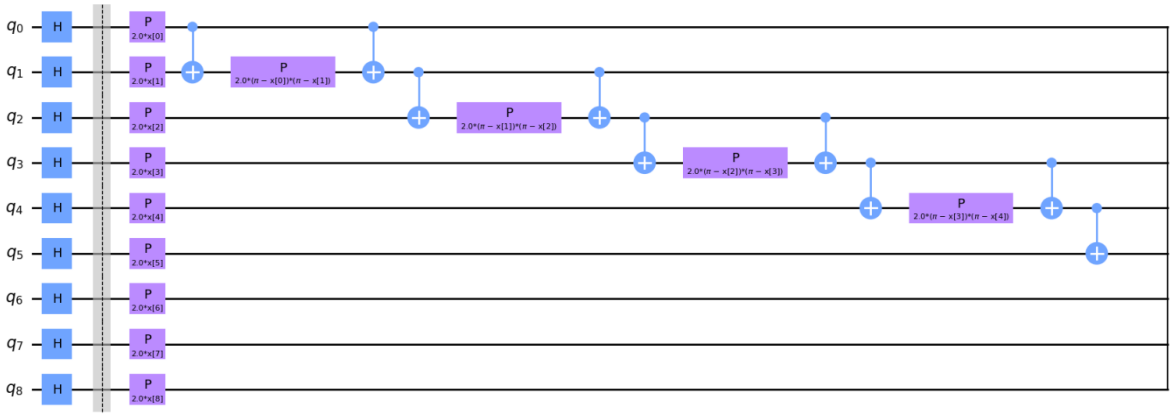


Рисунок 3.13 – Ілюстрація схеми оцінки квантового ядра моделі класифікації зловмисного трафіку

Далі реалізовано квантовий метод опорних векторів, використовуючи схему оцінки ядра, яку було побудовано на попередньому кроці. Спочатку побудували ядро функції, а далі застосували його в класичному методі опорних векторів. Поетапний процес побудови квантової моделі представлений на рисунку 3.14.

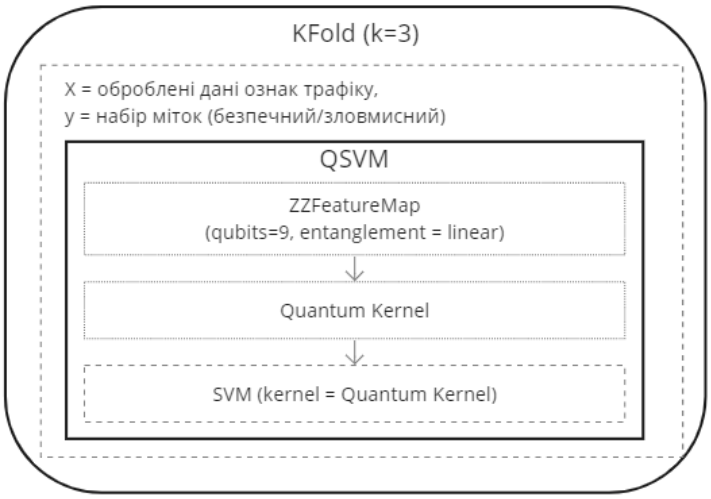


Рисунок 3.14 – Поетапний процес побудови квантової моделі класифікації зловмисного трафіку

Фінальним кроком є оцінка якості класифікаторів, що були побудовані. Результати класифікації представлені в таблиці 3.2.

Таблиця 3.2 – Результати класифікації зловмисного трафіку

Модель	**Точність**	**Влучність**	**Повнота**
Лінійний класифікатор	0.7162	0.6409	0.5458
Поліноміальний класифікатор	0.8974	0.8796	0.8410
Квантовий класифікатор	0.9255	0.8972	0.9043

3.2 Програмна реалізація з застосуванням квантового методу опорних векторів для класифікації якості води

Першим важливим кроком у впровадженні алгоритму класифікації якості води є детальний пошук та збір необхідного набору даних. Цей процес включає аналіз різних джерел і джерел з інформацією про якість води, такі як дослідницькі студії, офіційні звіти, водний моніторинг та інші джерела, що можуть містити відповідні дані.

Під час пошуку набору даних важливо враховувати різні аспекти, такі як розмір вибірки, географічне розподілення джерел даних, типи параметрів води, які необхідно включити, та їх точність.

Для реалізації даного алгоритму було використано набір даних під назвою "Water Quality" [51]. Цей набір даних містить ретельно зібрану і систематизовану інформацію про якість води з різних джерел. Структура даних описана за допомогою таблиці, наведеної нижче:

Таблиця 3.3 – Атрибути таблиці якості води

№ з/п	**Назва атрибуту**	**Тип даних**	**Опис**
1.	pH	FLOAT	Кислотно-лужний балансу води
2.	Hardness	FLOAT	Ідентифікатор мережевого комутатора
3.	Solids	FLOAT	Загальна кількість розчинених твердих речовин
4.	Chloramines	FLOAT	Кількість хлораміну
5.	Sulfate	FLOAT	Кількість розчинених сульфатів
6.	Conductivity	FLOAT	Електропровідність води
7.	Organic_carbon	FLOAT	Загальний органічний вміст
8.	Trihalomethanes	FLOAT	Кількість тригалометану

9.	Turbidity	FLOAT	Показник світлопропускної властивості води в одиницях
10.	Potability	INT	Мітка, чи безпечна вода для споживання людиною

Після збору необхідного набору даних наступним важливим етапом є аналіз цих даних. Аналіз даних дозволяє виявити взаємозв'язки, та важливі характеристики, які впливають на якість води.

Подивимось на розподіл зразків води. Після проведення аналізу було встановлено, що кількість безпечних запитів перевищує кількість зловмисних запитів на 22%. Проте, різниця в кількості міток не є значною, що дає підстави вважати, що класи достатньо збалансовані для подальшої обробки та моделювання. Результати аналізу було проілюстровано на рисунку 3.15.

Рисунок 3.15 – Розподіл зразків води

Наступним етапом є аналіз розподілу значень, який дозволяє отримати докладніші відомості про розподіл кожного параметра або ознаки у наборі даних. Під час аналізу було виявлено, що значення в значній мірі розподілені за нормальним розподілом. Результати аналізу було проілюстровано на рисунку 3.16.

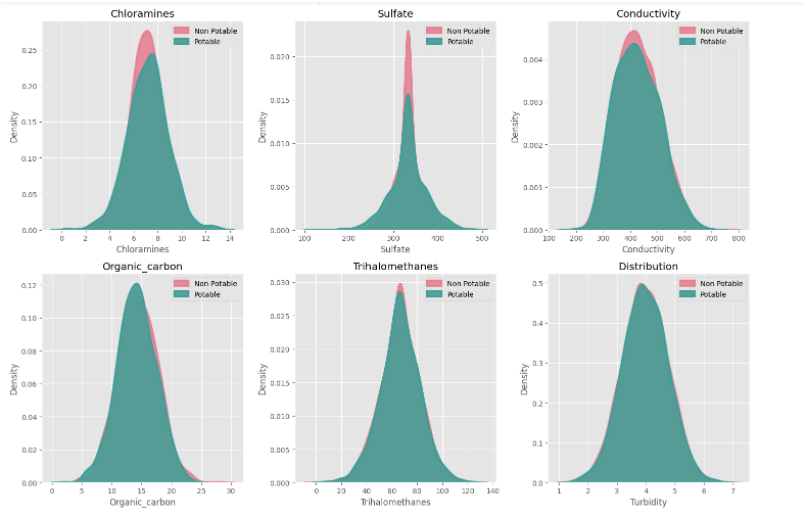


Рисунок 3.16 – Розподіл ознак якості води

Наступний крок це аналіз матриці кореляції ознак. Результати аналізу було проілюстровано на рисунку 3.17.

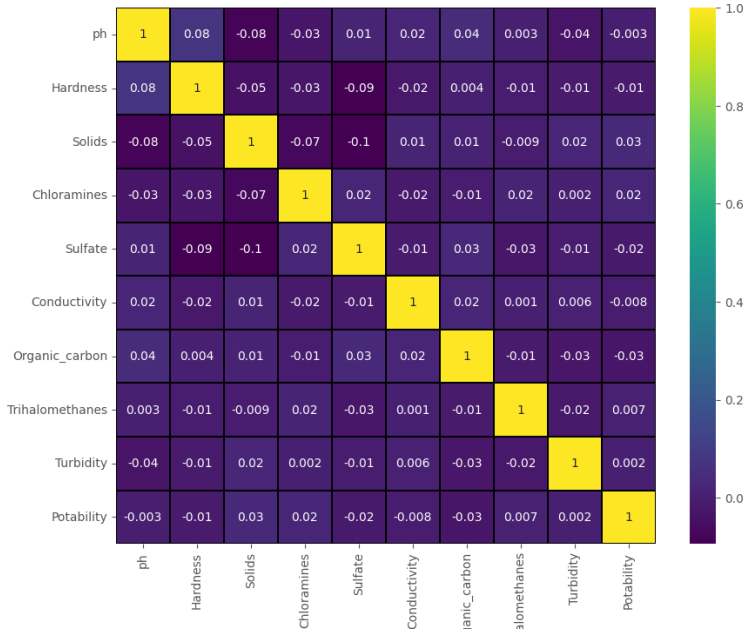


Рисунок 3.17 – Матриця кореляції ознак якості води

Після аналізу настає етап обробки даних, який є необхідним для підготовки їх до використання у моделях класифікації. У даному наборі даних, який містить числові значення, першим кроком є шкалювання даних та зменшення їх розмірності. Процес обробки даних якості води наглядно представлений на рисунку 3.18.

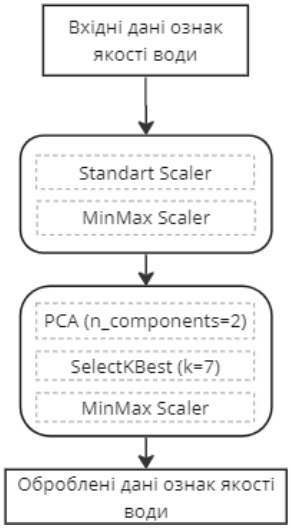


Рисунок 3.18 – Процес обробки даних якості води

Наступний етап – моделювання. Перша модель - класичного методу опорних векторів з лінійною функцією ядра. Побудова цієї лінійної моделі показана поетапно на рисунку 3.19.

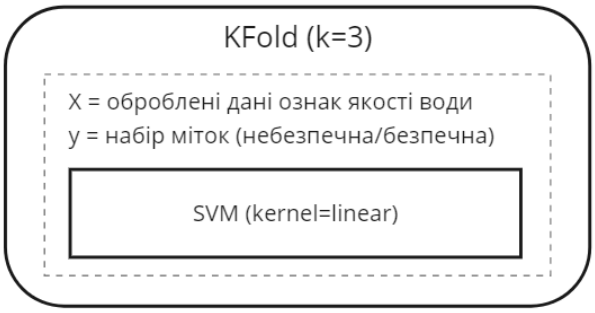


Рисунок 3.19 – Процес побудови лінійної моделі класифікації якості води

У другому методі використовується стандартний метод опорних векторів з використанням поліноміальної функції ядра. Покроковий процес побудови цієї поліноміальної моделі можна побачити на рисунку 3.20.

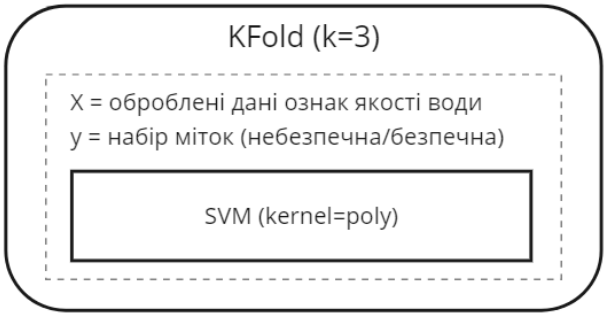


Рисунок 3.20 – Процес побудови поліноміальної моделі класифікації якості води

Для впровадження квантового методу опорних векторів було розроблено схему для оцінки квантового ядра. Ілюстрація квантового ядра для моделі класифікації якості води наведена на рисунку 3.21.

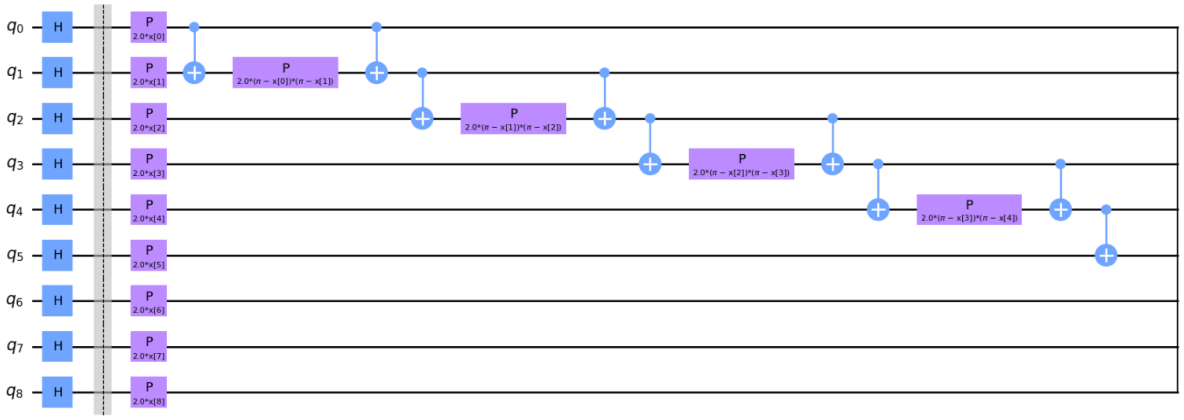


Рисунок 3.21 – Ілюстрація схеми оцінки квантового ядра моделі класифікації якості води

Подальшим кроком є створення квантового ядра функції шляхом застосування оцінки ядра, побудованої на попередньому етапі, а потім використовуємо його у класичному методі опорних векторів. На рисунку 3.22 показано послідовний процес створення квантової моделі.

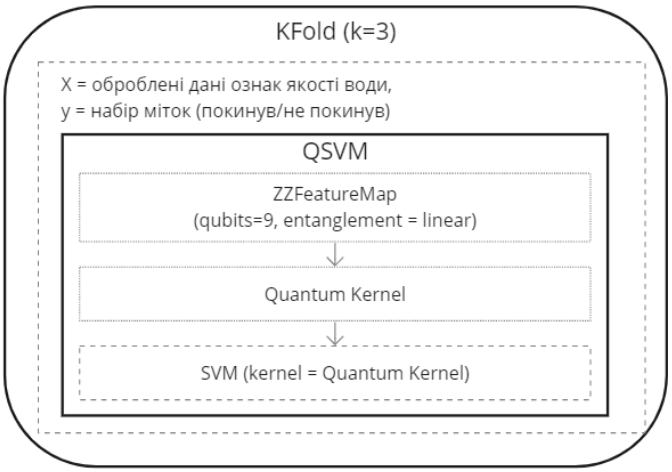


Рисунок 3.22 – Поетапний процес побудови квантової моделі класифікації якості води

Після успішного завершення процесу побудови класифікаторів, настає момент оцінки їх якості. У таблиці 3.4 представлені результати цієї оцінки.

Таблиця 3.4 – Результати класифікації якості води

Модель	**Точність**	**Влучність**	**Повнота**
Лінійний класифікатор	0.6098	0.0000	0.0000
Поліноміальний класифікатор	0.5952	0.4844	0.1280
Квантовий класифікатор	0.5863	0.4452	0.2029

3.3 Програмна реалізація з застосуванням квантового методу опорних векторів для класифікації відтоку клієнтів

Першим і найважливішим кроком у впровадженні алгоритму класифікації відтоку клієнтів є здійснення пошуку та збору необхідного набору даних. Цей процес включає в себе організацію та здійснення збору інформації про клієнтів, таку як їх особисті дані, історію покупок, взаємодії з компанією, поведінку на веб-сайті та інші фактори, що можуть впливати на їх рішення про зміну постачальника послуг.

Ефективний пошук набору даних включає у себе ідентифікацію та включення релевантних джерел інформації, таких як бази даних клієнтів, журнали взаємодії з клієнтами, інтернет-ресурси, опитування та будь-яку іншу наявну інформацію, яка може бути корисною для аналізу відтоку клієнтів.

Алгоритм класифікації відтоку клієнтів був реалізований з використанням набору даних, відомого як "Churn Modelling" [52]. Для опису структури даних використовується таблиця, яка подана нижче:

Таблиця 3.5 – Атрибути таблиці відтоку клієнтів

№ з/п	**Назва атрибуту**	**Тип даних**	**Опис**
1.	Surname	STRING	Прізвище клієнта
2.	CreditScore	INT	Кредитна оцінка клієнта
3.	Geography	STRING	Країна розташування клієнта
4.	Gender	STRING	Стать клієнта
5.	Age	INT	Вік клієнта
6.	Tenure	INT	Кількість років обслуговування в банку
7.	Balance	FLOAT	Баланс клієнта
8.	NumOfProducts	INT	Кількість продуктів які використовує клієнт
9.	HasCrCard	BOOLEAN	Чи має клієнт кредитну карту
10.	IsActiveMember	BOOLEAN	Активний клієнт чи ні
11.	EstimatedSalary	FLOAT	Приблизна зарплатня
12.	Exited	INT	Покинув банк чи ні

Після збору необхідного набору даних, наступним важливим етапом є їх аналіз. Цей етап включає обробку та візуалізацію даних з метою виявлення корисної інформації та закономірностей, які можуть бути використані для поліпшення алгоритмів класифікації відтоку клієнтів.

Початковим завданням аналізу було встановлення розподілу кількості кредитних карток залежно від віку клієнтів. Після проведеного аналізу було виявлено, що більшість клієнтів, незалежно від наявності кредитних карток, зосереджена в групі віком від 25 до 45 років. Одночасно, у інших вікових групах кількість власників кредитних карток та не користувачів залишається стабільною. Результати аналізу було проілюстровано на рисунку 3.23.

Рисунок 3.23 – Розподіл кількості кредитних карток за віком

Внаслідок аналізу попереднього графіку виникла необхідність визначити кількість клієнтів, які відмовляються від послуг банку, при цьому володіючи кредитними картками, і їх вікову категорію. Для цього було створено відповідну візуалізацію, яка показала, що найбільша кількість клієнтів без кредитних карток покидали банк у віці від 39 до 52 років (медіана = 45), що практично відповідає кількості клієнтів з кредитними картками, які покинули банк. Можна зробити висновок, що використання кредитних карток не має прямого впливу на рішення клієнтів про вихід з банку для будь-якої вікової групи.

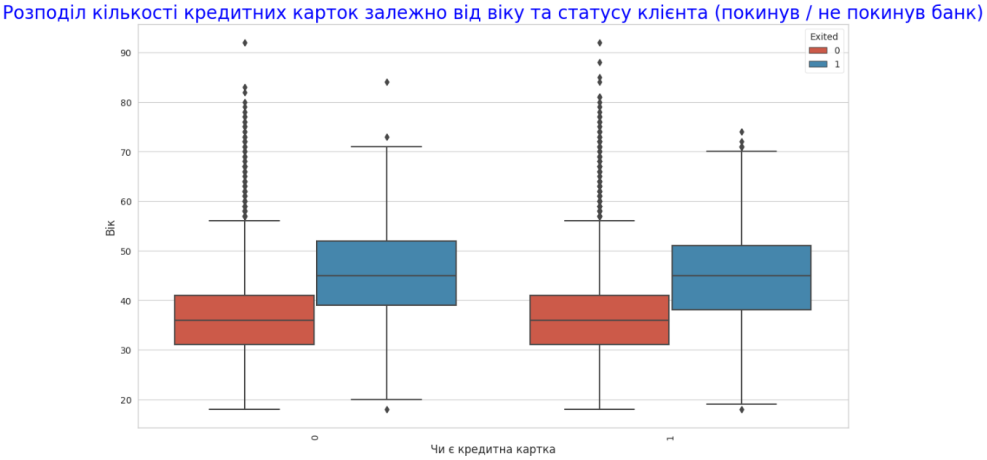


Рисунок 3.24 – Розподіл кількості кредитних карток залежно від віку та статусу клієнта (покинув / не покинув банк)

Під час дослідження відтоку клієнтів з банку виникло питання про вплив віку на цей процес. В результаті аналізу було встановлено, що особи у віковому діапазоні від 45 до 65 років мають найбільшу ймовірність покинути банк.

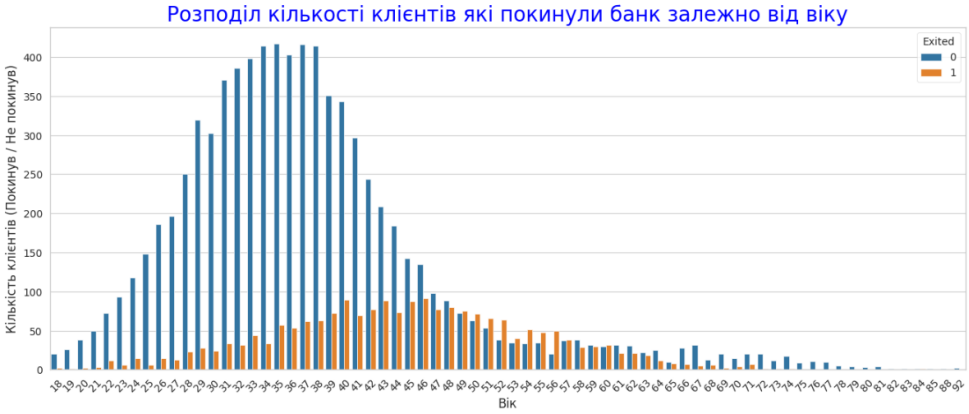


Рисунок 3.25 – Розподіл кількості клієнтів які покинули банк залежно від віку

Також було проведено дослідження, яке включало аналіз матриці кореляції між різними ознаками. Під час аналізу було встановлено, що ознаки "вік" (age) і "вихід з банку" (Exited) мають найвищий рівень кореляції, який становить 0.3. Результати аналізу було проілюстровано на рисунку 3.26.

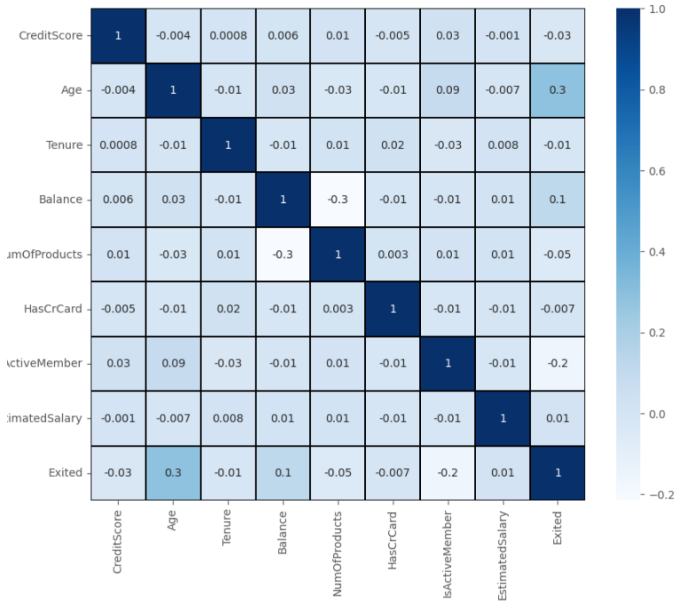


Рисунок 3.26 – Матриця кореляції ознак відтоку клієнтів

Після аналізу настає етап обробки даних, який необхідний для підготовки їх до використання у моделях класифікації. Першим кроком є групування ознак з метою покращення якості даних. В даному наборі даних виявлено дві ознаки - "вік" (age) та "кредитний рейтинг" (Credit Score), які потребують групування. Приклад кодування змінної age проілюстровано на рисунку 3.27.

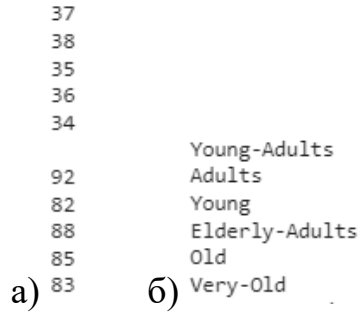


Рисунок 3.27 – а) Початкові значення ознаки age, б) Значення нової ознаки age_group після групування

Наступним етапом обробки даних є кодування категоріальних значень в дискретні. В наявному наборі даних представлено три категоріальні ознаки: Geography, Gender, age_group. Приклад кодування ознаки age_group проілюстровано на рисунку 3.28.

а)

Young-Adults
Adults
Young
Elderly-Adults
Old
Very-Old

б)

5
0
4
1
2
3

Рисунок 3.28 – а) Початкові значення ознаки age_group, б) Значення ознаки age_group після кодування

Наступний крок, приведення ознак до діапазону від 0 до 1. Цей процес поліпшує здатність моделі розпізнавати та узагальнювати структури даних, що дозволяє отримувати більш точні й надійні результати класифікації. Процедура масштабування показана на рисунку 3.29.

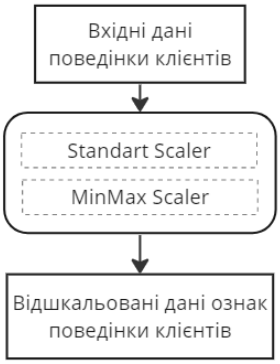


Рисунок 3.29 – Поетапний процес шкалювання ознак поведінки клієнтів

Далі переходимо до зменшення розмірності даних, що дозволяє знизити кількість ознак у наборі даних, зберігаючи максимальну кількість інформації. Процес зменшення розмірності представлений на рисунку 3.30, а ілюстрація методу головних компонент відображена на рисунку 3.31.

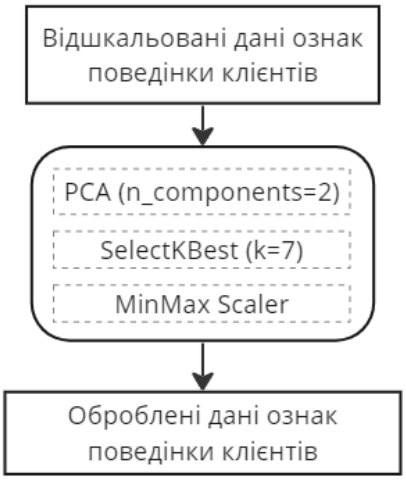


Рисунок 3.30 – Поетапний процес зменшення розмірності ознак поведінки клієнтів

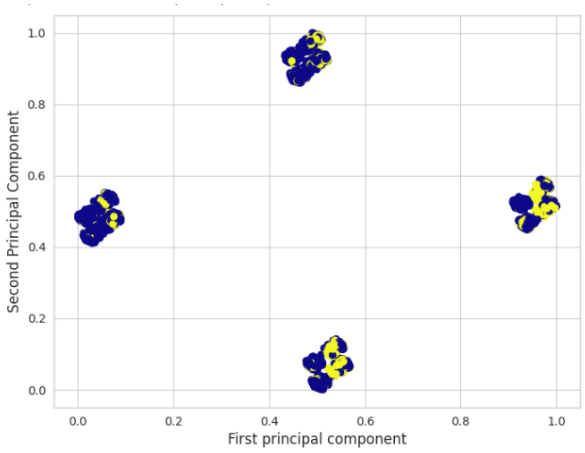


Рисунок 3.31 – Ілюстрація розподілу ознак поведінки клієнтів після зменшення розмірності

Тепер переходимо до моделювання, було застосовано різні методи класифікації та перехресну валідацію для тренування моделей. Першим методом була класичний метод опорних векторів з лінійною функцією ядра. Процес побудови цієї лінійної моделі показаний поетапно на рисунку 3.32.

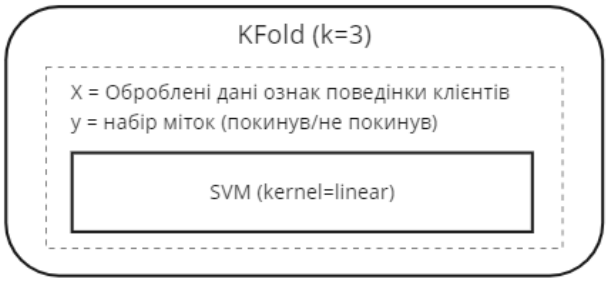


Рисунок 3.32 – Процес побудови лінійної моделі класифікації відтоку клієнтів

Для другого методу було застосовано класичний метод опорних векторів з поліноміальною функцією ядра. Процес побудови цієї поліноміальної моделі наведений поетапно на рисунку 3.33.

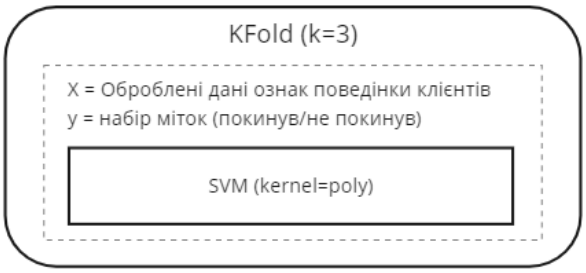


Рисунок 3.33 – Процес побудови поліноміальної моделі класифікації відтоку клієнтів

Для реалізації квантового методу опорних векторів побудували схему оцінки квантового ядра. Ілюстрація квантового ядра для квантового моделі класифікації відтоку клієнтів наведена на рисунку 3.34.

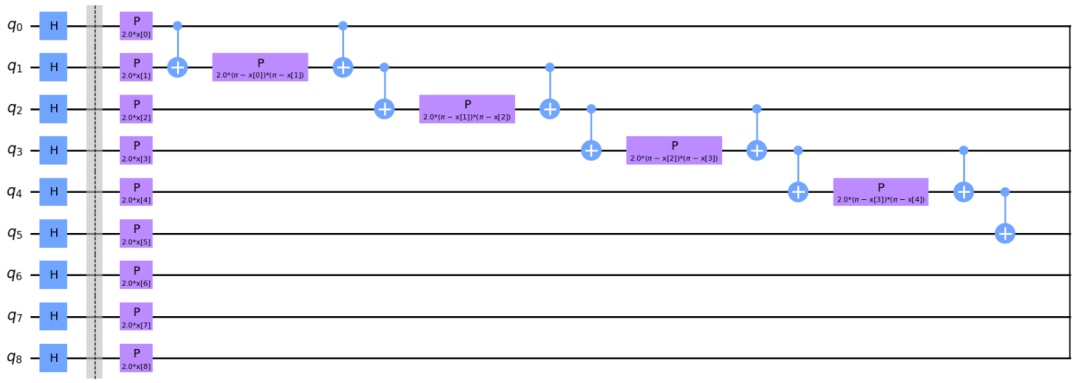


Рисунок 3.34 – Ілюстрація схеми оцінки квантового ядра моделі класифікації відтоку клієнтів

Далі сформували ядро функції, а потім застосовували його в класичному методі опорних векторів. На рисунку 3.35 представлений поетапний процес побудови квантової моделі.

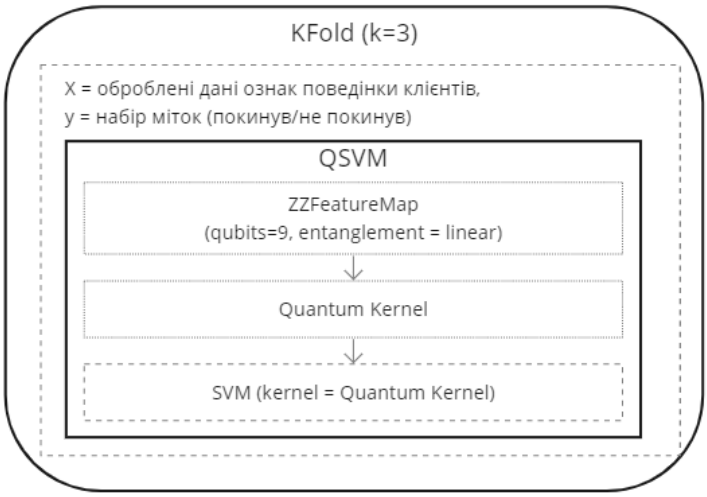


Рисунок 3.35 – Поетапний процес побудови квантової моделі класифікації відтоку клієнтів

Після завершення процесу побудови класифікаторів, було проведено оцінку їх якості. Результати цієї оцінки можна знайти в таблиці 3.6

Таблиця 3.6 – Результати класифікації відтоку клієнтів

Модель	**Точність**	**Влучність**	**Повнота**
Лінійний класифікатор	0.7984	1.000	0.01
Поліноміальний класифікатор	0.840	0.692	0.41
Квантовий класифікатор	0.843	0.695	0.41

3.4 Висновки до розділу 3

Реалізація класичних і квантових моделей класифікації дала нам можливість зробити кілька висновків. Перш за все, для досягнення кращих та швидших

результатів у квантових моделях необхідно мати достатню кількість квантових бітів та використовувати ефективні методи оптимізації моделей.

У випадку класифікації зловмисного трафіку, квантова модель продемонструвала кращу ефективність порівняно з класичними методами, хоча час навчання був трохи більшим через обмежену доступність квантових комп'ютерів. Враховуючи потенціал квантового обчислення для обробки великого обсягу даних та складних аналітичних завдань, його застосування в кібербезпеці може сприяти виявленню та прогнозуванню кібератак, захисту мереж та систем від загроз, а також виявленню вразливостей і встановленню механізмів їх усунення.

У випадку екологічних досліджень, класичні методи та квантовий класифікатор продемонстрували приблизно однакову якість класифікації, але результати не досягали еталонного рівня. Для поліпшення моделей можна використовувати кращі методи класифікації або комбінувати класичні методи з квантовими для отримання більш точних результатів.

У випадку прогнозування відтоку клієнтів у фінансових задачах, квантовий метод опорних векторів перевершив класичні методи, що свідчить про його великий потенціал, який може бути реалізований у якісні результати у майбутньому. Для подальшого розвитку таких застосувань необхідно вдосконалювати алгоритми, збільшувати кількість квантових ресурсів та активно використовувати нові підходи до навчання та оптимізації моделей.

Висновки

Основною метою кваліфікаційної роботи було дослідження особливостей використання квантового методу опорних векторів з варіаційними алгоритмами для класифікації даних у технічних, природничих і соціально-економічних системах з великим об'ємом даних.

Для досягнення цієї мети були визначені конкретні завдання, такі як огляд методів та засобів машинного навчання, огляд квантових обчислень для великої кількості даних, огляд інформаційних систем, що потребують класифікації та мають великі об'єми даних, запропонувати способи реалізації квантового методу опорних векторів для різних систем та провести їх реалізацію.

Об'єктом дослідження став процес використання квантових обчислень і квантового машинного навчання для задач класифікації з використанням методу опорних векторів.

Предметом дослідження були моделі, алгоритми та засоби квантових обчислень для методу опорних векторів, зокрема їх використання у технічних, природничих і соціально-економічних інформаційних системах.

Результати дослідження сприяють розумінню перспектив і можливостей використання квантового методу опорних векторів для класифікації великих об'ємів даних у різних галузях, що може мати значний вплив на розвиток машинного навчання та інформаційних систем.

Перелік посилань

1. Стаття про ріст сфери штучного інтелекту [Електронний ресурс] – Режим доступу https://www.forbes.com/sites/joemckendrick/2018/12/19/how-fast-is-artificial-intelligence-growing-look-at-the-key-bellwethers/?sh=10c7b80c474a

1. Стаття про історію штучного інтелекту та машинного навчання [Електронний ресурс] – Режим доступу https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/

2. Стаття про IBM Watson [Електронний ресурс] – Режим доступу https://www.techrepublic.com/article/ibm-watson-the-inside-story-of-how-the-jeopardy-winning-supercomputer-was-born-and-what-it-wants-to-do-next/

3. Wikipedia AlphaGo [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/AlphaGo

4. Wikipedia Applications of Artificial Intelligence [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Applications_of_artificial_intelligence

5. Wikipedia SVM [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Support_vector_machine

6. SVM Illustrated [Електронний ресурс] – Режим доступу https://www.researchgate.net/figure/Illustration-of-support-vector-machine-SVM-principle-a-Input-space-is-mapped-to-the_fig4_331801455

7. Using SVM with Big Data [Електронний ресурс] – Режим доступу https://www.researchgate.net/publication/353786557_Support_Vector_Machines_in_Big_Data_Classification_A_Systematic_Literature_Review

8. Стаття про використання QSVM на великих обсягах даних [Електронний ресурс] – Режим доступу https://arxiv.org/pdf/1307.0471.pdf

9. Wikipedia Quantum Computing [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Quantum_computing

10. Wikipedia Quantum Superposition [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Quantum_superposition

11. Стаття Pros and Cons of Quantum computing [Електронний ресурс] – Режим доступу https://www.itrelease.com/2020/10/advantages-and-disadvantages-of-quantum-computers/

12. Quantum computer Illustrated [Електронний ресурс] – Режим доступу https://www.carel.com/blog/-/blogs/quantum-computers-and-kid-s-lies

13. Wikipedia Quantum Entanglement [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Quantum_entanglement

14. Wikipedia Кіт Шредінгера [Електронний ресурс] – Режим доступу https://uk.wikipedia.org/wiki/%D0%9A%D1%96%D1%82_%D0%A8%D1%80%D0%B5%D0%B4%D1%96%D0%BD%D0%B3%D0%B5%D1%80%D0%B0

15. Стаття Disadvantages of Quantum computing [Електронний ресурс] – Режим доступу https://bit.ly/3otWYZU

16. Wikipedia Quantum Error Coercion [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Quantum_error_correction

17. Стаття Applications of Quantum Machine learning [Електронний ресурс] – Режим доступу https://www.analyticsinsight.net/what-is-quantum-machine-learning-applications-of-quantum-machine-learning/

18. Quantum SVM in Biology [Електронний ресурс] – Режим доступу https://members.cbio.mines-paristech.fr/~jvert/talks/110401mines/mines.pdf

19. Quantum SVM in Ecology [Електронний ресурс] – Режим доступу https://www.nature.com/articles/s41598-022-14876-6.pdf?origin=ppub

20. Quantum SVM in Finance [Електронний ресурс] – Режим доступу https://arxiv.org/pdf/2202.00599.pdf

21. Quantum SVM in Cryptography [Електронний ресурс] – Режим доступу https://eprint.iacr.org/2021/1058.pdf

22. Лінійні та нелінійні класифікатори [Електронний ресурс] – Режим доступу http://om.univ.kiev.ua/users_upload/15/upload/file/pr_lecture_06.pdf

23. Wikipedia Функції витрат [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Hinge_loss

24. SVM Kernels [Електронний ресурс] – Режим доступу https://www.geeksforgeeks.org/major-kernel-functions-in-support-vector-machine-svm/

25. Most used SVM Kernels [Електронний ресурс] – Режим доступу https://dataaspirant.com/svm-kernels/

26. Variational circuits [Електронний ресурс] – Режим доступу https://pennylane.ai/qml/glossary/variational_circuit

27. QSVM schema [Електронний ресурс] – Режим доступу https://www.researchgate.net/figure/Schematic-quantum-circuit-for-QSVM-where-UPEdocumentclass12ptminimal_fig4_334150381

28. Основні базові квантові гейти [Електронний ресурс] – Режим доступу https://quantum-ods.github.io/qmlcourse/book/qc/ru/gates.html

29. Quantum Logic Gates [Електронний ресурс] – Режим доступу https://upload.wikimedia.org/wikipedia/commons/thumb/e/e0/Quantum_Logic_Gates.png/500px-Quantum_Logic_Gates.png

30. Quantum Kernel Example [Електронний ресурс] – Режим доступу https://medium.com/@patrick.huembeli/introduction-into-quantum-support-vector-machines-727f3ccfa2b4

31. Binary Classification metrics [Електронний ресурс] – Режим доступу https://medium.com/analytics-vidhya/what-nobody-tells-you-about-binary-classification-metrics-4998574b668

32. Performance metrics visualized [Електронний ресурс] – Режим доступу https://www.researchgate.net/figure/Visualizing-accuracy-recall-aka-sensitivity-and-precision-which-are-the-common_fig3_346129022

33. Wikipedia Principal Component Analysis [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Principal_component_analysis

34. Principal Component Analysis visualized [Електронний ресурс] – Режим доступу https://setosa.io/ev/principal-component-analysis/

35. Wikipedia Overfitting [Електронний ресурс] – Режим доступу https://en.wikipedia.org/wiki/Principal_component_analysis

36. Cross Validation [Електронний ресурс] – Режим доступу https://scikit-learn.org/stable/modules/cross_validation.html

37. The Growing Role of Machine Learning in Cybersecurity [Електронний ресурс] – Режим доступу https://www.securityroundtable.org/the-growing-role-of-machine-learning-in-cybersecurity/

38. The Biggest Cybersecurity Issues and Challenges [Електронний ресурс] – Режим доступу https://www.checkpoint.com/cyber-hub/cyber-security/what-is-cybersecurity/biggest-cyber-security-challenges-in-2023/

39. Quantum Machine Learning in the Context of IT Security [Електронний ресурс] - Режим доступу https://bit.ly/3p3tCS8

40. What is DDoS [Електронний ресурс] - Режим доступу https://bit.ly/3LlZklh

41. Malicious DDoS Attacks Rise 150% [Електронний ресурс] - Режим доступу https://bit.ly/42fBbDB

42. Cirq [Електронний ресурс] - Режим доступу https://quantumai.google/cirq

43. Qiskit [Електронний ресурс] - Режим доступу https://qiskit.org/

44. PennyLane [Електронний ресурс] - Режим доступу https://pennylane.ai/

45. Main principles of Ecology [Електронний ресурс] - Режим доступу https://bit.ly/3Nu527l

46. Quantum computing in Ecology [Електронний ресурс] - Режим доступу https://arxiv.org/ftp/arxiv/papers/2108/2108.10855.pdf

47. Disease Impact of Unsafe Water [Електронний ресурс] - Режим доступу https://bit.ly/3LLQIpF

48. Customers churn problem [Електронний ресурс] - Режим доступу https://www.netsuite.com/portal/resource/articles/human-resources/customer-churn-analysis.shtml

49. DDoS SDN dataset [Електронний ресурс] - Режим доступу https://data.mendeley.com/datasets/jxpfjc64kr/1

50. Water quality dataset [Електронний ресурс] - Режим доступу https://www.kaggle.com/datasets/adityakadiwal/water-potability

51. Churn modeling dataset [Електронний ресурс] - Режим доступу https://www.kaggle.com/datasets/shrutimechlearn/churn-modelling

ДОДАТКИ

Додаток А

Програмні коди основних модулів класифікації зловмисного трафіку

```
# General Imports
import numpy as np
from time import time

# Visualisation Imports
import matplotlib.pyplot as plt

# Scikit Imports
from sklearn import datasets
from   sklearn.model_selection   import   cross_val_score,   train_test_split,   KFold,
cross_validate
from sklearn.svm import SVC
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.preprocessing import LabelEncoder
from  sklearn.metrics  import  accuracy_score,  confusion_matrix,  classification_report,
make_scorer, precision_score, recall_score, f1_score

from sklearn.pipeline import Pipeline, FeatureUnion

from sklearn.feature_selection import SelectKBest, chi2
from sklearn.feature_selection import f_classif

# Pandas Importsb
import pandas as pd
pd.set_option('display.max_columns', None)

#Seaborn Imports
import seaborn as sns

#Pickle Imports
import pickle

# Plotly
import plotly.graph_objects as go
import plotly.express as px

# Qiskit Imports
from qiskit import Aer, execute
from qiskit.circuit import QuantumCircuit, Parameter, ParameterVector
from qiskit.visualization import plot_histogram
from qiskit.circuit.library import PauliFeatureMap, ZFeatureMap, ZZFeatureMap
from qiskit.circuit.library import TwoLocal, NLocal, RealAmplitudes, EfficientSU2
from qiskit.circuit.library import HGate, RXGate, RYGate, RZGate, CXGate, CRXGate, CRZGate
from qiskit_machine_learning.kernels import QuantumKernel
from qiskit_machine_learning.algorithms import QSVC
import warnings

warnings.filterwarnings('ignore')

def build_traffic_distribution(data):
  colors = ['rgb(158,202,225)', 'red']
  labels = list(map(lambda x: 'Зловмисный трафік' if x == 1 else "Безпечний трафік",
data['label'].value_counts().index))
  values = data['label'].value_counts().values

  fig = go.Figure(data=[go.Pie(labels=labels, values=values, textinfo='label+percent',
                               insidetextorientation='radial'
                              )])

  fig.update_traces(textfont_size=15, opacity=0.9,
                    marker=dict(colors=colors, line=dict(color='#000000', width=2)))

  fig.show()


def build_traffic_distribution_by_port(data):

  labels = ["Безпечний трафік", "Зловмисный трафік"]

  labels_normal = data[data['label'] == 0]['src'].value_counts().index
```

```
    values_normal = data[data['label'] == 0]['src'].value_counts().values

    labels_malicious = data[data['label'] == 1]['src'].value_counts().index
    values_malicious = data[data['label'] == 1]['src'].value_counts().values

    fig = go.Figure()
    fig.add_trace(go.Bar(
        name=labels[0], x=labels_normal, y=values_normal,
        text=values_normal, marker_color='rgb(158,202,225)'
    ))

    fig.add_trace(go.Bar(
        name=labels[1], x=labels_malicious, y=values_malicious,
        text=values_malicious, marker_color='red'
    ))

    fig.update_traces(marker_line_color='rgb(8,48,107)',
                      marker_line_width=1.5, opacity=0.9,
                      textposition='outside')

    fig.update_xaxes(
      showline=False
    )

    fig.update_yaxes(
        showline=False
    )

    fig.update_layout(
      title='Порти з найбільшою кількістю запитів',
      title_font_size=24,
      xaxis_tickfont_size=14,
      xaxis = {'type' : 'category'},
      yaxis=dict(
          title='Кількість запитів',
          titlefont_size=16,
          tickfont_size=14,
      ),
      legend=dict(
          x=0.85,
          y=1.1,
          bgcolor='rgba(255, 255, 255, 0)',
          bordercolor='rgba(255, 255, 255, 0)'
      ),
      barmode='group',
      bargap=0.15, # gap between bars of adjacent location coordinates.
      bargroupgap=0.1, # gap between bars of the same location coordinate.
      plot_bgcolor='white'
    )

    fig.show()

def build_traffic_distribution_by_protocols(data):

    labels = ["Безпечний трафік", "Зловмисный трафік"]

    labels_normal = data[data['label'] == 0]['Protocol'].value_counts().index
    values_normal = data[data['label'] == 0]['Protocol'].value_counts().values

    labels_malicious = data[data['label'] == 1]['Protocol'].value_counts().index
    values_malicious = data[data['label'] == 1]['Protocol'].value_counts().values

    fig = go.Figure()
    fig.add_trace(go.Bar(
      name=labels[0], x=labels_normal, y=values_normal,
      text=values_normal, marker_color='rgb(158,202,225)'
    ))

    fig.add_trace(go.Bar(
      name=labels[1], x=labels_malicious, y=values_malicious,
      text=values_malicious, marker_color='red'
    ))

    fig.update_traces(marker_line_color='rgb(8,48,107)',
                      marker_line_width=1.5, opacity=0.9,
                      textposition='outside', textfont_size=22)

    fig.update_xaxes(
      showline=False
```

```
    )

    fig.update_yaxes(
      showline=False
    )

    fig.update_layout(
      title='Протоколи з найбільшою кількістю запитів',
      title_font_size=24,
      xaxis_tickfont_size=25,
      xaxis = {'type' : 'category'},
      yaxis=dict(
          title='Кількість запитів',
          titlefont_size=20,
          tickfont_size=20,
      ),
      legend=dict(
          x=0.85,
          y=1.1,
          bgcolor='rgba(255, 255, 255, 0)',
          bordercolor='rgba(255, 255, 255, 0)'
      ),
      barmode='group',
      bargap=0.15, # gap between bars of adjacent location coordinates.
      bargroupgap=0.1, # gap between bars of the same location coordinate.
      plot_bgcolor='white',
      uniformtext_minsize=16
    )

    fig.show()

  def build_traffic_distribution_by_switch(data):
    labels = ["Безпечний трафік", "Зловмисный трафік"]

    labels_normal = data[data['label'] == 0]['switch'].value_counts().index
    values_normal = data[data['label'] == 0]['switch'].value_counts().values

    labels_malicious = data[data['label'] == 1]['switch'].value_counts().index
    values_malicious = data[data['label'] == 1]['switch'].value_counts().values

    fig = go.Figure()
    fig.add_trace(go.Bar(
      name=labels[0], x=labels_normal, y=values_normal,
      text=values_normal, marker_color='rgb(158,202,225)'
    ))

    fig.add_trace(go.Bar(
      name=labels[1], x=labels_malicious, y=values_malicious,
      text=values_malicious, marker_color='red'
    ))

    fig.update_traces(marker_line_color='rgb(8,48,107)',
                      marker_line_width=1.5, opacity=0.9,
                      textposition='outside',  textfont_size=22)

    fig.update_xaxes(
      showline=False
    )

    fig.update_yaxes(
      showline=False
    )

    fig.update_layout(
      title='Коммутатори з найбільшою кількістю запитів',
      title_font_size=24,
      xaxis_tickfont_size=25,
      xaxis = {'type' : 'category'},
      yaxis=dict(
          title='Кількість запитів',
          titlefont_size=20,
          tickfont_size=20,
      ),
      legend=dict(
          x=0.85,
          y=1.1,
          bgcolor='rgba(255, 255, 255, 0)',
          bordercolor='rgba(255, 255, 255, 0)'
      ),
```

```
    barmode='group',
    bargap=0.15, # gap between bars of adjacent location coordinates.
    bargroupgap=0.1, # gap between bars of the same location coordinate.
    plot_bgcolor='white'
  )

  fig.show()

def build_traffic_distribution_by_duration(data):
  labels = ["Безпечний трафік", "Зловмисный трафік"]
  values_normal = data[data['label'] == 0]['dur'].values
  values_malicious = data[data['label'] == 1]['dur'].values

  fig = go.Figure()
  fig.add_trace(go.Histogram(
    name=labels[0],
    x=values_normal,
    marker_color='rgb(158,202,225)',
    nbinsx=40
  ))

  fig.add_trace(go.Histogram(
    name=labels[1],
    x=values_malicious,
    marker_color='red',
    nbinsx=40
  ))

  fig.update_traces(opacity=0.9)

  fig.update_layout(
    title='Розподіл тривалості запитів',
    title_font_size=24,
    xaxis_titlefont_size=25,
    xaxis_tickfont_size=20,
    xaxis_title_text='Тривалість (секунди)',
    yaxis=dict(
        title='Кількість запитів',
        titlefont_size=20,
        tickfont_size=20,
    ),
    legend=dict(
        x=0.85,
        y=1.1,
        bgcolor='rgba(255, 255, 255, 0)',
        bordercolor='rgba(255, 255, 255, 0)'
    ),
    barmode='stack',
    plot_bgcolor='white'
  )

  fig.show()

def build_traffic_features_correlation(data):
  plt.figure(figsize=(20,10))
  sns.heatmap(data.corr(), annot=True)

def prepare_traffic_data(data):
  lb = LabelEncoder()
  data["src"] = lb.fit_transform(data["src"])
  data["dst"] = lb.fit_transform(data["dst"])
  data["Protocol"] = lb.fit_transform(data["Protocol"])

  X = data.drop('label', axis=1)
  y = data['label']

  scaling_pipeline = Pipeline([
    ('standart_scaler', StandardScaler()),
    ('min_max_scaler', MinMaxScaler(feature_range = (0, 1)))
  ])

  pca = PCA(n_components=2)
  select_k_best = SelectKBest(chi2, k=7)

  combined_features = FeatureUnion([("pca", pca), ("select_k_best", select_k_best)])

  X = scaling_pipeline.fit_transform(X)
  X = combined_features.fit(X, y).transform(X)
  X = MinMaxScaler(feature_range = (0, 1)).fit_transform(X)
```

```
  scoring = {'accuracy' : make_scorer(accuracy_score),
             'precision' : make_scorer(precision_score),
             'recall' : make_scorer(recall_score)}

  return X, y

def build_pca_chart(X, y):
  plt.figure(figsize=(8,6))
  plt.scatter(X[:, 0] ,X[:,1],c=y, cmap='plasma')
  plt.xlabel('First principal component')
  plt.ylabel('Second Principal Component')

linear_svc = SVC(kernel='linear')
kfold = KFold(n_splits=3)

linear_svc_results = cross_validate(
    estimator=linear_svc,
    X=X, y=y,
    cv=kfold, scoring=scoring
)

poly_svc = SVC(kernel='poly')
kfold = KFold(n_splits=3)

poly_svc_results = cross_validate(
    estimator=poly_svc,
    X=X, y=y,
    cv=kfold, scoring=scoring
)

zz_map   =   ZZFeatureMap(feature_dimension=X.shape[1],   reps=2,   entanglement='linear',
insert_barriers=True)
zz_kernel                        =                        QuantumKernel(feature_map=zz_map,
quantum_instance=Aer.get_backend('statevector_simulator'))

qsvc_model = QSVC(quantum_kernel=zz_kernel)
qsvc_model.quantum_kernel.quantum_instance = Aer.get_backend('statevector_simulator')

kfold = KFold(n_splits=3)

quantum_svc_results = cross_validate(
    estimator=qsvc_model,
    X=X, y=y,
    cv=kfold, scoring=scoring
)

results_df = pd.DataFrame({
    'model': ['linear_svc', 'poly_svc', 'quantum_svc'],
    'accuracy':                                [linear_svc_results['test_accuracy'].mean(),
poly_svc_results['test_accuracy'].mean(), quantum_svc_results['test_accuracy'].mean()],
    'precision':                               [linear_svc_results['test_precision'].mean(),
poly_svc_results['test_precision'].mean(), quantum_svc_results['test_precision'].mean()],
    'recall':                                  [linear_svc_results['test_recall'].mean(),
poly_svc_results['test_recall'].mean(), quantum_svc_results['test_recall'].mean()]
})
```

Додаток Б

Програмні коди основних модулів класифікації якості води

```
# General Imports
import numpy as np
from time import time

# Visualisation Imports
import matplotlib.pyplot as plt

# Scikit Imports
from sklearn import datasets
from   sklearn.model_selection   import   cross_val_score,   train_test_split,   KFold,
cross_validate
from sklearn.svm import SVC
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.preprocessing import LabelEncoder
from  sklearn.metrics  import  accuracy_score,  confusion_matrix,  classification_report,
make_scorer, precision_score, recall_score, f1_score

from sklearn.pipeline import Pipeline, FeatureUnion

from sklearn.feature_selection import SelectKBest, chi2
from sklearn.feature_selection import f_classif

# Pandas Importsb
import pandas as pd
pd.set_option('display.max_columns', None)

#Seaborn Imports
import seaborn as sns

#Pickle Imports
import pickle

# Plotly
import plotly.graph_objects as go
import plotly.express as px

# Qiskit Imports
from qiskit import Aer, execute
from qiskit.circuit import QuantumCircuit, Parameter, ParameterVector
from qiskit.visualization import plot_histogram
from qiskit.circuit.library import PauliFeatureMap, ZFeatureMap, ZZFeatureMap
from qiskit.circuit.library import TwoLocal, NLocal, RealAmplitudes, EfficientSU2
from qiskit.circuit.library import HGate, RXGate, RYGate, RZGate, CXGate, CRXGate, CRZGate
from qiskit_machine_learning.kernels import QuantumKernel
from qiskit_machine_learning.algorithms import QSVC
import warnings

warnings.filterwarnings('ignore')

def build_water_potabibilty_distribution(data):
  d= pd.DataFrame(data['Potability'].value_counts())
  fig                                                                                =
px.pie(d,values='Potability',names=['Небезпечна','Безпечна'],hole=0.4,opacity=0.6,
             color_discrete_sequence=[colors_green[3],colors_blue[3]],
              labels={'label':'Potability','Potability':'No. Of Samples'})

  fig.update_layout(
    font_family='monospace',
    title=dict(text='Розподіл зразків води',x=0.47,y=0.98,
               font=dict(color=colors_dark[2],size=28)),
    legend=dict(x=0.37,y=-0.05,orientation='h',traceorder='reversed'),
    hoverlabel=dict(bgcolor='white'))

  fig.update_traces(textposition='inside', textinfo='percent+label',textfont_size=24)

  fig.show()

def build_water_features_distribution(data):

  non_potable = data.query("Potability == 0")
```

```
  potable = data.query("Potability == 1")

  plt.figure(figsize = (15,15))
  for ax, col in enumerate(data.columns[:9]):
      plt.subplot(3,3, ax + 1)
      plt.title(col)
      plotting = sns.kdeplot(x = non_potable[col], label = "Non Potable",fill=True,
common_norm=False, color="#E68193",alpha=.9, linewidth=3)
      plotting = sns.kdeplot(x = potable[col], label = "Potable",fill=True,
common_norm=False, color="#459E97",alpha=.9, linewidth=3)
      plt.legend()
  plt.tight_layout()
  plt.title('Distribution')
  plotting.figure.suptitle(' Distribution of features ',y=1.08, size = 26, weight='bold')

def build_waterq_features_correlation(data):
  plt.style.use("ggplot")
  f,ax=plt.subplots(figsize = (10,8))
  sns.heatmap(data.corr(), robust=True, fmt='.1g', linewidths=1.3, linecolor = 'black',
cmap="viridis", annot=True)

def preparing_water_data(data):
  data['ph'].fillna(value=data['ph'].median(),inplace=True)
  data['Sulfate'].fillna(value=data['Sulfate'].median(),inplace=True)
  data['Trihalomethanes'].fillna(value=data['Trihalomethanes'].median(),inplace=True)

  X = data.drop('Potability', axis=1)
  y = data['Potability']

  scaling_pipeline = Pipeline([
      ('standart_scaler', StandardScaler()),
      ('min_max_scaler', MinMaxScaler(feature_range = (0, 1)))
  ])

  pca = PCA(n_components=2)
  select_k_best = SelectKBest(chi2, k=7)

  combined_features = FeatureUnion([("pca", pca), ("select_k_best", select_k_best)])

  X = scaling_pipeline.fit_transform(X)
  X = combined_features.fit(X, y).transform(X)
  X = MinMaxScaler(feature_range = (0, 1)).fit_transform(X)

  return X, y

X, y = preparing_water_data(data)
scoring = {'accuracy' : make_scorer(accuracy_score),
           'precision' : make_scorer(precision_score),
           'recall' : make_scorer(recall_score)}

def build_pca_chart(X, y):
  plt.figure(figsize=(8,6))
  plt.scatter(X[:, 0] ,X[:,1], c=y, cmap='plasma')
  plt.xlabel('First principal component')
  plt.ylabel('Second Principal Component')

linear_svc = SVC(kernel='linear')
kfold = KFold(n_splits=3)

linear_svc_results = cross_validate(
    estimator=linear_svc,
    X=X, y=y,
    cv=kfold, scoring=scoring
)

poly_svc = SVC(kernel='poly')
kfold = KFold(n_splits=3)

poly_svc_results = cross_validate(
    estimator=poly_svc,
    X=X, y=y,
    cv=kfold, scoring=scoring
)

zz_map = ZZFeatureMap(feature_dimension=X.shape[1], reps=2, entanglement='linear',
insert_barriers=True)
zz_kernel = QuantumKernel(feature_map=zz_map,
quantum_instance=Aer.get_backend('statevector_simulator'))
```

```
qsvc_model = QSVC(quantum_kernel=zz_kernel)
qsvc_model.quantum_kernel.quantum_instance = Aer.get_backend('statevector_simulator')

kfold = KFold(n_splits=3)

quantum_svc_results = cross_validate(
    estimator=qsvc_model,
    X=X, y=y,
    cv=kfold, scoring=scoring
)

results_df = pd.DataFrame({
    'model': ['linear_svc', 'poly_svc', 'quantum_svc'],
    'accuracy':                                  [linear_svc_results['test_accuracy'].mean(),
poly_svc_results['test_accuracy'].mean(), quantum_svc_results['test_accuracy'].mean()],
    'precision':                                [linear_svc_results['test_precision'].mean(),
poly_svc_results['test_precision'].mean(), quantum_svc_results['test_precision'].mean()],
    'recall':                                     [linear_svc_results['test_recall'].mean(),
poly_svc_results['test_recall'].mean(), quantum_svc_results['test_recall'].mean()]
})
```

Додаток В

Програмні коди основних модулів класифікації відтоку клієнтів

```
# General Imports
import numpy as np
from time import time

# Visualisation Imports
import matplotlib.pyplot as plt

# Scikit Imports
from sklearn import datasets
from sklearn.model_selection import cross_val_score, train_test_split, KFold,
cross_validate
from sklearn.svm import SVC
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report,
make_scorer, precision_score, recall_score, f1_score

from sklearn.pipeline import Pipeline, FeatureUnion

from sklearn.feature_selection import SelectKBest, chi2
from sklearn.feature_selection import f_classif

# Pandas Importsb
import pandas as pd
pd.set_option('display.max_columns', None)

#Seaborn Imports
import seaborn as sns

#Pickle Imports
import pickle

# Plotly
import plotly.graph_objects as go
import plotly.express as px

# Qiskit Imports
from qiskit import Aer, execute
from qiskit.circuit import QuantumCircuit, Parameter, ParameterVector
from qiskit.visualization import plot_histogram
from qiskit.circuit.library import PauliFeatureMap, ZFeatureMap, ZZFeatureMap
from qiskit.circuit.library import TwoLocal, NLocal, RealAmplitudes, EfficientSU2
from qiskit.circuit.library import HGate, RXGate, RYGate, RZGate, CXGate, CRXGate, CRZGate
from qiskit_machine_learning.kernels import QuantumKernel
from qiskit_machine_learning.algorithms import QSVC
import warnings

warnings.filterwarnings('ignore')

def build_card_distribution_by_age(data):
  sns.set_style("whitegrid")

  plt.figure(figsize = (14,8))
  plt.xticks(rotation=90)
  plt.title('Розподіл кількості кредитних карток за віком', color = 'blue', fontsize=20)
  sns.countplot(x=data["Age"], hue = 'HasCrCard',data=data)
  plt.xlabel('Вік')
  plt.ylabel('Кількість кредитних карток')

def build_card_distribution_by_status(data):
  sns.set_style("whitegrid")

  plt.figure(figsize = (14,8))
  plt.xticks(rotation=90)
  plt.title('Розподіл кількості кредитних карток залежно від віку та статусу клієнта
(покинув / не покинув банк)', color = 'blue', fontsize=20)
  sns.boxplot(data=data, x="HasCrCard", y="Age", hue = 'Exited')
  plt.xlabel('Чи є кредитна картка')
  plt.ylabel('Вік')

def build_age_distribution_by_score(data):
  sns.set_style("whitegrid")
```

```
  plt.figure(figsize = (16,6))
  plt.xticks(rotation=45)
  sns.scatterplot(x=data['Age'], y = data["CreditScore"], hue = "Exited",data=data)

def build_age_distribution_by_exit_status(data):
  plt.figure(figsize = (16,6))
  plt.xticks(rotation=45)
  plt.title('Розподіл кількості клієнтів які покинули банк залежно від віку', color =
'blue', fontsize=20)
  sns.countplot(x=data["Age"], hue = 'Exited', data=data, palette="tab10")
  plt.xlabel('Вік')
  plt.ylabel('Кількість клієнтів (Покинув / Не покинув)')

def build_client_churn_features_correlation(data):
  plt.style.use("ggplot")
  f,ax=plt.subplots(figsize = (10,8))
  sns.heatmap(data.corr(), robust=True, fmt='.1g', linewidths=1.3, linecolor = 'black',
cmap="Blues", annot=True)

def create_age_bin(age):
  if age < 30:
    return 'Young'
  elif age >=30 and age < 40:
    return 'Young-Adults'
  elif age >=40 and age < 50:
    return 'Adults'
  elif age >=50 and age < 60:
    return 'Elderly-Adults'
  elif age >=60 and age < 74:
    return 'Old'
  else:
    return 'Very-Old'

def prepare_data(data):
  data['age_group'] = data['Age'].apply(create_age_bin)
  data['credit_score_group'] = data['CreditScore'].apply(lambda x: 0 if x < 405 else 1)

  lb = LabelEncoder()
  data["Gender"] = lb.fit_transform(data["Gender"])
  data["Geography"] = lb.fit_transform(data["Geography"])
  data["age_group"] = lb.fit_transform(data["age_group"])

  return data.drop(['Age',"CreditScore"], axis=1)

data = prepare_data(data)

X = data.drop('Exited', axis=1)
y = data['Exited']

scaling_pipeline = Pipeline([
    ('standart_scaler', StandardScaler()),
    ('min_max_scaler', MinMaxScaler(feature_range = (0, 1)))
])

pca = PCA(n_components=2)
select_k_best = SelectKBest(chi2, k=7)

combined_features = FeatureUnion([("pca", pca), ("select_k_best", select_k_best)])

X = scaling_pipeline.fit_transform(X)
X = combined_features.fit(X, y).transform(X)
X = MinMaxScaler(feature_range = (0, 1)).fit_transform(X)

scoring = {'accuracy' : make_scorer(accuracy_score),
           'precision' : make_scorer(precision_score),
           'recall' : make_scorer(recall_score)}

def build_pca_chart(X, y):
  plt.figure(figsize=(8,6))
  plt.scatter(X[:, 0] ,X[:,1], c=y, cmap='plasma')
  plt.xlabel('First principal component')
  plt.ylabel('Second Principal Component')

linear_svc = SVC(kernel='linear')
kfold = KFold(n_splits=3)

linear_svc_results = cross_validate(
    estimator=linear_svc,
```

```
    X=X, y=y,
    cv=kfold, scoring=scoring
)

poly_svc = SVC(kernel='poly')
kfold = KFold(n_splits=3)

poly_svc_results = cross_validate(
    estimator=poly_svc,
    X=X, y=y,
    cv=kfold, scoring=scoring
)

zz_map = ZZFeatureMap(feature_dimension=X.shape[1], reps=2, entanglement='linear',
insert_barriers=True)
zz_kernel = QuantumKernel(feature_map=zz_map,
quantum_instance=Aer.get_backend('statevector_simulator'))

qsvc_model = QSVC(quantum_kernel=zz_kernel)
qsvc_model.quantum_kernel.quantum_instance = Aer.get_backend('statevector_simulator')

kfold = KFold(n_splits=3)

quantum_svc_results = cross_validate(
    estimator=qsvc_model,
    X=X, y=y,
    cv=kfold, scoring=scoring
)

results_df = pd.DataFrame({
    'model': ['linear_svc', 'poly_svc', 'quantum_svc'],
    'accuracy': [linear_svc_results['test_accuracy'].mean(),
poly_svc_results['test_accuracy'].mean(), quantum_svc_results['test_accuracy'].mean()],
    'precision': [linear_svc_results['test_precision'].mean(),
poly_svc_results['test_precision'].mean(), quantum_svc_results['test_precision'].mean()],
    'recall': [linear_svc_results['test_recall'].mean(),
poly_svc_results['test_recall'].mean(), quantum_svc_results['test_recall'].mean()]
})
```

Додаток Г

Презентаційний матеріал

Квантовий метод опорних векторів

розробка варіаційних алгоритмів для реалізації класифікації даних у технічних, природничих і соціально-економічних системах

Виконав: студент групи КН-19-1 Казіонов М.А.

Керівник кваліфікаційної роботи бакалавра: д.т.н., професор, завідувач кафедри КН Бармак Олександр Володимирович

Машинне навчання та квантові обчислення – це дві технології, кожна з яких може змінити спосіб виконання обчислень, щоб вирішити проблеми, які раніше були недосяжними.

Проте, збільшення обсягу даних та складність обчислення ядер створюють обмеження для успішного вирішення завдань.

Саме тому, квантові методи машинного навчання можуть змінити парадигму обчислень і внести значний вклад у наукові дослідження, медицину, фінанси та інші сфери.

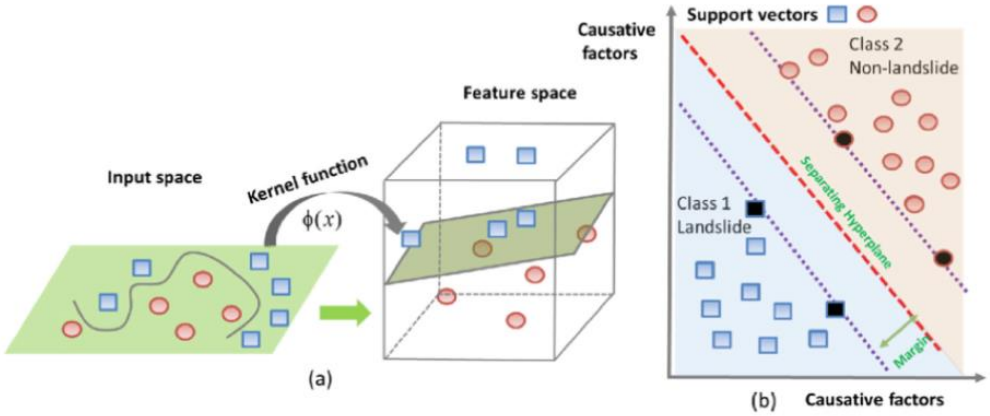
Мета і завдання роботи

Мета кваліфікаційної роботи полягає в дослідженні особливостей використання квантового методу опорних векторів для реалізації класифікації даних у технічних, природничих і соціально-економічних системах з великим об'ємом даних.

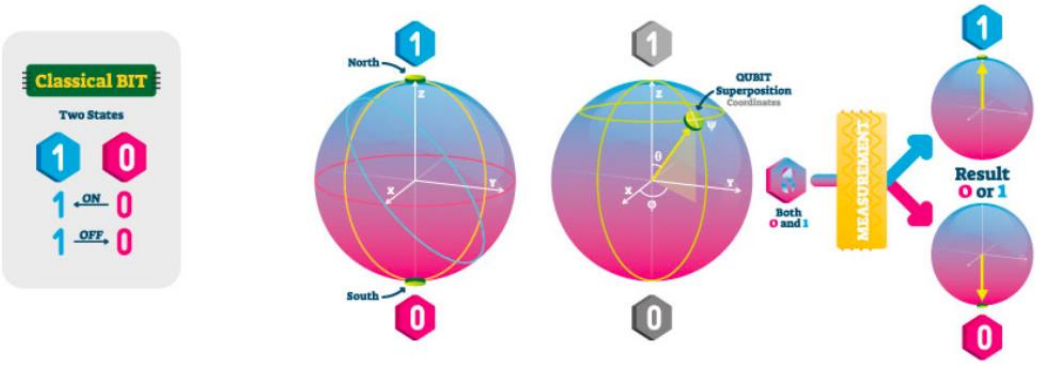
Ключові завдання дослідження:

- огляд методів та засобів машинного навчання;
- огляд квантових обчислень для великої кількості даних;
- огляд систем, що потребують класифікації та мають великі об'єми даних;
- реалізації квантового методу опорних векторів для технічних, природничих і соціально-економічних інформаційних систем.

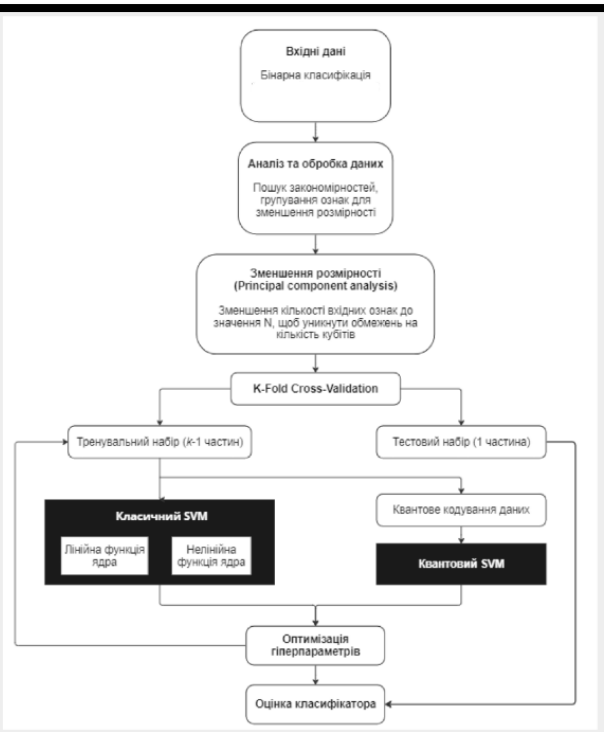
Принцип роботи методу опорних векторів



Принцип роботи квантового комп`ютера



Реалізація
квантового методу опорних
векторів для задач бінарної
класифікації



Результати класифікації зловмисного трафіку

Модель	Точність	Влучність	Повнота
Лінійний класифікатор	0.7162	0.6409	0.5458
Поліноміальний класифікатор	0.8974	0.8796	0.8410
Квантовий класифікатор	0.9255	0.8972	0.9043

Результати класифікації якості води

Модель	Точність	Влучність	Повнота
Лінійний класифікатор	0.6098	0.0000	0.0000
Поліноміальний класифікатор	0.5952	0.4844	0.1280
Квантовий класифікатор	0.5863	0.4452	0.2029

Результати класифікації відтоку клієнтів

Модель	Точність	Влучність	Повнота
Лінійний класифікатор	0.7984	1.000	0.01
Поліноміальний класифікатор	0.840	0.692	0.41
Квантовий класифікатор	0.843	0.695	0.41

Висновки

Реалізація класичних і квантових моделей класифікації дала нам можливість зробити кілька висновків:

- У випадку класифікації зловмисного трафіку, квантова модель продемонструвала кращу ефективність порівняно з класичними методами;
- У випадку екологічних досліджень, класичні методи та квантовий класифікатор продемонстрували приблизно однакову якість класифікації, але результати не досягали еталонного рівня.
- У випадку прогнозування відтоку клієнтів у фінансових задачах, квантовий метод опорних векторів перевершив класичні методи, що свідчить про його великий потенціал, який може бути реалізований у якісні результати у майбутньому.
- Для досягнення кращих результатів у квантових моделях необхідно мати достатню кількість квантових бітів та використовувати ефективні методи оптимізації моделей.

Fri May 26 08:30:23 EEST 2023, Петровський Сергій Степанович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 0.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 11%**

ID: 114054 Назва: КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА Додано в БД: 2023-05-26 Автора: М.А. Казіонов Керівники: О.В.Бармак Консультанти: Опоненти:	Документ		Сумарний збіг по Базі Даних	
	Символи	Лексеми	Символи	Лексеми
	73779	1104	742 (1%)	12 (1%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

UNICHECK

Ім'я користувача:
Кафедра КН

ID перевірки:
1015264838

Дата перевірки:
26.05.2023 09:27:50 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
26.05.2023 09:34:22 EEST

ID користувача:
100005671

Назва документа: **КН-19-1 Казіонов**

Кількість сторінок: **72** Кількість слів: **12155** Кількість символів: **88091** Розмір файлу: **2.72 MB** ID файлу: **1014939129**

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

3.69%
Схожість

Найбільша схожість: **1.08%** з джерелом з Бібліотеки (ID файлу: **1011230523**)

3.54% Джерела з Інтернету	505	Сторінка 74
1.73% Джерела з Бібліотеки	167	Сторінка 77

0% Цитат

Не знайдено жодних цитат

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи	6
Підозріле форматування	19 сторінок

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ КАФЕДРИ КОМП'ЮТЕРНИХ НАУК
ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:
Назва: *Квантовий метод опорних векторів: розробка варіаційних алгоритмів для реалізації класифікації даних у технічних, природничих і соціально-економічних системах*

Автор: *студент групи КН-19-1 Казіонов Максим Андрійович*
Спеціальність: *122 – Комп'ютерні науки*
Освітня програма: *освітньо-професійна*
Науковий керівник: *д.т.н., проф. Бармак О.В.*
Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	***відповідає***
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі Казіонова М.А., не є плагіатом, оскільки: запозичення розміщені в розділі огляду існуючих підходів, не описують безпосередньо авторську роботу і не стосуються її результатів; усі запозичення фрагментарні; до запозичень входять фрагменти поширених конструкцій мови та загальновідомі терміни та скорочення.

Обсяг запозичень, визначений системами виявлення збігів/ ідентичності/схожості, складає:

- за системою Anti-Plagiarism: 0%;
- за системою Unicheck: 3,69%.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ідентичності/ схожості, складає 0% і 3,69% відповідно, що, з урахуванням наведених обґрунтувань, відповідає характеру дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ *Олександр БАРМАК*

Гарант ОП _____ *Олександр МАЗУРЕЦЬ*

Завідувач кафедри КН _____ *Олександр БАРМАК*

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МОН УКРАЇНИ

Кафедра комп'ютерних наук

ВІДГУК НАУКОВОГО КЕРІВНИКА

на кваліфікаційну роботу бакалавра

студента *гр. КН-19-1 Казіонова Максима Андрійовича*

за темою *Квантовий метод опорних векторів: розробка варіаційних алгоритмів для реалізвції класифікації даних у технічних, природничих та соціально-економічних системах*

1. Актуальність теми

Актуальним завданням, яке потребує аналізу і досліджується у даній роботі, є застосування квантового методу опорних векторів у різних сферах, зокрема в технічних, природничих і соціально-економічних системах. Враховуючи потенціал квантового обчислення для обробки великого обсягу даних та складних аналітичних завдань, його застосування в кібербезпеці може сприяти виявленню та прогнозуванню кібератак, захисту мереж та систем від загроз, а також виявленню вразливостей і встановленню механізмів їх усунення. Розробка такого метода є актуальною задачею комп'ютерних наук.

2. Відповідність роботи предметній області Стандарту спеціальності 122 Комп'ютерні науки

За стандартом, а саме описом предметної області, об'єктами вивчення та діяльності є математичні, інформаційні, імітаційні моделі реальних явищ, об'єктів, систем і процесів та методи і технології отримання, зберігання, обробки, передачі та використання інформації. Метою роботи саме є розробка квантового методу опорних векторів у різних сферах, зокрема в технічних, природничих і соціально-економічних системах. При вирішенні поставленої задачі використано математичні моделі, методи та алгоритми розв'язання теоретичних і прикладних задач, що виникають при розробці інформаційних технологій. Тому результати виконання кваліфікаційної роботи бакалавра відповідають стандарту бакалавра спеціальності 122 – Комп'ютерні науки.

3. Професійні та особистісні якості бакалавра

При роботі над кваліфікаційною роботою бакалавра Казіонов Максим Андрійович проявила себе кваліфікованим фахівцем та дисциплінованим студентом, вчасно виконуючи поставлені етапи дослідження. Як в процесі написання пояснювальної записки, так і при розробці прикладного програмного забезпечення проявив достатні для одержання успішного результату компетентності та результати навчання. Опанував професійні скіли за напрямком «Комп'ютерні науки» та достатньо значний софт скіл .

4. Ступінь самостійності під час виконання кваліфікаційної роботи

Одержані в роботі результати є наслідком особистої діяльності студента, який самостійно виконував всі поставлені задачі.

5. Ступінь оволодіння методами дослідження

При реалізації кваліфікаційної роботи показав достатній рівень компетентностей та володіння необхідними інструментами та обладнанням, методами, методиками та технологіями предметної області комп'ютерних наук.

6. Повнота та якість розкриття теми роботи

Тема роботи в повній мірі обґрунтована й розкрита, проведено аналіз актуальності та відомих досліджень в межах обраної теми, поставлені завдання, які у роботі виконані, та розроблено програмне забезпечення для валідації та верифікації запропонованого метода.

7. Логічність, послідовність, аргументованість, літературна грамотність викладення матеріалу

Структура роботи та послідовність викладення логічні та відповідають поставленій меті. Викладення матеріалу послідовне, аргументоване, літературно грамотне.

8. Можливість практичного застосування кваліфікаційної роботи бакалавра, окремих її частин

Розроблений у роботі метод та його програмна реалізація може бути використана в кібербезпеці та може сприяти виявленню та прогнозуванню кібератак, захисту мереж та систем від загроз, а також виявленню вразливостей і встановленню механізмів їх усунення.

9. Висновок про можливість допуску кваліфікаційної роботи бакалавра до захисту, на яку оцінку заслуговує робота

Враховуючи високий рівень виконання та забезпечення усіх необхідних вимог, робота може бути допущена до захисту. Рекомендована оцінка « Відмінно *».*

Керівник _____ *д.т.н., проф. зав. каф. КН Олександр БАРМАК*

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МОН УКРАЇНИ

Кафедра комп'ютерних наук

РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра

студента *гр. КН-19-1 Казіонова Максима Андрійовича*

за темою *Квантовий метод опорних векторів: розробка варіаційних алгоритмів для реалізвції класифікації даних у технічних, природничих та соціально-економічних системах*

1. Актуальність обраної теми

Актуальним завданням, яке потребує аналізу і досліджується у даній роботі, є застосування квантового методу опорних векторів у різних сферах, зокрема в технічних, природничих і соціально-економічних системах. Враховуючи потенціал квантового обчислення для обробки великого обсягу даних та складних аналітичних завдань, його застосування в кібербезпеці може сприяти виявленню та прогнозуванню кібератак, захисту мереж та систем від загроз, а також виявленню вразливостей і встановленню механізмів їх усунення. Розробка такого метода є актуальною задачею комп'ютерних наук.

2. Повнота розкриття мети та завдань роботи

Тема роботи в повній мірі обґрунтована й розкрита, проведено аналіз актуальності та відомих досліджень в межах обраної теми, поставлені завдання, які у роботі виконані, та розроблено програмне забезпечення для валідації та верифікації запропонованого метода.

3. Зміст кожного розділу роботи

Для висвітлення способів застосування квантового підходу у технічних інформаційних системах було використано проблему класифікації зловмисного трафіку, для природничих систем, проблему класифікації якості води, для економічних систем, проблему класифікації відтоку клієнтів.

4. Оцінка розробленої інформаційної системи, її практична цінність

Розроблений у роботі метод та його програмна реалізація може бути використана в кібербезпеці та може сприяти виявленню та прогнозуванню кібератак, захисту мереж та систем від загроз, а також виявленню вразливостей і встановленню механізмів їх усунення.

5. Якість оформлення кваліфікаційної роботи бакалавра

Структура роботи та послідовність викладення логічні та відповідають поставленій меті. Викладення матеріалу послідовне, аргументоване, літературно грамотне.

6. Недоліки кваліфікаційної роботи бакалавра

Недоліків в даній кваліфікаційній роботі немає

7. Загальний висновок (допускається чи не допускається до захисту), та оцінка на яку заслуговує кваліфікаційна робота.

Враховуючи рівень виконання та забезпечення усіх необхідних вимог, робота може бути допущена до захисту. Рекомендована оцінка «відмінно».

Рецензент _____ доц., к.ф.-м.н. Наталія Дудчик