НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий фізико-технічний інститут
Кафедра математичного моделювання та аналізу даних

До захисту допущено:
В.о. завідувача кафедри
Г.О. Яйлимова
« __»_____ 2024р.

Дипломна робота
на здобуття ступеня бакалавра
зі спеціальності 113 «Прикладна Математика»
на тему: «Методи машинного навчання для класифікації інфікованих пацієнтів»

Виконав:
студент 4 курсу, групи ФІ-01
Щербина Іван Володимирович _____

Керівник:
ст. викладач кафедри ММАД ННФТІ, д-р філософії
Яйлимова Г.О. _____

Рецензент:
к.т.н. кафедри ПФ ННФТІ, доцент
Гордійко Н.О. _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань
Студент _____

Київ – 2024 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий фізико-технічний інститут
Кафедра математичного моделювання та аналізу даних

Рівень вищої освіти – перший (бакалаврський)
Спеціальність: 113 Прикладна математика,
Освітня програма: Математичні методи моделювання, розпізнавання образів та Комп'ютерного зору

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____**Г.О. Яйлимова**
«___»_____2024 р.

ЗАВДАННЯ
на дипломну роботу студенту
Щербині Івану Володимировичу

1. Тема роботи: «Методи машинного навчання для класифікації інфікованих пацієнтів»,
   науковий керівник роботи ст. викладач кафедри ММАД ННФТІ, д-р філософії Яйлимова Ганна Олексіївна,
   затверджені наказом по університету № 2251-С від «31» травня 2024 р.
2. Термін подання студентом роботи: «07» червня 2024 р.
3. Вихідні дані до роботи: дані з платформи kaggle за 1996 р. щодо інфікувань синдромом набутого імунодефіциту.
4. Зміст роботи: аналіз методів, що використовуються для класифікації інфікованих пацієнтів, реалізація та оцінка якості моделей, вибір найефективнішого методу.
5. Перелік ілюстративного матеріалу: презентація доповіді.
6. Дата видачі завдання: «8» лютого 2024 р.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання	Примітка
1	Узгодження теми роботи із науковим керівником	08.02.2024	Виконано
2	Пошук літератури по інтелектуальному аналізу	20.02.2024	Виконано
3	Пошук джерел по тематиці диплому	08.03.2024	Виконано
4	Пошук множини даних	25.03.2024	Виконано
5	Первинний аналіз даних	08.04.2024	Виконано
6	Імплементація методу головних компонент	26.04.2024	Виконано
7	Побудова методів машинного навчання для класифікації інфікувань	24.05.2024	Виконано
8	Оцінка якості моделей	30.05.2024	Виконано
9	Оформлення дипломної роботи	14.05.2024	Виконано

Студент _____ Іван ЩЕРБИНА

Науковий керівник роботи _____ Ганна ЯЙЛИМОВА

РЕФЕРАТ

Дипломна робота містить: 55 с., 11 табл., 9 рис., 4 дод., 12 джерел.

Об’єкт дослідження – дані про інфікування пацієнтів синдромом набутого імунодефіциту.

Мета дослідження – застосування методів машинного навчання для виявлення, аналізу та класифікації інфікування пацієнтів.

Методи дослідження – було проведено підготовку даних для подальшої побудови методів машинного навчання. Очищення даних від пропущених значень, нормалізацію та масштабування даних, а також за допомогою метода головних компонент проведено візуалізацію стовпців з множини даних, які містять більше всього інформації.

Було застосовано методи машинного навчання: логістичну регресію, ядерну логістичну регресію, опорних векторів та багатошаровий персептрон. Проведено експерименти для налаштування гіперпараметрів моделей та визначення оптимальних умов для проведення класифікації інфікувань.

За допомогою різних метрик для оцінки якості класифікації моделі, було визначено найоптимальнішу модель, для проведення класифікації інфікувань. Тож в подальшому буде можливо розробити застосунок, для того, щоб користувачі вільно могли дізнатися, чи є вони інфікованими за допомогою проведеної класифікації найоптимальнішою моделлю, зі зберіганням повної анонімності особистості.

МЕТОДИ МАШИННОГО НАВЧАННЯ, ПЕРВИННИЙ АНАЛІЗ, МОДЕЛІ КЛАСИФІКАЦІЇ, КЛАСИФІКАЦІЯ ІНФІКОВАНИХ ПАЦІЄНТІВ.

ABSTRACT

The diploma thesis comprises: 55 pages, 11 tables, 9 figures, 4 appendix, 12 sources.

The object of the research is data on patients infected with acquired immunodeficiency syndrome.

The aim of the study is to apply machine learning methods for the detection, analysis, and classification of patient infections.

Research methods – data preparation was carried out for further construction of machine learning methods. Data cleansing from missing values, normalization, and data scaling, as well as visualization of columns from the dataset containing the most information using the principal component method, were performed.

Machine learning methods were applied: logistic regression, kernel logistic regression, support vector machines, and multilayer perceptron. Experiments were conducted to tune the hyperparameters of the models and determine the optimal conditions for conducting infection classification.

Using various metrics to evaluate the quality of classification models, the most optimal model for conducting infection classification was determined. Therefore, it will be possible to develop an application so that users can freely determine whether they are infected through the classification conducted by the most optimal model, while maintaining complete anonymity of their identity.

MACHINE LEARNING METHODS, EXPLANATORY DATA ANALYSIS, CLASSIFICATION MODELS, CLASSIFICATION OF INFECTED PATIENTS.

ЗМІСТ

ВСТУП

Актуальність дослідження. Синдром набутого імунодефіциту є однією з важливих медичних проблем в сучасному світі. Він спричиняється вірусом імунодефіциту людини (ВІЛ), який вражає імунну систему, роблячи організм вразливим до різних інфекцій та хвороб. Також щороку реєструється значна кількість нових випадків інфікування. Тож розробка системи класифікації, яка може точно визначити, чи хворіє пацієнт на СНІД, є важливою, оскільки вчасна діагностика дозволить пацієнтам розпочати антиретровірусну терапію ще на ранніх стадіях захворювання, що значно полегшить стан пацієнтів. Система класифікації зможе виявити захворювання ще до появи очевидних симптомів. Також ця система допоможе виявити можливих інфікованих осіб, з подальшою діагностикою пацієнтів. Особи, які знають про інфікування, частіше вживають методи захисту для своїх партнерів. Система класифікації зможе зменшити навантаження на медичний персонал, що є перевагою, оскільки вони зможуть зосередити більше уваги на догляді та лікуванні пацієнтів.

Люди бояться йти до лікарів, щоб дізнатись про інфікування, та ще й під час війни не все населення має доступ, щоб зробити тест. Застосування методів машинного навчання для виявлення та аналізу інфікування СНІДом є надзвичайно актуальним і перспективним напрямом у медичній діагностиці. Це може значно покращити своєчасність і доступність діагностики, особливо в умовах воєнних конфліктів і кризових ситуацій, та сприяти зниженню страху перед діагнозом серед населення. Таким чином, впровадження таких технологій має великий потенціал для збереження здоров’я і життя людей.

Мета дослідження. Застосування методів машинного навчання для виявлення, аналізу та класифікації інфікування пацієнтів.

Завдання дослідження. Проаналізувати існуючі методи класифікації, розібратися з математичною постановкою даних методів. Дослідити фактори, які впливають на інфікування. Провести первинний аналіз даних для розуміння розподілу даних, числових характеристик та можливих аномалій в множині даних.

Очистити пропущені значення, провести нормалізацію даних для підготовки до застосування моделей. Розділити множину даних на тренувальний та тестові набори. Тренувати моделі на тестовому наборі даних. Знайти найбільш ефективні гіперпараметри моделей для покращеної точності класифікації. Провести тестування на тестовому наборі даних, та порівняти результати з оцінкою на тренувальному наборі. Порівняти результати моделей між собою та вибрати найбільш ефективні. Проаналізувати вплив різних ознак на результати класифікації. Сформулювати висновки щодо застосованих методів машинного навчання для класифікації інфікування.

Об’єкт дослідження. Дані про інфікування пацієнтів синдромом набутого імунодефіциту.

Предмет дослідження. Розробка методів машинного навчання для визначення за ознаками чи є пацієнт інфікованим.

Практичне застосування. Розроблені моделі можуть бути інструментом для систематичної класифікації, також це допоможе встановити взаємозв’язок між різними факторами та інфікуванням.

1 АНАЛІЗ КЛАСИЧНИХ ТА СУЧАСНИХ МЕТОДІВ КЛАСИФІКАЦІЇ ІНФІКУВАНЬ

1.1 Огляд літератури на тему методи машинного навчання для класифікації пацієнтів. Сучасні методи класифікації

У нашому світі є проблема з класифікацією інфікування. Ми будемо вирішувати цю проблему методами машинного навчання.

У книзі [1] надано базові знання з теорії ймовірностей. Особливу увагу було приділено до випадкових величин, основних розподілів випадкових величин, щоб сформувати міцний “місток” до математичної статистики.

Оцінка точності та продуктивності моделі, детально описана в книзі [2].

У книзі [3] описано взагалі значну та об’ємну частину методів машинного навчання, які будуть використовуватись для класифікації пацієнтів.

У статті [4] описується використання методів машинного навчання для прогнозування факторів, які спричиняють гіпертонію у літніх пацієнтів з СНІДом за допомогою електронних датчиків. Дослідження проводилось на пацієнтах, госпіталізованих у Ханаанській лікарні інфекційних захворювань. Було побудовано п’ять методів машинного навчання, а саме: логістична регресія, опорних векторів, нейронна мережа із зворотним розповсюдженням помилки, дерево прийняття рішень та екстремальне градієнте підсилювання. Найкращі результати показав метод екстремального градієнтного підсилювання. Також було застосовано метод головних компонент для візуалізації та зменшення розмірності множини вхідних даних. В результаті дослідження були виявлені ознаки, які впливають на гіпертонію в пацієнтів, які інфіковані синдромом набутого імунодефіциту людини.

У статті [5] досліджується важливість використання алгоритмів машинного навчання для визначення та передбачення ознак, які допоможуть у діагностиці та лікуванні різних захворювань, в тому числі СНІДу. Було використано такі методи машинного навчання: головних компонентів, логістичної регресії, дерев прийняття

рішень, нейронна мережа, k-найближчих сусідів та випадкового лісу. Дані були зібрані на основі 270 інфікованих пацієнтів. У множині даних було аж 104 ознаки, які вдалося зменшити до 23 ознак за допомогою методу головних компонент, це сприяло покращенню як в цілому роботи алгоритмів так і їхню точність. Найкращим алгоритмом виявився метод випадкового лісу з 86% точності. Логістична регресія показала найменший результат, всього 56%, що свідчить про нелінійне розбиття множини даних за залежною змінною. Нейронна мережа показала 59% точності, що є трохи кращим результатом у порівнянні з логістичною регресією. Метод k-найближчих сусідів показав результат на 1% кращий за нейронну мережу. А ось дерево прийняття рішень показало 68% точності, що є досить непоганим результатом у порівнянні з вище розглянутими методами, окрім випадкового лісу. Також автори роблять висновок, що методи машинного навчання можуть суттєво покращити діагностику у пацієнтів інфікованих синдромом набутого імунодефіциту людини. Вони також зазначають, що можна використати ці методи на більших множинах даних для покращення їхньої продуктивності, так як автори використовували в якості множини даних вибірку досить маленького розміру.

У статті [6] описано прогнозування ознак, які пов’язані з синдромом набутого імунодефіциту людини. Також досліджується розщеплення вірусу, вхід вірусу до організму та побічні ефекти антиретровірусної терапії. У статті були використані методи машинного навчання, а саме: нейронні мережі, опорних векторів та випадкові ліси. За допомогою цих алгоритмів вдалося спрогнозувати різні аспекти синдрому набутого імунодефіциту людини та навіть встановлено відповідне лікування, яке допоможе інфікованим пацієнтам. Тож методи машинного навчання допомогли покращити всебічне розуміння СНІДу та полегшили розробку лікування та профілактику для інфікованих пацієнтів.

На основі цих досліджень було обрано такі методи машинного навчання: логістичної регресії, її модифікацію- ядерну логістичну регресію, опорних векторів та нейронну мережу з прямим поширенням, а саме багатошаровий персептрон.

1.2 Методи машинного навчання

1.2.1 Метод головних компонент

Метод головних компонент зменшує розмірність множини даних, що дозволяє їх візуалізувати. Також дані з меншими розмірностями легше опрацювати.

Головні змінні– нові змінні, які побудовані як лінійні комбінації, створені таким чином, щоб головні компоненти були некорельованими, а більша частина інформації була стиснута в першому компоненті. З геометричної точки зору, головні компоненти представляють напрямок, який пояснює максимальну дисперсію в даних, тобто лінію, яка містить найбільшу інформацію в даних.

Першим кроком є стандартизація множини даних, оскільки метод головних компонент є досить чутливим до варіацій вхідних змінних. Тобто, якщо присутні дані з більшим діапазоном, то вони будуть домінувати над змінними з меншим діапазоном. Це може призвести до зміщеного результату. Стандартизація– це процес віднімання від змінних їхніх математичних сподівань та ділення цієї різниці на стандартне відхилення. Формула для стандартизації множини даних наведена в [7]:

$$z_i = \frac{x_i - \mu}{\sigma},$$

де

x_i – вхідні змінні;

μ – математичне сподівання;

σ – стандартне відхилення.

Другим кроком є встановлення того як змінні вхідного набору даних відхиляються від середнього значення. Для того щоб визначити такі взаємозв’язки потрібно обчислити коваріаційну матрицю. Згідно [1] коваріаційна матриця– це симетрична матриця, елементами якої є коваріації всіх вхідних змінних. Формула для коваріаційної матриці згідно [1]:

$$Cov(X,Y) = \frac{1}{N}\sum_{i=1}^{N}(X_i - \mu_x)\big(Y_i - \mu_y\big),$$

де

$Cov(X,Y)$ – коваріаційна матриця для вхідних параметрів та залежних змінних;

N – кількість вхідних параметрів;

μ_x, μ_y – математичні сподівання вхідних параметрів та залежних змінних відповідно.

Третім кроком є визначення власних векторів та значень матриці коваріацій. Власні вектори є фактично напрямками осей коваріаційної матриці, де є найбільша дисперсія, тож, як наслідок, в них міститься найбільше інформації. Також власні вектори матриці коваріацій називаються головними компонентами, а власні значення матриці потрібні для обрахунку цих компонент. Формула для обрахунку з [7] має вигляд:

$$Cov(X,Y)\boldsymbol{v} = \alpha\boldsymbol{v},$$

де

$\boldsymbol{v}$ – власний вектор;

α – відповідне власне значення вектора $\boldsymbol{v}$.

Записавши власні вектори за їхніми власними значеннями в порядку зростання, отримаємо головні компоненти в порядку значущості.

Четвертим кроком буде вибір головних компонент по числу значущості, тобто це і є зменшення розмірності вхідних параметрів. З вибраних головних компонент формують вектор ознак.

П’ятим та останнім кроком буде перерозподіл даних вздовж усіх головних компонент. Сенс в тому, щоб використати вектор ознак для переорієнтації даних з вихідних осей на ті, що представлені головними компонентами. З [7] у формульному вигляді це виглядає так:

$$X_i^{new} = f_i z_i,$$

де

X_i^{new} – переорієнтовані вхідні дані;

f_i – компоненти вектора ознак.

1.2.2 Логістична регресія

У своєму дослідженні я обрав модель логістичної регресії, тому що її можна досить добре використовувати для бінарної класифікації. Модель логістичної регресії допомагає моделювати апостеріорні ймовірності класів за допомогою лінійних функцій по x і гарантує, що сума цих функцій дорівнює одиниці і належить інтервалу [0, 1].

Згідно [3] модель має вигляд:

$$\log\frac{P(G=1 \mid X=x)}{P(G=K \mid X=x)} = \beta_{10} + \beta_1^T x,$$

$$\log\frac{P(G=2 \mid X=x)}{P(G=K \mid X=x)} = \beta_{20} + \beta_2^T x,$$

$$\log\frac{P(G=K-1 \mid X=x)}{P(G=K \mid X=x)} = \beta_{(K-1)0} + \beta_{K-1}^T x,$$

де

$G(x)$ – класифікатор;

K – кількість класів;

β – вагові коефіцієнти, які потрібно знайти для прогнозування значення залежної змінної.

Модель задається в термінах $K-1$ логарифмів відношення [3]:

$$P(G=k|X=x) = \frac{\exp(\beta_{m0} + \beta_m^T x)}{1 + \sum_{i=1}^{m}\exp(\beta_{i0} + \beta_i^T x)},$$

$$P(G=k|X=x) = \frac{1}{1 + \sum_{i=1}^{m}\exp(\beta_{i0} + \beta_i^T x)},$$

де $m = 1, 2, \ldots, K-1$.

Множина параметрів моделі має вигляд:

$$\theta = \{\beta_{10}, \beta_1^T, \ldots, \beta_{(K-1)0}, \beta_{K-1}^T\}$$

При задачі бінарної класифікації $K = 2$, множина параметрів моделі набуває вигляд:

$$\theta = \{\beta_{10}, \beta_1^T\}$$

Навчання моделі логістичної регресії проходить за методом максимальної правдоподібності з використанням умовної правдоподібності G при відомих значеннях X. Ймовірність $P(G = K|X = x)$ повністю задає умовний розподіл.

Логарифмічна функція правдоподібності для N спостережень задається формулою [3]:

$$l(\theta) = \sum_{i=1}^{N} logP(G = 1|X = x_i; \theta)\,.$$

Нехай $y_i = 1$, при $k = 1$ та $y_i = 0$, при $k = 1$, тоді:

$$P(G = 1|\, X = x;\ \theta) + P(G = 2|\, X = x;\ \theta) = 1.$$

Звідси маємо, що:

$$P(G = 2|\, X = x;\ \theta) = 1 - P(G = 1|\, X = x;\ \theta).$$

Тепер логарифмічну функцію правдоподібності можна записати у вигляді:

$$l(\theta) = \sum_{i=1}^{N} y_i\ \log P(G = 1 \mid X =\ x_i;\ \theta) +$$

$$+ \sum_{i=1}^{N} (1 -\ y_i)\ \log(\ 1 - P(G = 1 \mid X =\ x_i;\ \theta))\,,$$

$$P(G = 1 \mid X =\ x_i;\ \theta) = \frac{\exp(\theta)}{1 + \exp(\theta)},$$

$$P(G = 2 \mid X =\ x_i;\ \theta) =\ 1 - P(G = 1 \mid X =\ x_i;\ \theta) =\ 1 - \frac{\exp(\theta)}{1 + \exp(\theta)}$$

Тоді:

$$P(G = 2 \mid X =\ x_i;\ \theta) = \frac{1}{1 +\ \exp(\theta)},$$

$$\log P(G = 1 \mid X =\ x_i;\ \theta) =\ \log \frac{\exp(\theta)}{1+\exp(\theta)} =\ \log \exp(\theta) - \log(\ 1 + \exp(\theta)\ ),$$

$$\log P(G = 1 \mid X =\ x_i;\ \theta) = \theta - \log(\ 1 + \exp(\theta)\ ),$$

$$\log P(G = 2 \mid X = x_i;\ \theta) = \log \frac{1}{1 + \exp(\theta)} = \log 1 - \log(1 + \exp(\theta)),$$

$$\log P(G = 2 \mid X = x_i;\ \theta) = -\log(1 + \exp(\theta))$$

Підставимо отримані формули логарифмів у функцію максимальної правдоподібності:

$$l(\theta) = \sum_{i=1}^{N} y_i[\theta - \log(1 + \exp(\theta))] + \sum_{i=1}^{N} (1 - y_i)\left[-\log\left(1 + exp(\theta)\right)\right] =$$

$$= \sum_{i=1}^{N} y_i\theta - y_i \log(1 + \exp(\theta)) - \log(1 + \exp(\theta)) + y_i \log(1 + \exp(\theta))$$

Звідси маємо:

$$l(\theta) = \sum_{i=1}^{N} y_i\theta - \log(1 + \exp(\theta)),$$

де $\theta = \beta^T x_i$ та $\beta = \{\beta_{10}, \beta_1\}$

$$l(\beta) = \sum_{i=1}^{N} y_i\, \beta^T x_i - \log(1 + \exp(\beta^T x_i))$$

Ми припускаємо, що вектор змінних x_i включає в себе постійний член рівний одиниці для урахування зсуву. Щоб максимізувати функцію правдоподібності прирівняємо її похідні до нуля. В результаті отримаємо систему оціночних рівнянь [3]:

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{N} y_i\, x_i - \frac{1}{1 + \exp(\beta^T x_i)} \exp(\beta^T x_i)\, x_i,$$

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{N} y_i\, x_i - P(G = 1 \mid X = x_i;\ \beta)\, x_i = \sum_{i=1}^{N} x_i[y_i - P(G = 1 \mid X = x_i;\ \beta)]$$

Маємо систему оціночних рівнянь:

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{N} x_i[y_i - P(G = 1 \mid X = x_i;\ \beta)] = 0$$

Ця система складається з $p + 1$ рівняння нелінійного по β.

При першому значенні $x_i = 1$, маємо вигляд:

$$\sum_{i=1}^{N} y_i - P(G = 1 \mid X = x_i;\ \beta) = 0,$$

$$\sum_{i=1}^{N} y_i = \sum_{i=1}^{N} P(G = 1 \mid X = x_i;\ \beta)$$

Звідси робимо висновок, що очікувана кількість класів співпадає з спостерігаємою кількістю пар класів.

Для розв'язання системи оціночних рівнянь використовується алгоритм Ньютона-Рафсона [3]. Для якого необхідно знати матрицю Гессе:

$$\frac{\partial l(\beta)}{\partial \beta} = \frac{\partial l(\beta^{old})}{\partial \beta} + (\beta - \beta^{old}) \frac{\partial^2 l(\beta^{old})}{\partial \beta \partial \beta^T},$$

де β^{old}– попереднє значення вектора β.

$$\frac{\partial l(\beta)}{\partial \beta} = 0 \Rightarrow \frac{\partial l(\beta^{old})}{\partial \beta} + (\beta - \beta^{old}) \frac{\partial^2 l(\beta^{old})}{\partial \beta \partial \beta^T} = 0,$$

$$\beta = \beta^{old} - \left(\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T}\right)^{-1} \frac{\partial l(\beta)}{\partial \beta},$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = \sum_{i=1}^{N} (x_i y_i - x_i P(G = 1 \mid X = x_i;\ \beta))'_{\beta^T},$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = \sum_{i=1}^{N} \left(-x_i \frac{\exp(\beta^T x_i)}{1 + \exp(\beta^T x_i)}\right)'_{\beta^T},$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} x_i \frac{{x_i}^T \exp(\beta^T x_i)\,[1 + \exp(\beta^T x_i)] - {x_i}^T \exp(\beta^T x_i) \exp(\beta^T x_i)}{[1 + \exp(\beta^T x_i)]^2},$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} x_i {x_i}^T \frac{\exp(\beta^T x_i) + [\exp(\beta^T x_i)]^2 - [\exp(\beta^T x_i)]^2}{[1 + \exp(\beta^T x_i)]^2},$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} x_i {x_i}^T \frac{\exp(\beta^T x_i)}{1 + \exp(\beta^T x_i)} \frac{1}{1 + \exp(\beta^T x_i)},$$

На основі того, що

$$P(G = 1 | X = x_i; \beta) = \frac{\exp(\beta^T x_i)}{1 + \exp(\beta^T x_i)},$$

$$P(G = 2|X = x_i; \beta) = \frac{1}{1 + \exp(\beta^T x_i)}$$

Маємо кінцеву формулу для матриці Гессе:

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} x_i {x_i}^T P(G = 1|X = x_i; \beta) P(G = 2|X = x_i; \beta)$$

Починаючи з наступного значення вектора β, окремий крок метода Ньютона має вигляд [3]:

$$\beta^{new} = \beta^{old} - \left(\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T}\right)^{-1} \frac{\partial l(\beta)}{\partial \beta},$$

де

β^{new} – наступне значення вектора β;

похідні беруться по вектору β^{old}.

Тоді рівняння відповідно [3] мають наступний вигляд:

$$\frac{\partial l(\beta)}{\partial \beta} = \mathbf{X}^T(\mathbf{y} - \hat{\mathbf{y}}),$$

$$\frac{\partial^2 l(\beta)}{\partial \beta \partial \beta^T} = -\mathbf{X}^T \mathbf{W} \mathbf{X}$$

Крок в методі Ньютона відповідно [3] має вигляд:

$$\beta^{new} = \beta^{old} - (-\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T (\mathbf{y} - \hat{\mathbf{y}}) = \beta^{old} + (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T (\mathbf{y} - \hat{\mathbf{y}}),$$

де

$\mathbf{y}$ – вектор значень y_i;

$\mathbf{X}$ – матриця N × (p + 1) значень x_i;

$\hat{\mathbf{y}}$ – вектор апроксимованих умовних ймовірностей;

$\mathbf{W}$ – діагональна матриця вагів N × N з i-м діагональним елементом.

За допомогою методу Ньютона-Рафсона отримуємо $\widehat{\beta}$ − оцінку параметра β. Тепер ми можемо скористатись цією оцінкою, щоб знайти прогнозовані значення залежної змінної за наступними формулами:

$$\hat{y}_i = \hat{P}\left(G = 1 \middle| X = x_i; \widehat{\beta}\right) = \frac{exp\left([\widehat{\beta}]^T x_i\right)}{1 + exp\left([\widehat{\beta}]^T x_i\right)},$$

$$1 - \hat{y}_i = \hat{P}\left(G = 2 \middle| X = x_i; \hat{\beta}\right) = \frac{1}{1 + exp\left([\widehat{\beta}]^T x_i\right)}$$

Правило класифікації виглядає наступним чином:

$$\begin{cases} \hat{y}_i = 1, \text{якщо } \hat{y}_i \geq 0{,}5 \\ 0, \text{в іншому випадку} \end{cases}$$

Однак модель логістичної регресії є лінійним класифікатором, тому вона не зможе класифікувати лінійно нероздільні дані. Для класифікації лінійно нероздільних даних було використано її модифікацію – ядерну логістичну регресію.

1.2.3 Ядерна логістична регресія

Класична логістична регресія не підходить, якщо дані лінійно нероздільні. Щоб розв'язати цю проблему застосовують логістичну регресію з ядрами. Відповідно до [8] векторний простір описується як лінійна комбінація вхідних векторів:

$$\beta = \sum_{i=1}^{m} a_i \varphi(x_i),$$

де

$\boldsymbol{a} = (a_1, \dots, a_m)$ – дуальний вектор;

функція $\varphi(x_i)$ відображає дані з меншої розмірності в більшу, де дані можливо буде лінійно розділити.

Відображення φ приймає вигляд [8]:

$$\varphi: \boldsymbol{x} \in \mathbb{R}^{\boldsymbol{m}} \to \varphi(\boldsymbol{x}) \in \mathbb{R}^{\boldsymbol{p}}$$

Ядро приймає наступний вигляд:

$$K\left(x_i, x_j\right) = \langle \varphi(x_i), \varphi\left(x_j\right) \rangle,$$

$$\beta\varphi(\boldsymbol{x}) = \boldsymbol{a}\langle \varphi(\boldsymbol{x}_i), \varphi\left(\boldsymbol{x}_j\right) \rangle = \boldsymbol{a}K(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

де

$K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ (m, m)– матриця, що характеризує собою ядро;

$\boldsymbol{a}$ (m, 1)– дуальний вектор.

Тоді, згідно з роботою [8], маємо формулу для умовної ймовірності:

$$P(Y=1|X)=\frac{\exp(\beta\varphi(\boldsymbol{x}))}{1+\exp(\beta\varphi(\boldsymbol{x}))}=\frac{\exp\left(\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)}{1+\exp\left(\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)},$$

$$P(Y=0|X)=\frac{1}{1+\exp(\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j))}$$

Ядерна логістична регресія дуже схильна до перенавчання, тому вона потребує регуляризації. Функція максимальної правдоподібності з регуляризацією набуває вигляду [8]:

$$L=\frac{\prod_{i,j=1}^{m}P(Y=1|X)^{y_i}P(Y=0|X)^{(1-y_i)}}{\exp\left(\frac{\alpha}{2}\boldsymbol{a}^{\boldsymbol{T}}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\boldsymbol{a}\right)},$$

$$l=\log L=\sum_{i,j=1}^{m}y_i\log P(Y=1|X)+(1-y_i)\log P(Y=0|X)-\frac{\alpha}{2}\boldsymbol{a}^{\boldsymbol{T}}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\boldsymbol{a},$$

$$l=\sum_{i,j=1}^{m}y_i\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)-y_i\log\left(1+\exp\left(\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)\right)-$$

$$-(1-y_i)\log\left(1+exp\left(\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)\right)-\frac{\alpha}{2}\boldsymbol{a}^{\boldsymbol{T}}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\boldsymbol{a},$$

$$l=\sum_{i,j=1}^{m}y_i\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)-\log\left(1+\exp\left(\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)\right)-\frac{\alpha}{2}\boldsymbol{a}^{\boldsymbol{T}}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\boldsymbol{a},$$

$$l=\boldsymbol{y}\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)-\log\left(1+\exp\left(\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)\right)-\frac{\alpha}{2}\boldsymbol{a}^{\boldsymbol{T}}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\boldsymbol{a}$$

Перепишемо функцію максимальної правдоподібності в термінах задачі мінімізації, щоб потім скористатись методом найшвидшого спуску для визначення таких $\boldsymbol{a}$, які мінімізують її.

$$-l=-\boldsymbol{y}\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)+\log\left(1+\exp\left(\boldsymbol{a}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)\right)+\frac{\alpha}{2}\boldsymbol{a}^{\boldsymbol{T}}K(\boldsymbol{x}_i,\boldsymbol{x}_j)\boldsymbol{a}$$

Щоб максимізувати функцію правдоподібності, використаємо метод найшвидшого спуску:

$$\boldsymbol{a}^{\boldsymbol{new}}=\boldsymbol{a}^{\boldsymbol{old}}-\delta\boldsymbol{\nabla}\boldsymbol{l},$$

де

$\boldsymbol{a}^{\boldsymbol{new}}$– значення вектора $\boldsymbol{a}$ на наступній ітерації;

$\boldsymbol{a^{old}}$– значення вектора $\boldsymbol{a}$ на поточній ітерації;

δ – коефіцієнт навчання, $\delta \in \mathbb{R}$ та приймає значення [0, 1].

$$\boldsymbol{A} = \boldsymbol{a}K(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

$$\boldsymbol{\nabla l} = \boldsymbol{y}K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \frac{(\exp(\boldsymbol{A}))}{(1 + \exp(\boldsymbol{A}))}K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \alpha K(\boldsymbol{x}_i, \boldsymbol{x}_j)\boldsymbol{a}$$

Користуючись, тим що

$$P(Y = 1|X) = \frac{(\exp(\boldsymbol{A}))}{(1 + \exp(\boldsymbol{A}))}$$

Маємо, що градієнт функції максимальної правдоподібності приймає вигляд:

$$\boldsymbol{\nabla l} = \boldsymbol{y}K(\boldsymbol{x}_i, \boldsymbol{x}_j) - P(Y = 1|X)K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \alpha K(\boldsymbol{x}_i, \boldsymbol{x}_j)\boldsymbol{a},$$

$$\boldsymbol{\nabla l} = K(\boldsymbol{x}_i, \boldsymbol{x}_j)\big(\boldsymbol{y} - P(Y = 1|X)\big) - \alpha K(\boldsymbol{x}_i, \boldsymbol{x}_j)\boldsymbol{a},$$

$$\boldsymbol{a^{new}} = \boldsymbol{a^{old}} - \delta K(\boldsymbol{x}_i, \boldsymbol{x}_j)\big(\boldsymbol{y} - P(Y = 1|X)\big) + \alpha K(\boldsymbol{x}_i, \boldsymbol{x}_j)\boldsymbol{a}$$

Звідси отримаємо оцінку, що мінімізує функцію максимальної правдоподібності методом найшвидшого спуску: $\widehat{\boldsymbol{a}}$. Отже, маємо:

$$\hat{y}_i = \hat{P}(Y = 1|X) = \frac{\exp(\widehat{\boldsymbol{a}}K(\boldsymbol{x}_i, \boldsymbol{x}_j))}{1 + \exp(\widehat{\boldsymbol{a}}K(\boldsymbol{x}_i, \boldsymbol{x}_j))}$$

Правило класифікації виглядає наступним чином:

$$\begin{cases} \hat{y}_i = 1, \text{якщо } \hat{y}_i \geq 0{,}5 \\ 0, \text{в іншому випадку} \end{cases}$$

Відповідно до [8] маємо ядра, які використовуються в ядерній логістичній регресії:

1) Лінійне ядро: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x_i x^T_j}$.

2) Поліноміальне ядро: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \big(\boldsymbol{x_i x^T_j} + c\big)^d$, де $c \geq 0$ та d степінь полінома.

3) Радіально базисна функція: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = exp\left(-\gamma\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2\right)$, де $\gamma = \frac{1}{2\sigma^2}$.

4) Ядро Лапласа: $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = exp\big(-\gamma\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|\big)$, де $\gamma = \frac{1}{2\sigma^2}$.

1.2.4 Метод опорних векторів для проведення класифікації

Гіперплощина з [3] має вигляд:

$$\{\boldsymbol{x}: \boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{\beta} + \boldsymbol{\beta_0} = \boldsymbol{0}\},$$

де β – це одиничний вектор: $\|\beta\|$=1.

Формула відстані від довільної точки x до гіперплощини $f(x)$ має вигляд:

$$\boldsymbol{d}\big(\boldsymbol{x}, \boldsymbol{f}(\boldsymbol{x})\big) = \frac{\left|\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{\beta} + \boldsymbol{\beta_0}\right|}{\|\boldsymbol{\beta}\|^{\boldsymbol{2}}} = \left|\boldsymbol{x}^{\boldsymbol{T}}\boldsymbol{\beta} + \boldsymbol{\beta_0}\right| = |\boldsymbol{f}(\boldsymbol{x})|$$

Тобто функція $f(x)$ є відстанню від точки x до гіперплощини зі знаком.

$$f(x) = x^T\beta + \beta_0 = 0$$

Так як класи є роздільними, то ми можемо знайти функцію

$$f(x) = x^T\beta + \beta_0,$$

де $y_i f(x) > 0$ для $\forall i$.

Тоді у нас з'явиться можливість знайти гіперплощину, яка створить найбільший проміжок між даними для класів -1 та 1.

Цю концепцію визначає задача оптимізації відповідно до [3]:

$$\max_{\beta, \beta_0, \|\beta\|=1} M, \tag{1.1}$$

де $M = \frac{1}{\|\beta\|}$, за умовою, що $y_i f(x) = y_i(x^T\beta + \beta_0) \geq M$, для $i = 1, \dots, N$.

Нехай класи перекриваються у просторі. Щоб впоратись за перекриттям, потрібно максимізувати M, треба допустити, щоб деякі точки знаходились на неправильній стороні проміжку. Введемо фіктивні змінні $\xi = (\xi_1, \dots, \xi_N)$. Тоді змінимо обмеження в (1.1) на:

$$y_i(x^T\beta + \beta_0) \geq M(1 - \xi_i), \ \ \forall i, \xi_i \geq 0, \sum_{i=1}^{N} \xi_i \leq C = const$$

Значення ξ_i в обмеженні $y_i(x^T\beta + \beta_0) \geq M(1 - \xi_i)$ є величиною, пропорційною відстані, на якій прогноз $f(x_i) = x_i^T\beta + \beta_0$ знаходиться на неправильній стороні свого проміжку. В результаті, обмежуючи суму ξ_i, обмежуємо загальну пропорціональну величину, яка характеризує собою сумарну відстань, на якій прогнози виявляються на неправильній стороні свого проміжку.

Помилкова класифікація є, коли $\xi_i > 1$, тому обмеження $\sum \xi_i$ константою C, обмежує загальну кількість помилкових класифікацій навчання величиною C.

Ми можемо зняти обмеження на норму вектора, як це показано в [3]: β: $\|\beta\|$=1, використовуючи те, що $M = \frac{1}{\|\beta\|}$, напишемо:

$$\min_{\beta,\beta_0} \|\beta\|,$$
$$y_i(x^T\beta + \beta_0) \geq 1 - \xi_i,$$
$$\xi_i \geq 0,$$
$$\sum_{i=1}^{N} \xi_i \leq C. \quad (1.2)$$

Задача (1.2) є квадратичною з обмеженнями у вигляді лінійних нерівностей, тому вона є опуклою задачею оптимізації. Опишемо рішення для квадратичного програмування з використанням множників Лагранжа. Тому виразимо (1.2.) в еквівалентній формі [3]:

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N} \xi_i,$$
$$y_i(x^T\beta + \beta_0) \geq 1 - \xi_i,$$
$$\xi_i \geq 0 \text{ для } \forall i$$

Розглянемо загальний випадок задачі нелінійного програмування [9]:

$$f(x) \to min,$$
$$g_i(x) \leq 0,$$
$$h_j(x) = 0$$

Тоді функція Лагранжа має вигляд

$$L(x, a, b) = f(x) + \sum_{i=1}^{m} a_i g_i(x) + \sum_{j=1}^{p} b_j h_j(x)$$

Теорема Каруша-Куна-Такера про необхідні умови розв'язання задачі нелінійного програмування. Обмеження виглядають таким чином:

$$\begin{cases} \dfrac{\partial L(x,a,b)}{\partial x} = 0 \\ \dfrac{\partial L(x,a,b)}{\partial b} = 0 \\ a_i 0 \\ a_i > 0 \\ a_i g_i(x) = 0 \\ g_i(x) < 0 \end{cases}$$

Враховуючи вище зазначене функція Лагранжа має вигляд:

$$L(\beta, \beta_0, \xi_i) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N} a_i\left[y_i\left(x_i^T\beta + \beta_0\right) - (1-\xi_i)\right] - \sum_{i=1}^{N} b_i\xi_i$$

Маємо розв'язати систему:

$$\begin{cases} \dfrac{\partial L(\beta,\beta_0,\xi_i)}{\partial \beta} = \beta - \sum_{i=1}^{N} a_i y_i x_i^T = 0 \\ \dfrac{\partial L(\beta,\beta_0,\xi_i)}{\partial \beta_0} = -\sum_{i=1}^{N} a_i\beta_0 = 0 \\ \dfrac{\partial L(\beta,\beta_0,\xi_i)}{\partial \xi_i} = C - a_i - b_i = 0 \end{cases}$$

Маємо:

$$\begin{cases} \beta = \sum_{i=1}^{N} a_i y_i x_i^T \\ \sum_{i=1}^{N} a_i\beta_0 = 0 \\ a_i = C + b_i \end{cases}$$

Підставимо вищезазначену систему в функцію Лагранжа:

$$L(\beta, \beta_0, \xi_i) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N} a_i\left[y_i x_i^T \sum_{i'=1}^{N} a_{i'} y_{i'} x_{i'} - (1-\xi_i)\right] - \sum_{i=1}^{N} b_i\,\xi_i,$$

$$L(\beta, \beta_0, \xi_i) = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\sum_{i'=1}^{N} a_i a_{i'} y_i y_{i'} x_i^T x_{i'} +$$

$$+ \sum_{i=1}^{N} a_i - \sum_{i=1}^{N} a_i\xi_i - \sum_{i=1}^{N} b_i\xi_i,$$

$$L(\beta,\beta_0,\xi_i)=\frac{1}{2}\|\beta\|^2+C\sum_{i=1}^{N}\xi_i-\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}x_i^T x_{i'}+$$

$$+\sum_{i=1}^{N}a_i-\sum_{i=1}^{N}(C+b_i)\xi_i-\sum_{i=1}^{N}b_i\xi_i,$$

$$L(\beta,\beta_0,\xi_i)=\frac{1}{2}\|\beta\|^2+C\sum_{i=1}^{N}\xi_i-\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}x_i^T x_{i'}+$$

$$+\sum_{i=1}^{N}a_i-C\sum_{i=1}^{N}\xi_i+\sum_{i=1}^{N}b_i\xi_i-\sum_{i=1}^{N}b_i\xi_i,$$

$$L(\beta,\beta_0,\xi_i)=\frac{1}{2}\|\beta\|^2+\sum_{i=1}^{N}a_i-\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}x_i^T x_{i'}$$

Розпишемо квадрат вектора норми:

$$\|\beta\|^2=\beta\beta^T$$

Тоді:

$$L(\beta,\beta_0,\xi_i)=\frac{1}{2}\beta\beta^T+\sum_{i=1}^{N}a_i-\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}x_i^T x_{i'}=$$

$$=\frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}x_i^T x_{i'}+\sum_{i=1}^{N}a_i-\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}x_i^T x_{i'}=$$

$$=\sum_{i=1}^{N}a_i-\frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}x_i^T x_{i'}$$

Отже, маємо формулу для функції Лагранжа:

$$L=\sum_{i=1}^{N}a_i-\frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}x_i^T x_{i'}\to max,$$

$$-L=\frac{1}{2}\sum_{i=1}^{N}\sum_{i'=1}^{N}a_i a_{i'}y_i y_{i'}<x_i^T x_{i'}>-\sum_{i=1}^{N}a_i\to min$$

З обмеженнями:

$$\begin{cases} -a_i \le 0 \\ a_i \le C \\ \sum_{i=1}^{N} a_i y_i = 0 \end{cases}$$

Розв'язавши цю задачу, знаходимо $\widehat{a_i}$. Видно, що розв'язок для β має вигляд:

$$\hat{\beta} = \sum_{i=1}^{N} \widehat{a_i} y_i x_i^T$$

Коефіцієнти a_i не дорівнюють нулю ($a_i > 0$) тільки для тих спостережень i, для яких точно виконується: $y_i(x^T\beta + \beta_0) \ge (1 - \xi_i)$. Ці спостереження називаються опорними векторами, оскільки вектор β виражається тільки через них. Серед тих опорних точок деякі будуть лежати на краю проміжку $\widehat{\xi_i}$=0. З того, що $b\xi_i = 0$ та $a_i = C + b_i$ випливає, що в цьому випадку 0 < $\widehat{a_i}$ < C. При $\widehat{\xi_i} > 0$: $\widehat{a_i}$ = C. Правило класифікації, має вигляд:

$$y = sign(x^T\beta + \beta_0)$$

Отже, при відомих значеннях $\hat{\beta}$ та $\widehat{\beta_0}$, маємо:

$$\hat{y} = sign\left(x^T\hat{\beta} + \widehat{\beta_0}\right)$$

1.2.5 Класифікація багатошаровим персептроном

Відповідно до [10] багатошаровий персептрон – це нейронна мережа прямого поширення, яка має вхідний шар, приховані шари та вихідний шар. Шари складаються з нейронів, на вхід яких подаються вагові коефіцієнти з попереднього шару, з виходами, які йдуть в наступний шар. Вони поєднуються нелінійною активаційною функцією.

X – вхідна множина даних, матриця (m, n). Тоді позначимо x – вектор-стовпець з матриці X, розмірність якого (m, 1).

Матриця вагів W на кожному шарі є різної розмірності, а саме (s, v), де s-кількість нейронів на наступному шарі та v - кількість нейронів на поточному шарі.

Активаційний потенціал [10]:

$$z_i = \sum_{i=1}^{s} w_{ij} x_j + b_i,$$

де

z_i – активаційний потенціал;

j = $\overline{1, v}$;

w_{ij} – елемент матриці вагів W;

b_i – баяс-нейрон, що відповідає відступу.

Вище описане можна написати у векторній формі:

$$\boldsymbol{z = Wx + b}$$

Активаційна функція активаційного потенціалу [10]:

$$\boldsymbol{a = f(z)},$$

де

f – нелінійна функція.

Тоді маємо пряме поширення [10]:

$$x = z^{(1)} = a^{(1)},$$

де

$z^{(1)}$ – активаційний потенціал на першому шарі, тобто на вхідному шарі;

$a^{(1)}$ – активаційна функція на першому шарі.

$$\boldsymbol{z^{(2)} = W^{(1)} x + b^{(1)}},$$

де

$z^{(2)}$ – активаційний потенціал на другому шарі, тобто на першому прихованому шарі;

$W^{(1)}$ – матриця вагів, елементами якої є $w_{ij}{}^{(1)}$- елемент цієї матриці, що з'єднує i-тий елемент вхідного шару та j-тий елемент першого прихованого шару.

$b^{(1)}$ – відступ для першого прихованого шару.

Тоді активаційна функція для активаційного потенціалу на першому прихованому шарі має вигляд:

$$a^{(2)} = f(z^{(2)})$$

Активаційний потенціал на другому прихованому шарі:

$$z^{(3)} = W^{(2)}a^{(2)} + b^{(2)},$$

де

$W^{(2)}$ – матриця вагів, елементами якої є ${w_{ij}}^{(2)}$ – елемент цієї матриці, що з'єднує i-тий елемент першого прихованого шару та j-тий елемент другого прихованого шару.

$b^{(2)}$ – відступ для другого прихованого шару.

Тоді активаційна функція для активаційного потенціалу на другому прихованому шарі має вигляд:

$$a^{(3)} = f(z^{(3)})$$

Тоді активаційний потенціал на l-тому прихованому шарі [10]:

$$z^{(l+1)} = W^{(l)}a^{(l)} + b^{(l)},$$

де

$W^{(l)}$ – матриця вагів, елементами якої є ${w_{ij}}^{(l)}$- елемент цієї матриці, що з'єднує i-тий елемент $(l-1)$ прихованого шару та j-тий елемент l-го прихованого шару;

$b^{(l)}$ – відступ для l-го прихованого шару.

Тоді активаційна функція для активаційного потенціалу на l-тому прихованому шарі має вигляд:

$$a^{(l+1)} = f(z^{(l+1)})$$

Якщо k-тий прихований шар є останнім, то за ним йде вихідний шар. Кількість нейронів у вихідному шарі дорівнює кількості класів для задачі класифікації. Тоді активаційний потенціал на k -тому прихованому шарі [10]:

$$z^{(k+1)} = W^{(k)}a^{(k)} + b^{(k)},$$

де

$W^{(k)}$ – матриця вагів, елементами якої є ${w_{ij}}^{(k)}$- елемент цієї матриці, що з'єднує i-тий елемент $(k-1)$ прихованого шару та j-тий елемент k-го прихованого шару;

$b^{(k)}$ – відступ для k-го прихованого шару.

Тоді активаційна функція для активаційного потенціалу на l-тому прихованому шарі має вигляд:

$$a^{(k+1)} = f(z^{(k+1)})$$

Тоді вихідний шар приймає вигляд [10]:

$$\hat{y} = U^T a^{(k+1)},$$

де

$\hat{y}$ – прогнозовані значення y;

U – матриця вагів для вихідного шару.

Так як проводиться бінарна класифікація, то в якості функції помилки задається функція бінарної перехресної-ентропії [10]:

$$L = -\frac{1}{m}\sum_{i=1}^{m} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i),$$

де

m – кількість залежних змінних;

y_i – i-тий елемент вектора залежних змінних;

$\hat{y}_i$ – i-тий елемент вектора прогнозованих залежних змінних.

Тепер переходимо до етапу зворотного поширення, відповідно до [10] потрібно мінімізувати функцію помилки, та оновити ваги. Формульно для l-го прихованого шару з [10] це виглядає так:

$$\delta^{(L)} = y - \hat{y},$$

де $\delta^{(L)}$- помилка на вихідному шарі.

$$\delta^{(l)} = \left(\left(W^{(l)}\right)^T \delta^{(l+1)}\right) \otimes f'\left(z^{(l)}\right),$$

де

$\delta^{(l)}$- помилка на для l-му прихованому шарі;

$\otimes$- тензорний добуток.

Для оновлення вагів, скористаємось методом швидкого спуску. Для цього потрібно обчислити градієнт функції бінарної перехресної-ентропії для l-го прихованого шару відповідно до [10], маємо:

$$\frac{\partial L}{\partial W^{(l)}} = \delta^{(l+1)}\left(a^{(l)}\right)^T,$$

$$\frac{\partial L}{\partial b^{(l)}} = \delta^{(l+1)}$$

Тепер застосовуємо метод найшвидшого спуску з [10] та маємо оновлення вагів для l-го прихованого шару:

$$W^{(l)} = W^{(l)} - \alpha\frac{\partial L}{\partial W^{(l)}},$$

$$b^{(l)} = b^{(l)} - \alpha\frac{\partial L}{\partial b^{(l)}},$$

де α- коефіцієнт навчання, $\alpha \in \mathbb{R}$ та приймає значення [0, 1].

Після застосування методу найшвидшого спуску, маємо оптимальні значення вагів для прогнозу. Отже, маємо прогнозований вектор вихідних значень:

$$\hat{y} = \hat{U}^T\hat{a}^{(k+1)},$$

де

$\hat{U}$ – матриця оптимальних вагів для вихідного слою;

$\hat{a}$ – активаційна функція, для бінарної класифікації використовується сигмоїдна функція.

1.3 Оцінки якості моделей

Для того, щоб порахувати точність (accuracy) моделі використовують формулу з [11]

$$accuracy = \frac{1}{m}\sum_{i=1}^{m} \mathrm{I}[\hat{y}_i = y_i],$$

де

m – кількість елементів у векторі y;

$\mathrm{I}[\hat{y}_i = y_i]$ – індикаторна функція, якщо $\hat{y}_i = y_i$, то функція дорівнює 1;

$\hat{y}_i$ – спрогнозовані значення залежних змінних;

y_i – залежні змінні.

Ще однією оцінкою якості моделі є f1-оцінка, формула для якої є в [11]:

$$F_1 = 2\frac{Precision \cdot Recall}{Precision + Recall},$$

де F_1 – це f1 – оцінка;

Формули для $Precision$ та $Recall$ відповідно:

$$Precision = \frac{TP}{TP + FP},$$

$$Recall = \frac{TP}{TP + FN},$$

де

TP – це кількість істинно позитивних передбачень;

FP – кількість хибно позитивних передбачень;

FN – кількість хибно негативних передбачень.

Macro avarage - це середнє арифметичне значень метрик (такі як precision, recall та f1-оцінка) для всіх класів. Загальна формула для обрахунку [11]

$$Macro\ average = \frac{1}{N}\sum_{i=1}^{N} M_i,$$

де

N – це кількість класів, для бінарної класифікації N=2;

M_i – це метрика для i-го класу (наприклад значення f1-оцінки).

Так як середнє арифметичне вразливе до викидів, то ще беруть weighted average – це зважене середнє для метрик усіх класів, зважене по кількості прикладів у кожному класі (support), оскільки класи з більшою кількістю прикладів роблять більший внесок на підсумкове значення. Формула для обчислення середнього зваженого наведена в [11] та має вигляд:

$$Weighted\ average = \frac{\sum_{i=1}^{N} M_i \cdot support_i}{\sum_{i=1}^{N} support_i},$$

де $support_i$ – це i-тий приклад у i-тому класі.

Висновки до розділу 1

Отже, було проаналізовано сучасні статті для проведення прогнозування та класифікації інфікованих пацієнтів. На основі цих досліджень було обрано такі методи машинного навчання: логістичну регресію, її модифікацію - ядерну логістичну регресію, опорних векторів та багатошаровий персептрон. Задля того щоб візуалізувати ознаки множини даних був обраний метод головних компонент, який здатен зменшити розмірності вхідних даних. До того ж він може бути застосований, коли множина вхідних даних має завелику кількість ознак для коректної класифікації. Також метод головних компонент допоможе оптимізувати роботу алгоритмів машинного навчання за рахунок зменшення розмірності множини даних. Для аналізу та опису методів машинного навчання була проаналізована класична література з теорії ймовірностей, математичної статистики, методів оптимізації та інтелектуального аналізу даних. Всі методи, окрім алгоритму головних компонент, використовуються для бінарної класифікації, а саме для визначення чи є пацієнт за деякими ознаками інфікованим вірусом імунодефіциту людини, чи ні.

Були розглянуті різні оцінки якості моделей, а саме: точність (accuracy), precision, recall, f1-оцінка, середнє арифметичне значень метрик (macro average) та зважене середнє (weighted average). Ці оцінки допоможуть зрозуміти, чи є модель адекватною та чи можна її взагалі застосовувати для класифікації пацієнтів. Також оцінки якості моделей допоможуть вибрати найоптимальнішу модель, яка буде краще за інші моделі проводити класифікацію.

2 ПОБУДОВА МОДЕЛІ ДЛЯ КЛАСИФІКАЦІЇ ПАЦІЄНТІВ

2.1 Опис множини даних

Множина даних містить медичну статистику та інформацію про пацієнтів з діагнозом синдрому набутого імунодефіциту. Цей набір даних був вперше опублікований у 1996 році. Множина даних складається з 2139 рядків та 23 стовпців (ознак). Типи даних та їх кількість наведені в таблиці 2.1

Таблиця 2.1 — Типи даних

int64	22
float64	1

Множина даних не має пропущених значень. Дані було взято з [12]. Опис ознак множини даних:

- time: час до відмови або цензури
- trt: показник лікування
- age: вік (років) на початку дослідження
- wtkg: вага (кг) на початку дослідження
- hemo: гемофілія (0=ні, 1=так)
- homo: гомосексуальна активність (0=ні, 1=так)
- drugs: історія вживання наркотиків внутрішньовенно (0=ні, 1=так)
- karnof: оцінка функціонального стану за шкалою Карновського (від 0 до 100)
- oprior: антиретровірусна терапія (0=ні, 1=так)
- z30: ЗДВ протягом 30 до 175 днів (0=ні, 1=так)
- preanti: дні до антиретровірусної терапії
- race: раса (0=біла, 1=небіла)
- gender: стать (0=жінка, 1=чоловік)
- str2: антиретровірусна історія (0=початківець, 1=досвідчений)
- strat: стратифікація антиретровірусного анамнезу (1='Наявний', 2='> 1, але <= 52 тижнів попередньої антиретровірусної терапії', 3='> 52 тижнів')

- symptom: симптоматичний індикатор (0=асимптом, 1=симптом)
- treat: індикатор лікування (0=тільки ЗДВ, 1=інші)
- offtrt: індикатор відсутності лікування до 96+/-5 тижнів (0=ні, 1=так)
- cd40: CD4 на початковому рівні
- cd420: CD4 на 20+/-5 тижні
- cd80: CD8 на початковому рівні
- cd820: CD8 на 20+/-5 тижні
- infected: інфікований СНІДом (0=Ні, 1=Так)

Першим кроком обробки даних було знаходження викидів та видалення рядків з порожніми значеннями. Також було проведено стандартизацію даних, аби покращити обробку, аналіз даних та використання методу головних компонент.

Другим кроком обробки множини даних буде візуалізація даних, а саме: побудова гістограм розподілів даних, стовпчастих діаграм. Також буде проаналізовано кореляцію даних за допомогою коефіцієнта кореляції Пірсона. Задля кращого сприйняття побудовано матрицю кореляцій між ознаками множини даних.

Формула коефіцієнта кореляції Пірсона з [11] має вигляд:

$$r = \frac{\sum_{i=1}^{m}(x_i - \bar{x}_i)(y_i - \bar{y}_i)}{\sqrt{\sum_{i=1}^{m}(x_i - (\bar{x}_i)^2)\sum_{i=1}^{m}(y_i - (\bar{y}_i)^2)}}$$

2.2 Імплементація первинного аналізу

На рисунку 2.1 наведено гістограми для кожної з ознак, що відображають розподіл значень кожної ознаки серед вибірки пацієнтів. З них видно, що ознаки hemo, homo, drugs, opriror, z30, race, gender, str2, symptom, treat, offtrt та infected мають бінарний відгук.

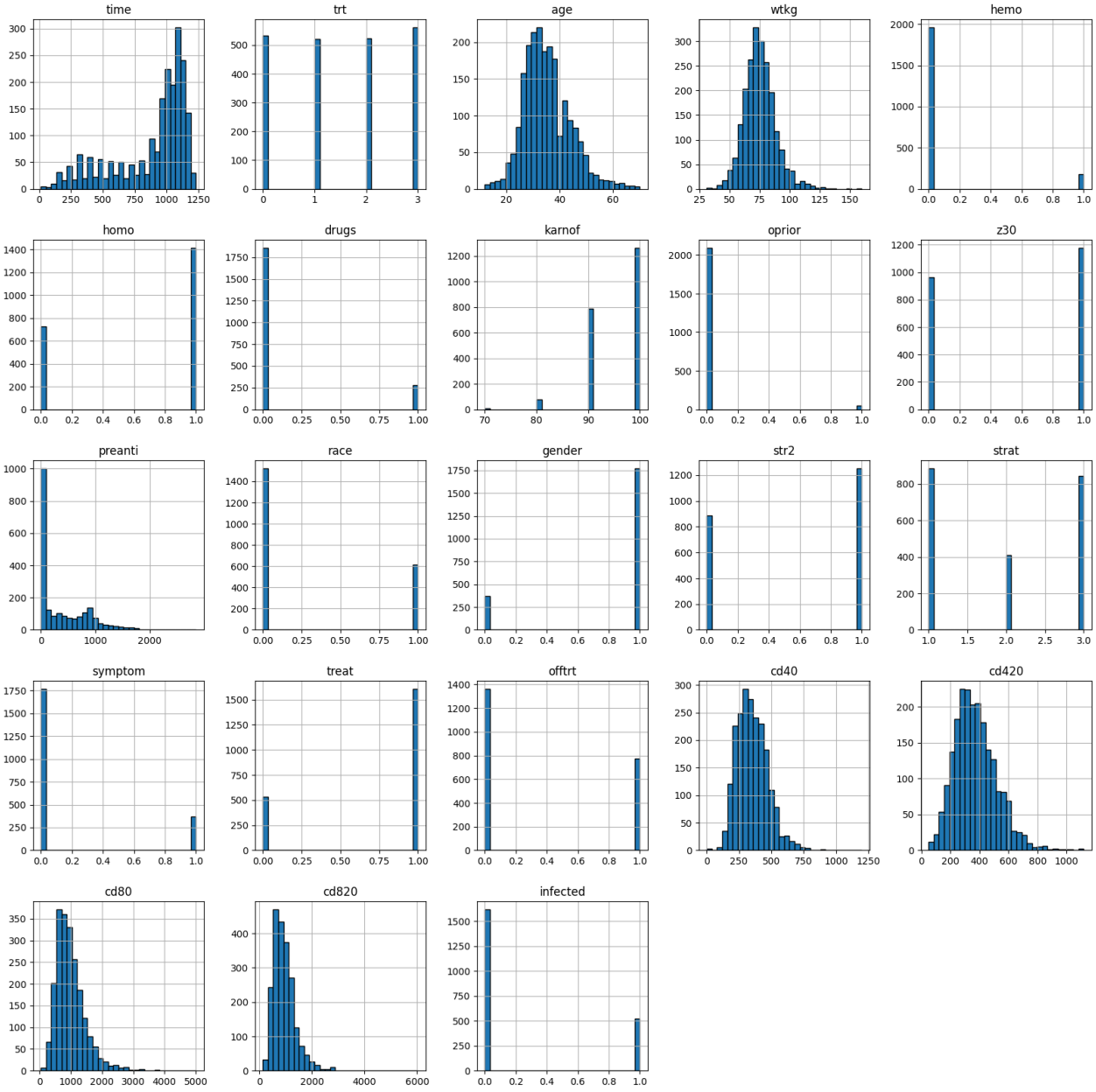

Рисунок 2.1 — Візуалізація розподілу даних

Проведено візуалізацію викидів. Як видно з рисунку 2.2 найбільше викидів мають ознаки preanti, cd80 та cd820. Найменшу кількість викидів мають такі колонки як age, wtkg, hemo, homo, drugs, karnof, opriror, z30, race, gender, str2, symptom, treat, offtrt та infected, більшість яких є ознаки з бінарними відгуками.

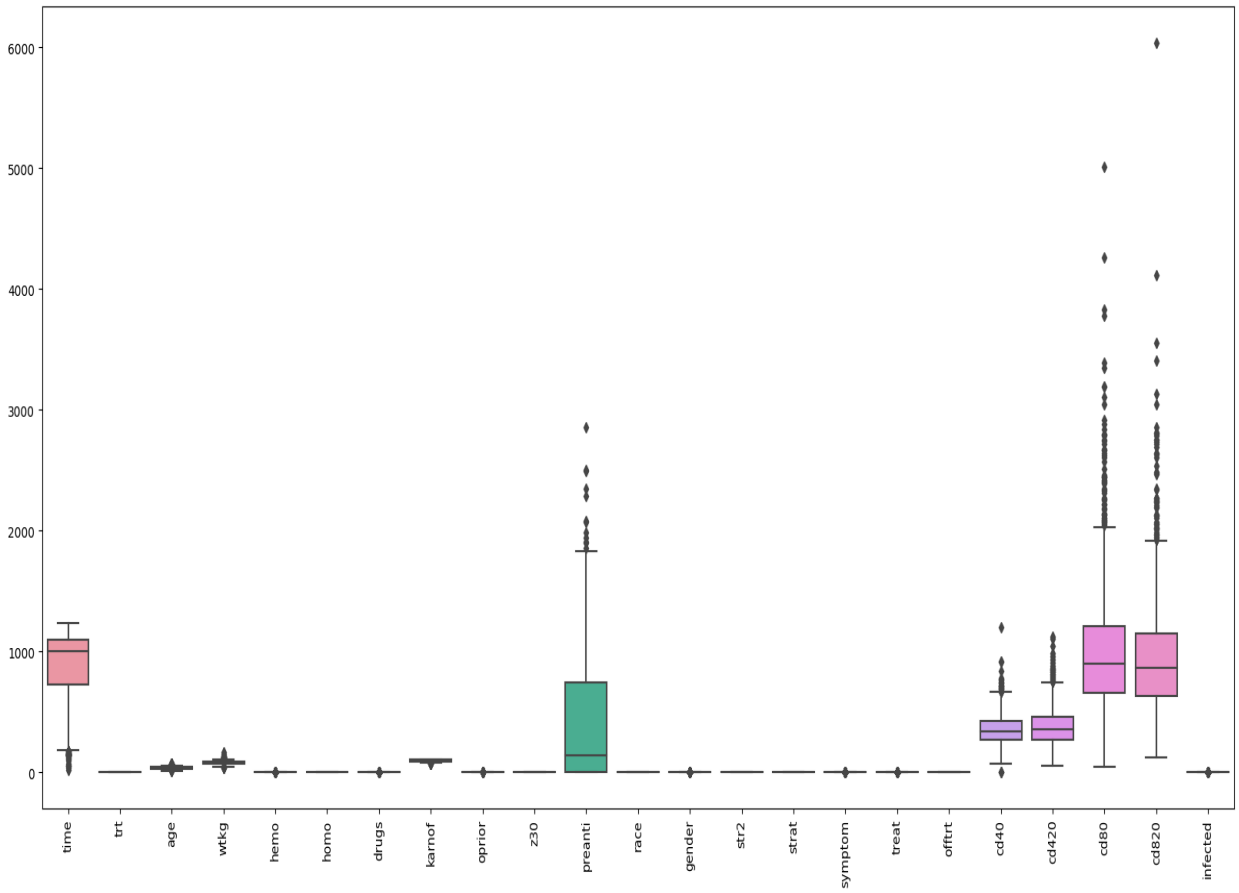

Рисунок 2.2 — Візуалізація викидів

З рисунку 2.3 видно, що більшість пацієнтів, у множині даних, не є інфікованими, а саме 1618 чоловік. Інфікованими ж синдромом імунодефіциту людини є 521 пацієнт. Якщо цю кількість перевести у відсотки, то 75,64% не є інфікованими та 24,36% є інфікованими. Тобто було відібрано саме тих пацієнтів, які належать оцим 24,36%, щоб проаналізувати більш детально, які саме ознаки впливають на інфікування. Як і очікувалось найбільш вагомими колонками виявились hemo, homo та drugs.

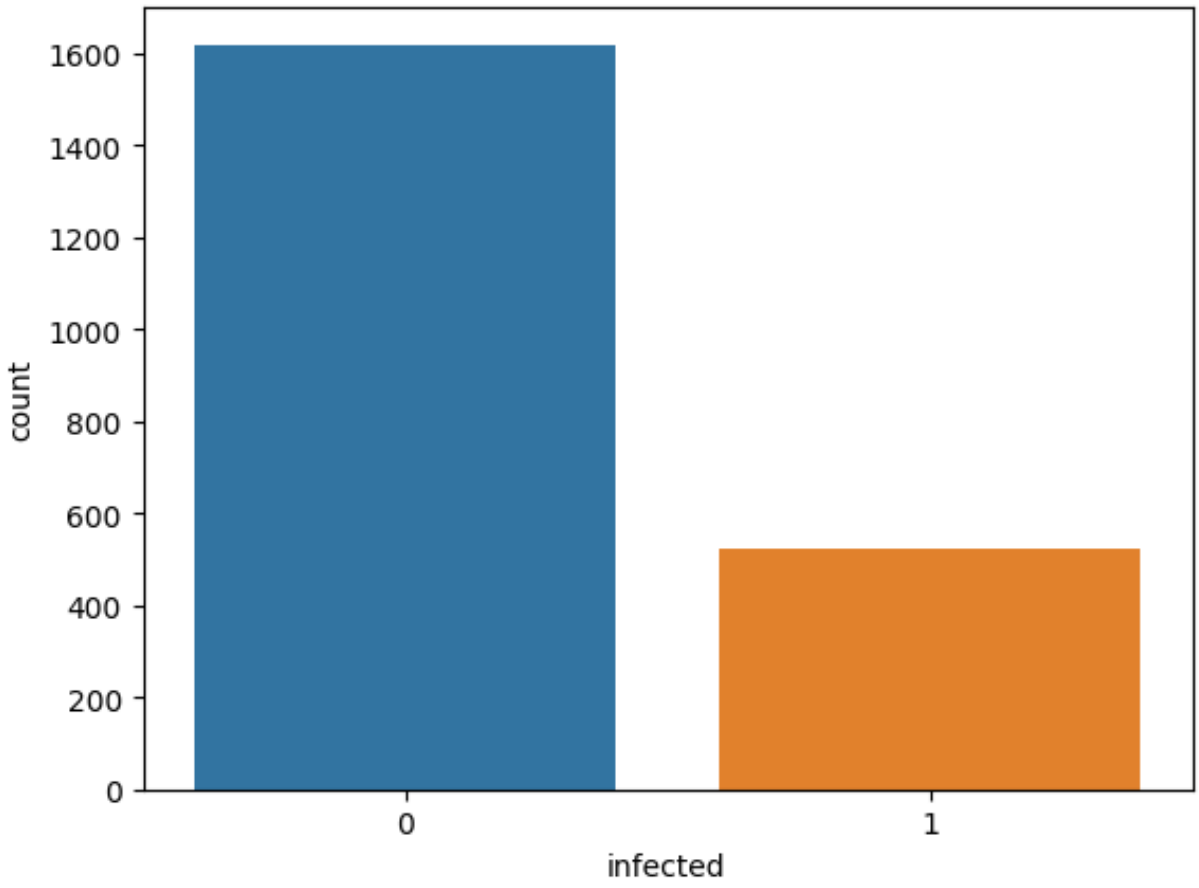

Рисунок 2.3 — Кількість не інфікованих пацієнтів (значення 0) та інфікованих(значення 1)

На рисунку 2.4 показано як корелюють між собою ознаки коефіцієнтів кореляції Пірсона. Як видно, найбільше корелюють між собою ознаки:

- preanti та z30 з коефіцієнтом кореляції рівним 0,66
- gender та homo з коефіцієнтом кореляції рівним 0,61
- str2 та z30 мають один з найбільших коефіцієнтів кореляції, а саме 0,90
- str2 та preanti з коефіцієнтом кореляції рівним 0,68
- strat та z30 з коефіцієнтом кореляції рівним 0,85
- strat та preanti з коефіцієнтом кореляції рівним 0,83
- strat та str2 мають найбільший коефіцієнт кореляції, а саме 0,92
- treat та trt мають коефіцієнт кореляції 0,78
- cd820 та cd80 теж мають доволі високий коефіцієнт кореляції, а саме 0,76

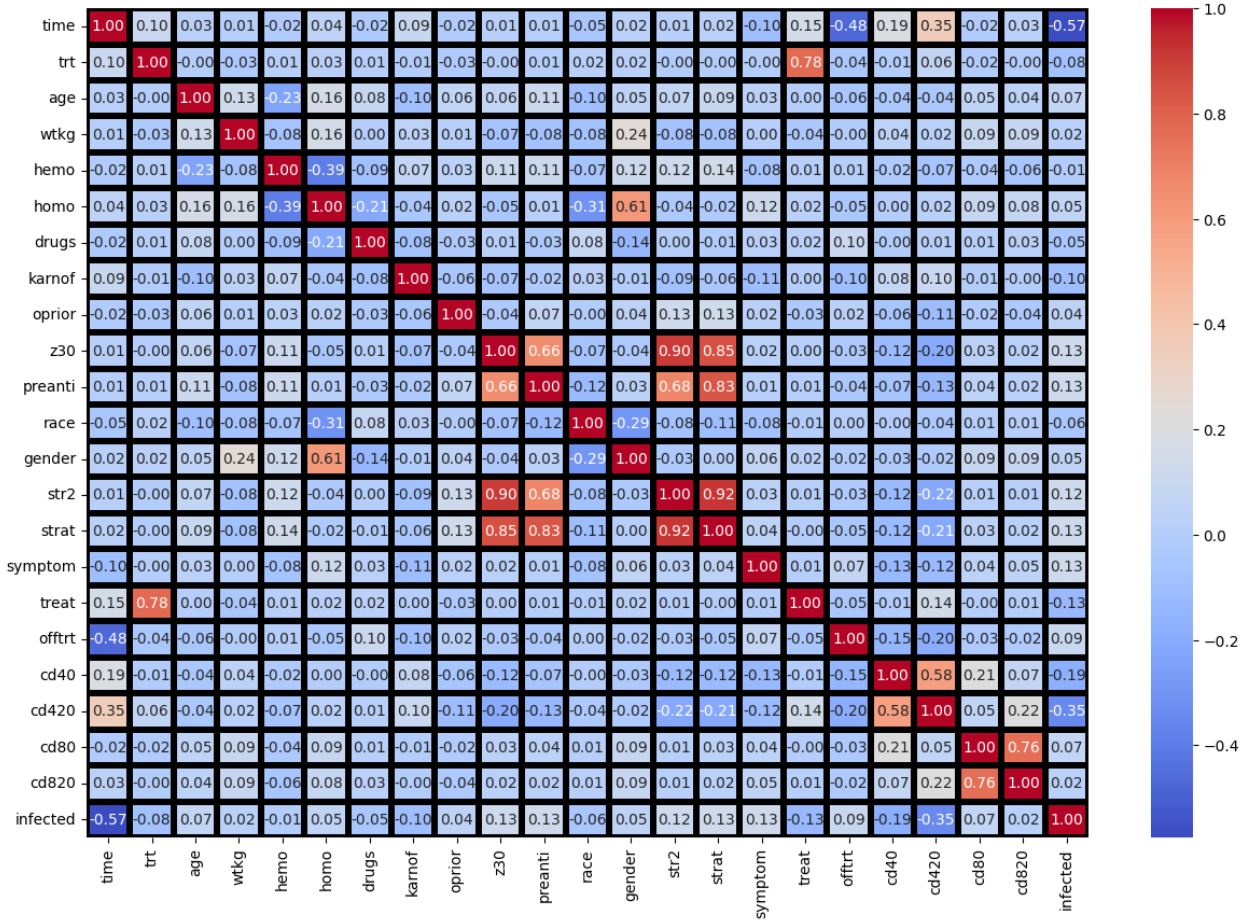

Рисунок 2.4 — Кореляційна матриця

Було обрано зменшити розмірність вхідних даних до 5 компонент для кращого сприйняття. На рисунку 2.5 видно, що дані можливо класифікувати з досить непоганою точністю. Однак вже ясно, що вони є лінійно нероздільними, що значно ускладнює задачу класифікації. Тим не менш обрані для експерименту методи машинного навчання, такі як ядерна логістична регресія, опорних векторів та багатошаровий персептрон, здатні якісно класифікувати лінійно нероздільні дані.

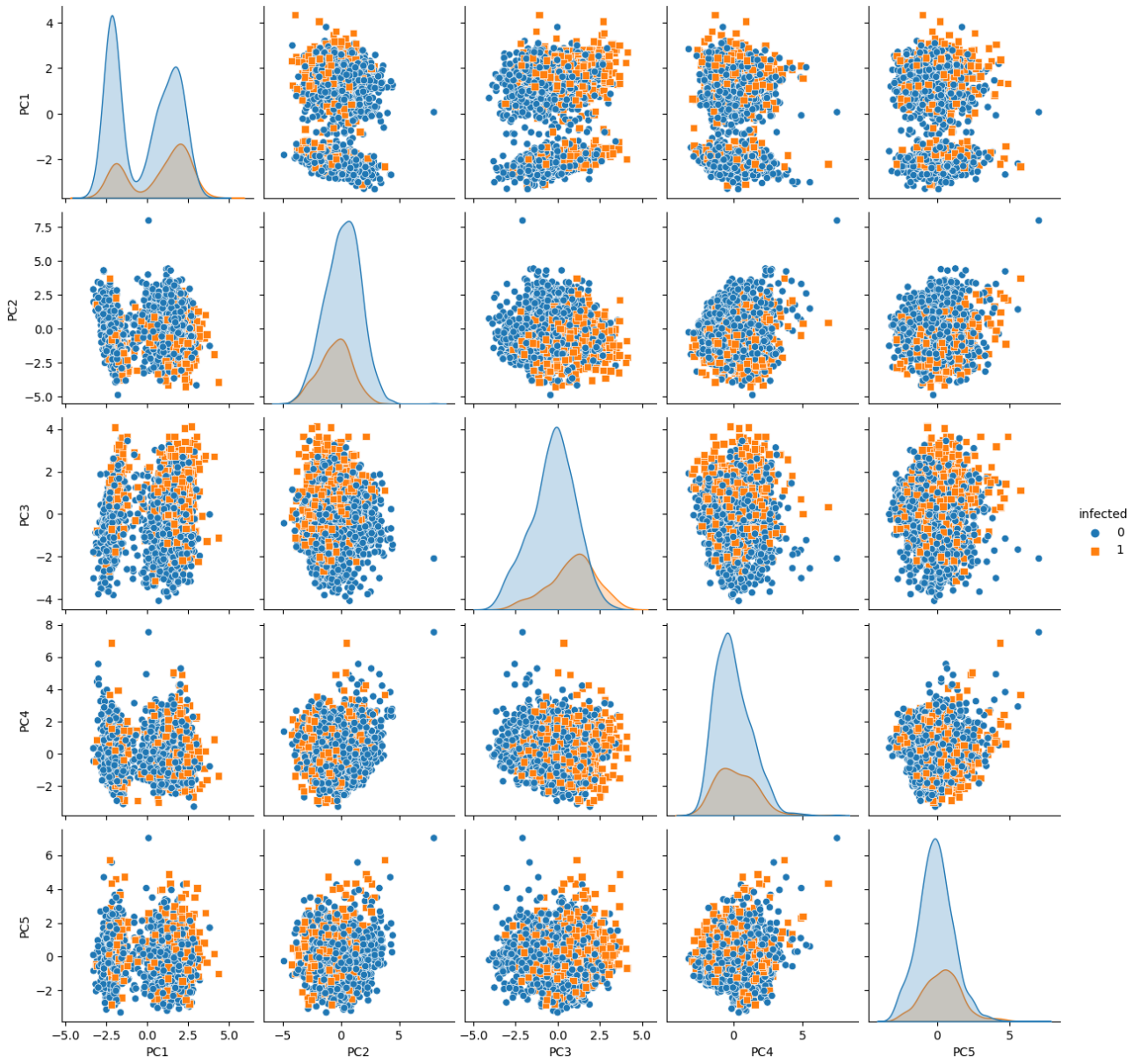

Рисунок 2.5 — Візуалізація ознак, які пояснюють найбільшу частину варіації в даних

Графік показує, як перші п’ять головних компонент відображають структуру даних і наскільки вони здатні розрізняти інфікованих та неінфікованих осіб. Незважаючи на те, що певне перекриття між класами є, ці компоненти містять потрібну для дослідження інформацію, а також необхідну для побудови моделей класифікації, щоб спрогнозувати, чи є інфікованим пацієнт синдромом набутого імунодефіциту.

Також було побудовано порівняльні графіки між різними головними компонентами, знову ж таки за допомогою метода головних компонент:

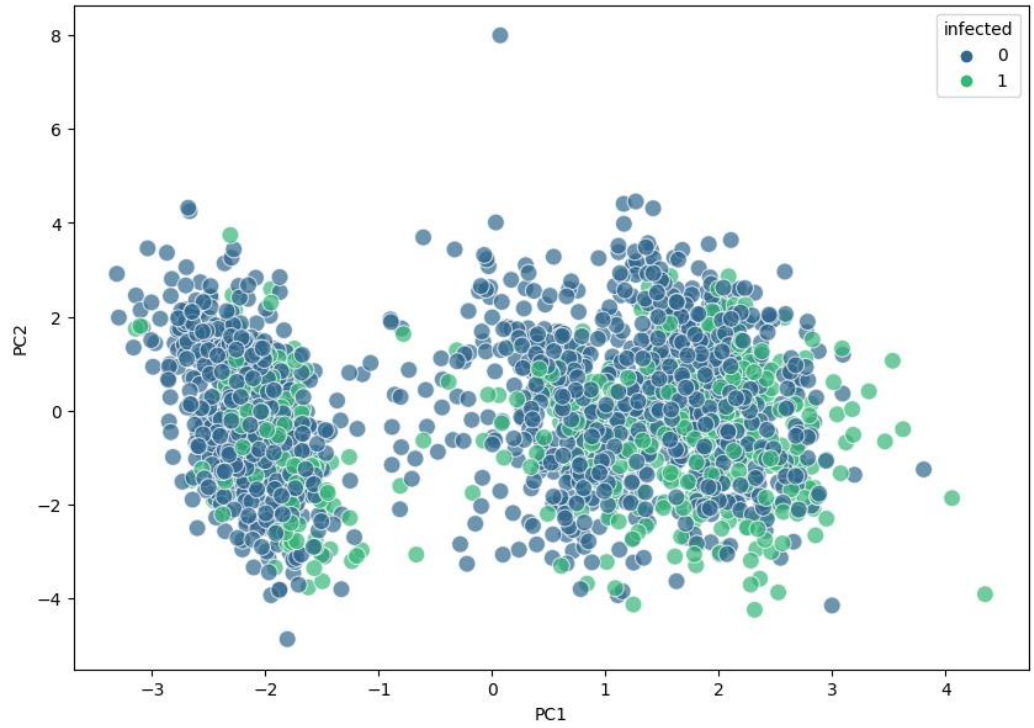

Рисунок 2.6 — порівняння першої головної компоненти та другої

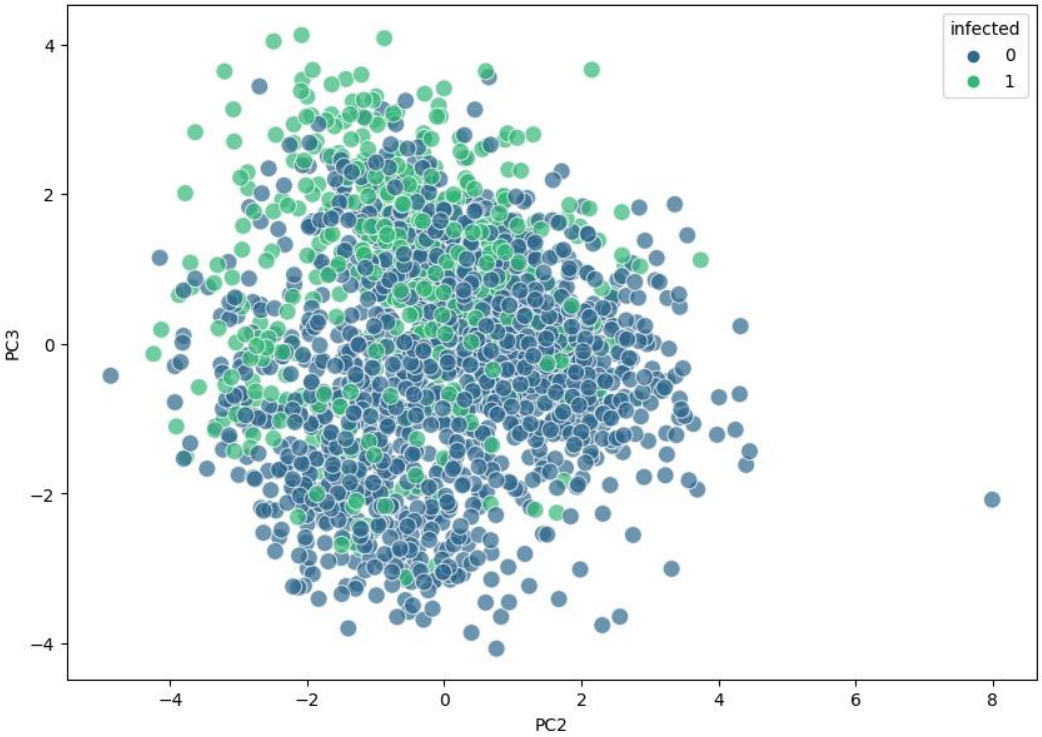

Рисунок 2.7 — порівняння другої головної компоненти та третьої

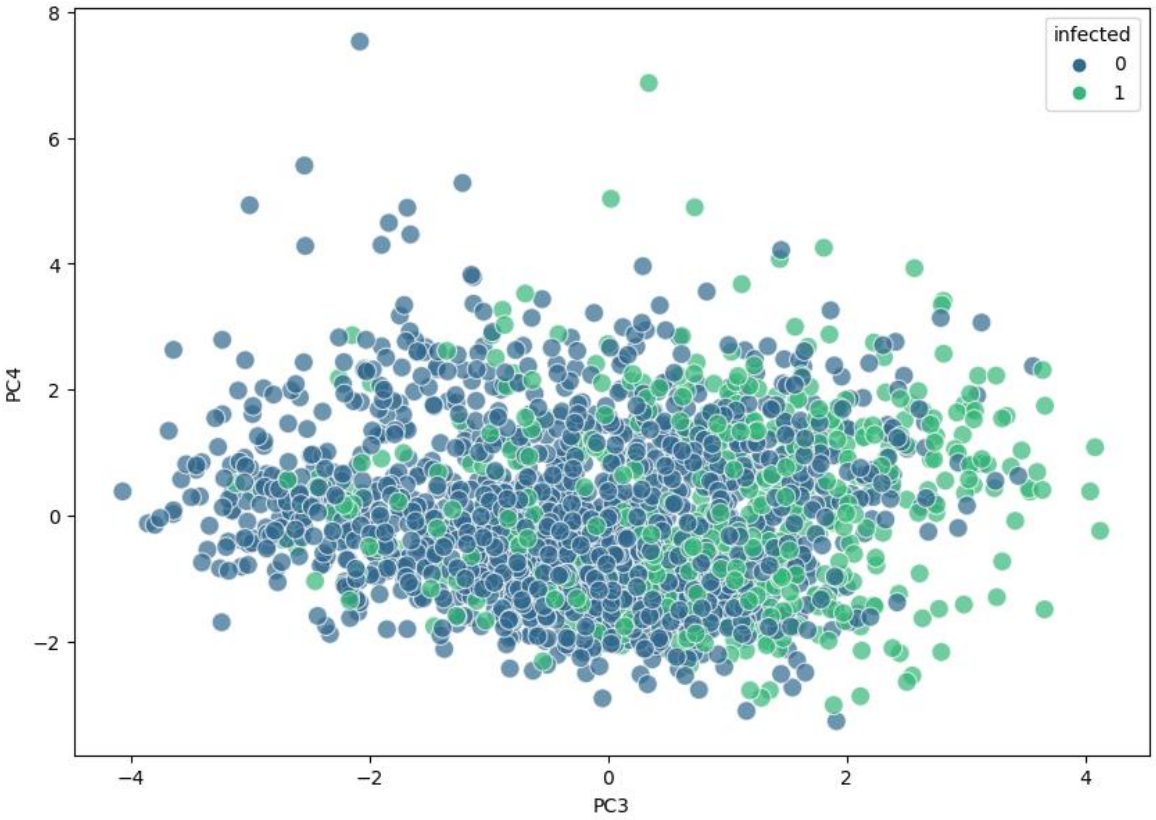

Рисунок 2.8 — порівняння третьої головної компоненти та четвертої

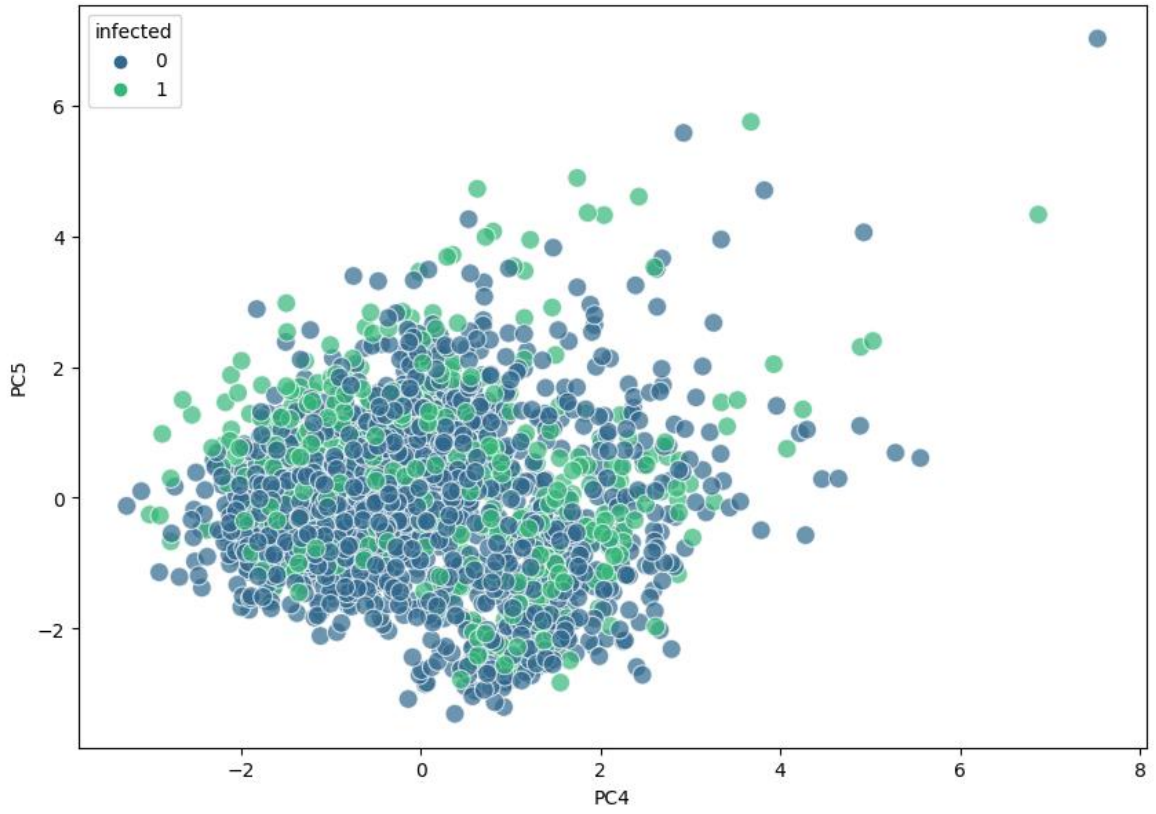

Рисунок 2.9 — порівняння четвертої головної компоненти та п’ятої

2.3 Класифікація логістичною регресією

За допомогою мови програмування Python та додатка Jupyter notebook був побудований клас логістичної регресії, використовуючи математичні викладки з першого розділу. Аби уникнути перенавчання та протестувати модель на різних даних, було вирішено розділи множину даних на тренувальну та тестову вибірки, кожна з яких випадково містить 50 відсотків зі всіх даних.

Результати якості моделі на тренувальному наборі містяться в таблиці 2.2. Як видно з таблиці 2.2, модель має досить непогану точність, а саме 86%, що свідчить про досить добру лінійну роздільність даних, хоча дані як вже було згадано вище мають лінійно нероздільну природу. Тим не менш f1-оцінка для інфікованих пацієнтів є 68%, що не є якісним результатом для класифікації.

Таблиця 2.2 — Звіт по класифікації на тренувальному наборі даних логістичної регресії без вбудованої бібліотеки

	precision	recall	f1-score	support
0	0.94	0.88	0.91	863
1	0.61	0.77	0.68	206
accuracy			0.86	1069
macro avg	0.78	0.83	0.80	1069
weighted avg	0.88	0.86	0.87	1069

Порівняння результатів на тренувальному наборі з вбудованою логістичною регресією з математичного пакета scikit learn містяться в таблиці 2.3.

Таблиця 2.3 — Звіт по класифікації на тренувальному наборі даних з вбудованою бібліотекою scikit learn

	precision	recall	f1-score	support
0	0.94	0.88	0.91	864
1	0.61	0.77	0.68	205
accuracy			0.86	1069
macro avg	0.77	0.83	0.79	1069
weighted avg	0.88	0.86	0.87	1069

Результати якості моделі на тестовому наборі містяться в таблиці 2.4. Тепер точність є трохи вищою, ніж на тренувальному наборі та дорівнює 88%. Також значення F1-оцінки теж підвищилось для інфікованих пацієнтів, та становить тепер 73%.

Таблиця 2.4 — Звіт по класифікації на тестовому наборі даних логістичної регресії без вбудованої бібліотеки

	precision	recall	f1-score	support
0	0.95	0.90	0.92	861
1	0.66	0.82	0.73	209
accuracy			0.88	1070
macro avg	0.81	0.86	0.83	1070
weighted avg	0.90	0.88	0.89	1070

Порівняння результатів на тестовому наборі з вбудованою логістичною регресією з математичного пакета scikit learn містяться в таблиці 2.5.

Таблиця 2.5 — Звіт по класифікації на тестовому наборі даних логістичної регресії з вбудованою бібліотекою scikit learn

	precision	recall	f1-score	support
0	0.95	0.90	0.92	862
1	0.65	0.82	0.73	208
accuracy			0.88	1070
macro avg	0.80	0.86	0.82	1070
weighted avg	0.89	0.88	0.89	1070

Отже, логістична регресія показала досить непогані результати на тренувальному наборі та ще кращі на тестовому. Тим не менш модель не може класифікувати лінійно нероздільні дані, що свідчить про подальшу її модифікацію, а саме вже ядерною логістичною регресією. В результаті якої вже буде можливо класифікувати лінійно нероздільні дані, та значно підвищити точність класифікації.

2.4 Класифікація ядерною логістичною регресією

Ядерну логістичну регресію було обрано задля того, щоб покращити результат класифікації у порівнянні з логістичною регресією. За допомогою ядерних методів вдалося зробити класифікацію для лінійно нероздільних класів. Також було обрано найбільш оптимальне ядро – ядро Лапласа. Тим не менш ядерна логістична регресія має високу здатність до перенавчання. Задля того, щоб уникнути перенавчання, було застосовано сильну регуляризацію.

Значення коефіцієнта навчання для методу швидкого спуску дорівнює 0,01. Було обрано ядро Лапласа, максимальна кількість ітерацій для методу швидкого спуску дорівнює 1000 ітерацій. Також параметри регуляризації $\alpha = 0{,}1$ для функції максимальної правдоподібності та $\gamma = 0{,}1$ для ядра Лапласа відповідно.

Результати якості моделі на тренувальному наборі містяться в таблиці 2.6. Як видно з таблиці 2.6, модель має вже більшу точність у порівнянні з моделлю логістичної регресії, а саме 95%, що свідчить про покращення результатів та що тепер модель може враховувати нелінійні взаємозв'язки між ознаками. Також f1-оцінка теж підвищилась та тепер для інфікованих пацієнтів становить 90 відсотків.

Таблиця 2.6 — Звіт по класифікації на тренувальному наборі даних ядерної логістичної регресії

	precision	recall	f1-score	support
0	0.94	1.00	0.96	759
1	0.99	0.83	0.90	310
accuracy			0.95	1069
macro avg	0.96	0.91	0.93	1069
weighted avg	0.95	0.95	0.95	1069

Результати якості моделі на тестовому наборі містяться в таблиці 2.7. Тепер точність є трохи нижчою, ніж на тренувальному наборі та дорівнює 94%. Також значення f1-оцінки теж зменшилось для інфікованих пацієнтів, та становить тепер 87%.

Таблиця 2.7 — Звіт по класифікації на тестовому наборі даних ядерної логістичної регресії

	precision	recall	f1-score	support
0	0.99	0.94	0.96	1188
1	0.80	0.96	0.87	309
accuracy			0.94	1497
macro avg	0.90	0.95	0.92	1497

Отже, модифікована логістична регресія має більшу точність на тренувальному та тестовому наборах та тепер може класифікувати лінійно нероздільні дані. Тим не менш час роботи алгоритму виріс у декілька разів у порівнянні з методом логістичної регресії. Однак, метод опорних векторів є менш затратним в плані часу роботи алгоритму в порівнянні з ядерною логістичною регресією, використовує такі ж ядерні методи та ще й максимізує відстань між класами, також менше схильний до перенавчання.

2.5 Класифікація методом опорних векторів

Метод опорних векторів було обрано задля того, щоб покращити результат класифікації у порівнянні з ядерною логістичною регресією. Також було обрано найбільш оптимальне ядро - радіально базисне ядро. Було застосовано регуляризацію, щоб результати на тренувальному та тестовому наборах були оптимальними.

Значення константи C, яка використовується в обмеженні, дорівнює 1 . Було обрано радіально базисне ядро, для якого $\gamma = 0{,}1$. Також у векторі залежної змінної всі 0 були замінено на -1.

Результати якості моделі на тренувальному наборі містяться в таблиці 2.8. Як видно з таблиці 2.8, модель має таку саму точність у порівнянні з моделлю ядерної логістичної регресії, а саме 95%. А ось f1-оцінка знизилась на 1% для інфікованих пацієнтів, але це не є надто критичним.

Таблиця 2.8 — Звіт по класифікації на тренувальному наборі даних

	precision	recall	f1-score	support
-1	0.99	0.94	0.97	848
1	0.82	0.97	0.89	221
accuracy			0.95	1069
macro avg	0.91	0.96	0.93	1069
weighted avg	0.96	0.95	0.95	1069

Результати якості моделі на тестовому наборі містяться в таблиці 2.9. Точність однакова у порівнянні з результатами на тренувальних даних. Також значення F1-оцінки піднялось на 1% для інфікованих пацієнтів та становить тепер 90%.

Таблиця 2.9 — Звіт по класифікації на тестовому наборі даних

	precision	recall	f1-score	support
-1	0.99	0.95	0.97	846
1	0.83	0.97	0.90	224
accuracy			0.95	1070
macro avg	0.91	0.96	0.93	1070
weighted avg	0.96	0.95	0.95	1070

Отже, побудована модель має таку саму точність на тренувальному та тестовому наборах, що й ядерна логістична регресія. Тим не менш час роботи алгоритму значно зменшився, через оптимізацію квадратичної задачі програмування з обмеженнями. Однак метод опорних векторів є набагато повільнішим, ніж нейронна мережа прямого поширення, оскільки для вирішення задачі квадратичного програмування, в якій кількість змінних дорівнює кількості вхідних змінних, число яких може бути дуже великим. Використовуючи комбінації нелінійних функцій та вибору прихованих шарів, багатошаровий персептрон може підвищити точність моделі.

2.6 Класифікація багатошаровим персептроном

Багатошаровий персептрон було обрано задля того, щоб покращити результат та зменшити час класифікації у порівнянні з методом опорних векторів. Були застосовані нелінійні функції активації для бінарної класифікації, а також приховані шари, щоб підвищити точність.

Вибрано три прихованих шарів. Перший шар містить 50 нейронів, другий 10 нейронів та третій 2 нейрони відповідно. Коефіцієнт навчання для методу швидкого спуску дорівнює 0,01.

Результати якості моделі на тренувальному наборі містяться в таблиці 2.10. Як видно з таблиці 2.10, модель має вищу точність у порівнянні з методом опорних векторів, а саме 99%, що є найвищим показником з усіх використаних в дослідженні моделях. Також f1-оцінка має аж 99%, що є найбільшим показником.

Таблиця 2.10 — Звіт по класифікації на тренувальному наборі даних

	precision	recall	f1-score	support
0	0.99	1.00	1.00	802
1	1.00	0.98	0.99	267
accuracy			0.99	1069
macro avg	1.00	0.99	0.99	1069
weighted avg	0.99	0.99	0.99	1069

Результати якості моделі на тестовому наборі містяться в таблиці 2.11. Точність однакова у порівнянні з результатами на тренувальних даних.

Таблиця 2.11 — Звіт по класифікації на тестовому наборі даних

	precision	recall	f1-score	support
0	0.99	1.00	1.00	810
1	1.00	0.98	0.99	260
accuracy			0.99	1070
macro avg	1.00	0.99	0.99	1070
weighted avg	0.99	0.99	0.99	1070

Отже, модель багатошарового персептрону дала найбільшу точність та f1-оцінку з усіх використаних моделей у дослідженні, а саме 99 відсотків.

Висновки до розділу 2

Отже, в розділі було описано множину даних, яка містить дані про інфікування пацієнтів синдромом набутого імунодефіциту людини. Було побудовано гістограми розподілів, візуалізовано викиди, пораховано у відсотковому співвідношенні інфікованих до неінфікованих пацієнтів. Побудовано матрицю кореляцій за допомогою коефіцієнта кореляції Пірсона. Застосовано метод головних компонент для того, щоб зрозуміти на скільки добре можна розбити множину вхідних даних. Також було порівняно між собою п’ять головних компонент. Було імплементовано методи машинного навчання: логістичну регресію, ядерну логістичну регресію, опорних векторів та багатошаровий персептрон. Обрано для проведення класифікації найоптимальнішу модель багатошаровий персептрон, точність якого становить 99 відсотків на тренувальному та тестовому наборах даних.

ВИСНОВКИ

Для проведення класифікації інфікованих пацієнтів синдромом набутого імунодефіциту було проаналізовано класичну літературу з теорії ймовірностей, математичної статистики та методів машинного навчання [1-3]. На основі статей [4-6] було обрано методи машинного навчання, які будуть використовуватись в експериментах. Опрацьовано літературу, щоб провести первинний аналіз [2]. Задля того, щоб більш глибше проаналізувати взаємозв'язки між ознаками та їх візуалізації, було опрацьовано [11] для подальшої побудови гістограм розподілу, коробчасті діаграми, кількісну діаграму залежної змінної. Проаналізувавши [7], стало можливим візуалізувати основні стовпці з множини даних за допомогою методу головних компонент, незважаючи на те, що множина даних має високу розмірність, а саме 23 незалежних змінних. Значна частина математичних викладок по методам машинного навчання міститься в [3], що значною мірою допомогло у дослідженні, також було виділено основні переваги та недоліки моделей, які використовувались в експерименті.

Загалом перевагою логістичної регресії є її простота в реалізації. Вона здатна інтерпретувати коефіцієнти моделі, як показники важливості ознак множини даних. Але логістична регресія не може класифікувати лінійно нероздільні дані, тому не є оптимальною моделлю для даного дослідження. Крім того, погано працює з мультиколінеарльними даними та є достатньо чутливою до викидів. Щоб класифікувати лінійно нероздільні дані, використовують модифікацію логістичної регресії - ядерну логістичну регресію.

Ядерна логістична регресія досить добре працює з даними високої розмірності та здатна проводити класифікацію для лінійно нероздільних класів за допомогою ядерних методів, які відображають простір ознак в простір більшої розмірності, де дані вже будуть лінійно роздільними. Тим не менш робота алгоритму є обчислювально складною та потребує жорсткої регуляризації, через велику тенденцію до перенавчання. Окрім того, ядерна логістична регресія не

здатна класифікувати дані з найбільшою відстанню між класами, як це , наприклад, робить метод опорних векторів.

Метод опорних векторів теж досить добре працює в просторах з великою кількістю ознак, як це робить ядерна логістична регресія, але крім того, цей метод намагається знайти оптимальну гіперплощину , яка розділяє класи з максимальним проміжком. Більше того цей метод може бути ефективним навіть при невеликій кількості зразків у вибірці, також він здатний підтримувати ядерні методи для класифікації лінійно нероздільних даних. Тим не менш метод опорних векторів досить не ефективно класифікує великі за розміром множини даних, оскільки алгоритм потребує значної кількості пам’яті та часу. Однак, нейронна мережа з прямим поширенням, наприклад, багатошаровий персептрон здатен обробляти великі обсяги даних.

Багатошаровий персептрон здатний до класифікації лінійно нероздільних даних, доволі швидко будує прогнози після навчання, також він може використовувати приховані шари, завдяки яким стає можливим вивчати складні закономірності в даних. Окрім того, багатошаровий персептрон може використовувати нелінійні функції активації в прихованих шарах, що дозволяє йому моделювати складні зв’язки між входами та виходами. Тим не менш багатошаровий персептрон може використовувати високі обчислювальні витрати через його ітеративну природу.

Загалом багатошаровий персептрон було обрано як найоптимальнішу модель для проведення класифікації, через його найвищі значення точності серед усіх моделей, які використовувались у дослідженні. Тож наступним кроком в дослідженні може бути написання застосунку, який на основі моделі багатошарового персептрону буде відповідати на питання, чи є користувач інфікованим синдромом імунодефіциту людини, зі зберіганням повної анонімності особистості.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. William Feller. An introdaction to probability // William Feller – 1968. – Режим доступу: https://www.climet.com/toolbox/feller-correction-calculator/Feller-1968.pdf.
2. Peter Bruce. Practical Statistics for data scientists // Peter Bruce, Andrew Bruce & Peter Gedeck – 2020.
3. Trevor Hastie. The elements of Statistical Learning // Trevor Hastie, Robert Tibshirani, Jerome Friedman– 2009.
4. Mingyang An. Machine Learning Model Based Prediction of Risk Factors for Hypertension in Elderly HIV/AIDS Patients Using Electronic Medical Records // Mingyang An, Chunjie Wang, Wei Pan, Xiao Wang, Qiong Cai, Qian Zhao. – 2024. – Режим доступу: https://www.researchsquare.com/article/rs-3905254/v1.
5. Luana Ibiapina. Application of Data Mining Algorithms for Dementia in People with HIV/AIDS // Luana Ibiapina Cordeiro Calíope Pinheiro , Maria Lúcia Duarte Pereira ,Marcial Porto Fernandez , Francisco Mardônio Vieira Filho ,Wilson Jorge Correia Pinto de Abreu , and Pedro Gabriel Calíope Dantas Pinheiro. – 2021. – Режим доступу: https://onlinelibrary.wiley.com/doi/epdf/10.1155/2021/4602465.
6. Sweta Kumari. Machine learning approaches to study HIV/AIDS infection: A Review // Sweta Kumari, Usha Chouhan and Sunil Kumar Suryawanshi.– 2017. – Режим доступу: https://www.researchgate.net/profile/SherifSharawy/publication/315810299_Evaluation_of_antimicrobial_and_synergistic_effects_of_selected_medicinal_plants_of_Hail_area_with_antibiotics/links/58e780ec4585152528df1c26/Evaluation-of-antimicrobial-and-synergistic-effects-of-selected-medicinal-plants-of-Hail-area-with-antibiotics.pdf#page=44.
7. Zakaria Jaadi. Step-by-step explanation of principal component analysis (PCA). // Zakaria Jaadi. – Режим доступу: https://builtin.com/data-science/step-step-explanation-principal-component

analysis#:~:text=Principal%20component%20analysis%2C%20or%20PCA,information%20in%20the%20large%20set.

8. Ezukwoke K. LOGISTIC REGRESSION AND KERNEL LOGISTIC REGRESSION. A comparative study of logistic regression and kernel logistic regression for binary classification // Ezukwoke K.I., Zareian S.J. – 2019. – Режим доступу: https://www.researchgate.net/publication/337932960_LOGISTIC_REGRESSION_AND_KERNEL_LOGISTIC_REGRESSION_A_comparative_study_of_logistic_regression_and_kernel_logistic_regression_for_binary_classification.
9. Данилов В.Я. Числові алгоритми оптимізації // В.Я. Данилов, П.М. Зінько. – 2013.
10. Michael Nielsen. Neural Networks and Deep Learning. // Michael Nielsen – 2019. – Режим доступу: http://neuralnetworksanddeeplearning.com/index.html.
11. Aurélien Géron. Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems // Aurélien Géron.– 2021. – Режим доступу: https://powerunit-ju.com/wp-content/uploads/2021/04/Aurelien-Geron-Hands-On-Machine-Learning-with-Scikit-Learn-Keras-and-Tensorflow_-Concepts-Tools-and-Techniques-to-Build-Intelligent-Systems-OReilly-Media-2019.pdf
12. Дані для проведення класифікації пацієнтів інфікованих синдромом імунодефіциту людини, які використовувались в експерименті можна отримати за посиланням: https://www.kaggle.com/datasets/aadarshvelu/aids-virus-infectionprediction/data.

ДОДАТОК А

ПРОГРАМНА РЕАЛІЗАЦІЯ ПОПЕРЕДНЬОЇ ОБРОБКИХ ДАНИХ

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier

df = pd.read_csv("../Diploma 2/AIDS_Classification.csv")
df.head()

np.shape(df)
df = df.dropna()
df.shape
df.info()
df.dtypes.value_counts()
print(df.isnull().sum())
df.drop(columns=['infected']).describe()

df.hist(figsize=(20, 20), bins=30, edgecolor='black')
plt.show()
plt.figure(figsize=(20, 10))
sns.boxplot(data=df)
plt.xticks(rotation=90)
plt.show()

print(df['infected'].value_counts())

sns.countplot(x='infected', data=df)
plt.show()

Corr_matrix = df.corr()
plt.figure(figsize = (15,10))
sns.heatmap(Corr_matrix, annot = True, cmap = 'coolwarm', fmt ='.3f', linewidths =
3, linecolor='black'  )
plt.show()
```

ДОДАТОК Б

ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДУ ГОЛОВНИХ КОМПОНЕНТ

```
X = df.drop(columns=['infected'])
y = df['infected']
X = StandardScaler().fit_transform(X)
X = np.array(X)
y = np.array(y)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,
random_state = 40)
pca = PCA(n_components=5, random_state=42)
principal_components = pca.fit_transform(X)
pca_df = pd.DataFrame(data=principal_components, columns=[f'PC{i+1}' for i in
range(5)])
pca_df['infected'] = df['infected']
sns.pairplot(pca_df, hue='infected', markers=["o", "s"])
plt.show()
plt.figure(figsize=(10, 7))
sns.scatterplot(data=pca_df, x='PC1', y='PC2', hue='infected', palette='viridis',
s=100, alpha=0.7)
plt.title('PCA: PC1 vs PC2')
plt.show()
plt.figure(figsize=(10, 7))
sns.scatterplot(data=pca_df, x='PC2', y='PC3', hue='infected', palette='viridis',
s=100, alpha=0.7)
plt.title('PCA: PC2 vs PC3')
plt.show()

# Графік розсіювання для PC3 та PC4
plt.figure(figsize=(10, 7))
sns.scatterplot(data=pca_df, x='PC3', y='PC4', hue='infected', palette='viridis',
s=100, alpha=0.7)
plt.title('PCA: PC3 vs PC4')
plt.show()

# Графік розсіювання для PC4 та PC5
plt.figure(figsize=(10, 7))
sns.scatterplot(data=pca_df, x='PC4', y='PC5', hue='infected', palette='viridis',
s=100, alpha=0.7)
plt.title('PCA: PC4 vs PC5')
plt.show()
```

ДОДАТОК В
ПРОГРАМНА РЕАЛІЗАЦІЯ ЛОГІСТИЧНОЇ РЕГРЕСІЇ

```
class My_LogisticRegression:
    def __init__(self, l):
        self.l = l

    def fit(self, X, y):
        X = np.hstack((np.ones( (X.shape[0] , 1)), X)) # додаємо до матриці X
стовпець одиниць
        N = len(y)
        betta_old = np.zeros(X.shape[1])
        W = np.zeros((N, N))
        for i in range(N):
            exp_term = np.exp(np.dot(betta_old, X[i]))
            W[i, i] = exp_term / ((1 + exp_term) * (1 + exp_term))
        y_hat = np.exp(np.dot(X, betta_old)) / (1 + np.exp(np.dot(X, betta_old)))
        betta_new = betta_old + np.dot(np.dot(np.linalg.inv(np.dot(np.dot(X.T, W),
X)), X.T), (y - y_hat))
        while np.linalg.norm(betta_new - betta_old) > self.l:
            betta_old = betta_new
            y_hat = np.exp(np.dot(X, betta_old)) / (1 + np.exp(np.dot(X,
betta_old)))
            W = np.diag(y_hat * (1 - y_hat))
            betta_new = betta_old + np.dot(np.dot(np.linalg.inv(np.dot(np.dot(X.T,
W), X)), X.T), (y - y_hat))
        return betta_new

    def predict(self, X, betta_estimated): # y_hat = P(G=1 | X=x; betta^) =
exp(betta.T * X)/(1 + exp(betta.T * X))
        X = np.hstack((np.ones( (X.shape[0] , 1)), X)) # додаємо до матриці X
стовпець одиниць
        y_hat = np.exp(np.dot(X, betta_estimated)) / (1 + np.exp(np.dot(X,
betta_estimated)))
        return np.where(y_hat > 0.5, 1, 0).astype(int)   # Якщо y_hat >= 0.5 то
повертаємо 1 інакше 0
my_logreg = My_LogisticRegression(l=10e-8)
betta_hat = my_logreg.fit(X_train, y_train)
y_train_hat = my_logreg.predict(X_train, betta_hat)
print(classification_report(y_train_hat, y_train))
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
y_hat = logreg.predict(X_train)
print(classification_report(y_hat, y_train))
my_logreg = My_LogisticRegression(l=10e-8)
betta_hat = my_logreg.fit(X_test, y_test)
y_test_hat = my_logreg.predict(X_test, betta_hat)
print(classification_report(y_test_hat, y_test))
logreg = LogisticRegression()
logreg.fit(X_test, y_test)
y_hat = logreg.predict(X_test)
```

ДОДАТОК Г
ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДУ ОПОРНИХ ВЕТОРІВ І БАГАШАРОВОГО ПЕРСЕПТРОНУ

Лістинг Г.1. Метод опорних векторів

```
svc=SVC(kernel='rbf', C=1, gamma=0.1)
svc.fit(X_train,y_train)
y_train_hat = svc.predict(X_train)
print(classification_report(y_train_hat, y_train))
svc=SVC(kernel='rbf', C=1, gamma=0.1)
svc.fit(X_test, y_test)
y_test_hat = svc.predict(X_test)
print(classification_report(y_test_hat, y_test))
```

Лістинг Г.2. Багатошаровий персептрон

```
Model = MLPClassifier(hidden_layer_sizes=(50, 10, 5),
                      random_state=5,
                      verbose=True,
                      learning_rate_init=0.01)
Model.fit(X_train, y_train)
y_hat = Model.predict(X_train)
print(classification_report(y_hat, y_train))
Model = MLPClassifier(hidden_layer_sizes=(50, 10, 5),
                      random_state=5,
                      verbose=True,
                      learning_rate_init=0.01)
Model.fit(X_test, y_test)
y_hat = Model.predict(X_test)
print(classification_report(y_hat, y_test))
```