Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

Магістр
(назва освітнього ступеня)

на тему: Математичне та програмне забезпечення прогнозування фінансових показників компаній на основі методів машинного навчання та моделей штучних нейромереж

Виконав: студент 6 курсу, групи СНм-61
спеціальності
122 Комп’ютерні науки
(шифр і назва спеціальності)
_____ Кучеренко О.А.
(підпис) (прізвище та ініціали)

Керівник _____ доц. Гром’як Р.С.
(підпис) (прізвище та ініціали)
Нормоконтроль _____ доц. Дуда О.М.
(підпис) (прізвище та ініціали)
Завідувач кафедри _____ доц. Боднарчук І.О.
(підпис) (прізвище та ініціали)
Рецензент _____ Жаровський Р.О.
(підпис) (прізвище та ініціали)

м. Тернопіль – 2023

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ доц. Боднарчук І.О.
(підпис) (прізвище та ініціали)
« » 20___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня магістр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

студенту Кучеренку Олексію Анатолійовичу

1. Тема роботи Математичне та програмне забезпечення прогнозування фінансових показників компаній на основі методів машинного навчання та моделей штучних нейромереж

Керівник роботи Гром'як Роман Сильвестрович, к.ф.-м.н., доц. каф. КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом по університету від «__24__» _листопада_ 2023 року №__4/7-1099_

2. Термін подання студентом роботи 25.12.2023

3. Вихідні дані до роботи наукові літературні джерела

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1 Аналіз предметної області дослідження. 2. Методи машинного навчання та моделі штучних нейромереж. 3. Практична частина.
4. Охорона праці та безпека в надзвичайних ситуаціях

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Тема роботи. 2. Актуальність. 3. Мета, задачі дослідження. 4. Об'єкт, предмет дослідження наукова новизна, практичне значення роботи. 5. Дані, котрі використані в дослідженні. 6. Показники рентабельності компанії. 7. ПЗ, яке використовувалося в роботі. 8. Етапи попередньої обробки даних у машинному навчанні. 9. Методи машинного навчання та моделі штучних нейромереж, які використані в роботі. 10. Дерево рішень, випадковий ліс. 11. Наївний Баєс, стохастичний градієнтний спуск. 12. XGBoost, метод k-найближчих сусідів 13. Логістична регресія, лінійна регресія. 14. Метод опорних векторів. 15. Штучна нейронна мережа. 16. Архітектура програмного засобу. 17. Інтерфейс системи. 18. Діаграма варіантів використання. 19. Зведена таблиця результатів роботи моделей. 20.Основні результати дослідження

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Сенчишин В.С., доцент каф. МТ		
Безпека в надзвичайних			
ситуаціях	Клепчик В.М., проректор з АГРБ		

7. Дата видачі завдання _____ 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Затвердження теми кваліфікаційної роботи	24.11.23	Виконано
2	Аналіз літературних джерел	25.11–28.11.23	Виконано
3	Обґрунтування актуальності дослідження	29.11–01.12.23	Виконано
4	Аналіз предмету дослідження та предметної області	01.12–03.12.23	Виконано
5	Проведення дослідження методів та засобів аналітичного опрацювання даних	04.12–06.12.23	Виконано
6	Оформлення розділу «Аналіз предметної області дослідження»	07.12–10.12.23	Виконано
7	Оформлення розділу «Методи машинного навчання та моделі штучних нейромереж»	11.12–13.12.23	Виконано
8	Оформлення розділу «Практична частина»	14.12–15.11.23	Виконано
9	Оформлення розділу «Охорона праці та безпека в надзвичайних ситуаціях»	05.12–12.12.23	Виконано
10	Нормоконтроль	13.12–15.12.23	Виконано
11	Перевірка на плагіат	13.12–15.12.23	Виконано
12	Попередній захист роботи	20.12.23	Виконано
13	Захист кваліфікаційної роботи	26.12.23	

Студент _____ Кучеренко О.А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Гром’як Р.С.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Математичне та програмне забезпечення прогнозування фінансових показників компаній на основі методів машинного навчання та моделей штучних нейромереж // Кваліфікаційна робота освітнього рівня «Магістр» // Кучеренко Олексій Анатолійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра комп'ютерних наук, група СНм–61 // Тернопіль, 2023 // С. – 72, рис. – 26 , табл.– 13, слайдів – 20, додат. – 1, бібліогр. – 28.

Ключові слова: ПЕРЕДОБРОБКА ДАНИХ, ПРОГНОЗУВАННЯ, ШТУЧНА НЕЙРОМЕРЕЖА, КЛАСИФІКАТОР, МАШИННЕ НАВЧАННЯ, ФІНАНСОВІ ПОКАЗНИКИ

Кваліфікаційна робота присвячена розробці програмного засобу для прогнозування фінансових показників компаній.

Описано специфіку даних, які використовувалися при виконанні дослідження. Докладно проаналізовано поняття попередньої обробки даних для машинного навчання та особливості її використання для методів прогнозування. Здійснено критичний огляд проблеми незбалансованості даних при проведенні інтелектуального аналізу даних та ймовірних шляхів її вирішення. Проаналізовані наявні методи машинного навчання та моделі штучних нейронних мереж. Наведено особливості використання методів та моделей на мові pytnon (із додатковими бібліотеками), приведені параметри налаштування, значення оцінок на збалансованих та незбалансованих даних.

Наведено загальну архітектуру програмного засобу для прогнозування. Здійснено аналіз усіх моделей. Проведені експерименти доводять, що для більшості моделей отримані кращі результати для збалансованих даних.

ANNOTATION

Mathematical maintenance and software for forecasting the financial inicators of companies based on machine learning methods and artificial neural network models // Kucherenko Oleksii // Ternopil Ivan Pul'uj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science // Ternopil, 2023 // P. - 72, Fig. - 26, Table – 13, Slide - 20, References - 28.

Keywords: DATA PREPROCESSING, FORECASTING, ARTIFICIAL NEURAL NETWORK, CLASSIFIER, MACHINE LEARNING, FINANCIAL INDICATORS

Thesis deals with the development of a software tool for forecasting financial indicators of companies.

The specifics of the data used in the research are described. The concept of data preprocessing for machine learning and the features of its use for forecasting methods are analyzed in detail. A critical review of the problem of data imbalance during intelligent data analysis and possible ways to solve it was carried out. Existing methods of machine learning and models of artificial neural networks are analyzed. Features of the use of methods and models in the python language (with additional libraries), setting parameters, values of estimates on balanced and unbalanced data are given.

The general architecture of the software tool for forecasting is given. All models were analyzed. The conducted experiments prove that for most models better results are obtained for balanced data.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ СКОРОЧЕНЬ І ТЕРМІНІВ

ANN (Artificial Neural network), ШНМ (штучна нейронна мережа) – математична модель, а також її програмне або апаратне втілення, побудована за принципом організації та функціонування біологічних нейронних мереж - мереж нервових клітин живого організму.

AUC (Area Under Curve) – площа під кривою, статистичний показник.

FPR (false positive rate) – хибнопозитивне відношення.

Kaggle – платформа для змагань з аналітики та передбачувального моделювання.

KNN (k-nearest Neighbors) – метод найближчих сусідів.

ML (Machine Learning) – машинне навчання.

ROC (Receiver Operating Characteristic) – крива, робоча характеристика приймача.

SMOTE (Synthetic Minority Over-sampling Technique) – алгоритм попередньої обробки даних, який використовується для усунення дисбалансу класів у наборі даних.

SGD (Stochastic Gradient Descent) – стохастичний градієнтний спуск.

SVM (Support Vector Machines) – метод опорних векторів.

TPR (True Positive Rate) – істинопозитивне відношення.

БД – база даних.

ІАД – інтелектуальний аналіз даних.

НД – набір даних.

ПК – персональний комп’ютер.

ПЗ – програмне забезпечення.

ПОД – попередня обробка даних.

ПФПК – прогнозування фінансових показників компанії.

ШІ – штучний інтелект.

ЗМІСТ

ВСТУП

Актуальність теми. На даний час серйозною проблемою є ПФПК, складання різних моделей прогнозування, їхнього порівняння, та на їх основі здійснення вибору найточнішої моделі ПФПК.

Існуючі програми ПФПК спрямовані на отримання комерційного прибутку, а алгоритми побудови моделі прогнозування приховані від користувача. Саме тому виникає необхідність реалізації програми з доступними для розуміння алгоритмами побудови моделей. Доступність розуміння роботи моделей прогнозування досягається за рахунок графічного представлення моделей та прикріплених до кожної моделі теоретичних матеріалів. Завдяки цьому програму можна використовувати в освітніх цілях як практичне застосування алгоритмів, що вивчаються, чого не вистачає в існуючих аналогах, що використовуються в комерційних цілях. Користувачеві доступний не тільки вибір моделі зі списку запропонованих, але й зміна параметрів моделі для збільшення точності прогнозування.

Варто також відзначити, що існуючі програми та методології в ПФПК здебільшого ґрунтуються на економетричних підходах [1]. В системі, що розробляється, наголошується на використанні методів ML, а також ШНМ. Такий підхід у прогнозуванні дозволить виявити приховані закономірності у тенденціях фінансових показників компаній, а також зробить свій внесок у популяризацію використання методів ML.

Мета дослідження: створення програмного засобу для побудови моделей ПФПК.

Для досягнення мети, в роботі поставлено та розв’язано **такі задачі:**

- вивчити існуючі технології прогнозування;
- виконати передобробку даних;
- вивчити особливості фінансових рядів, роботу з бібліотеками Python по роботі з тимчасовими рядами, практика роботи Jupiter Notebook;

– побудувати різні моделі прогнозування, у тому числі за допомогою ШНМ, із застосуванням методів ML та економетрики у прогнозуванні фінансових показників;

– реалізувати та порівняти регресійні методи прогнозування з економетричними одновимірними моделями часових рядів.

Об’єкт дослідження: моделі даних про прогнозування фінансових показників компаній.

Предмет дослідження: технології прогнозування фінансових показників компаній.

Наукова новизна роботи:

– для вирішення задачі балансування класів у наборі даних запропоновано впровадити в систему алгоритм SMOTE;

– виконано порівняльний аналіз отриманих результатів точності досліджуваних моделей на збалансованих та незбалансованих даних;

– сформовано методологічний посібник, який дозволить впровадити реалізовані моделі в інші проекти.

Практичне значення одержаних результатів. Окрім, власне, можливості реального застосування на підприємствах, при потенційній масштабованості, як потужного бізнес-інструменту, систему можна рекомендувати до використання як інтерактивний навчальний посібник для тих, хто тільки починає своє знайомство з ІАД, оскільки містить не тільки теоретичну частину, але й приклади коду мовою Python.

Апробація. Окремі результати роботи були представлені на XI науково-технічній конференції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя (13-14 грудня 2023 р.) у вигляді опублікованих тез [2].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ДОСЛІДЖЕННЯ

1.1 Дані, використані в дослідженні

Для кваліфікаційної роботи застосовуються дані платформи Kaggle, котра містить відкриту інформацію різних конкурсних задач у галузі аналізу даних [3]. Цей сайт дає змогу пропонувати свої рішення назагал, проводити обговорення процесу аналізу та змагання щодо точності сформованих моделей. Застосвані дані із вибірки "Financial Distress Prediction: Bankruptcy Prediction". НД, котрий призначений для прогнозування фінансових бід для вибірки підприємств, носить назву контексту даних. Окремі стовпці з даними наведено на рис. 1.1.

	Company	Time	Financial Distress	x1	x2	x3	x4
0	1	1	0.010636	1.2810	0.022934	0.87454	1.21640
1	1	2	-0.455970	1.2700	0.006454	0.82067	1.00490
2	1	3	-0.325390	1.0529	-0.059379	0.92242	0.72926
3	1	4	-0.566570	1.1131	-0.015229	0.85888	0.80974
4	2	1	1.357300	1.0623	0.107020	0.81460	0.83593
...	...	...	...	...	...	...	...

Рисунок 1.1 – НД фінансових показників різних компаній

Наведемо короткий опис стовпців

Перший – це ідентифікатор підприємства.

Другий – відображає різні часові проміжки, котрим дані належать. Довжина такого проміжку змінюється від 1 до 14 для кожного підприємства.

Третій – змінна мети позначається як «Фінансова криза» (коли її значення більше, ніж -0,5, то підприємство вважається рентабельним (приймається 0). Всі інші випадки вважаються, що компанія є фінансово неблагополучною

(приймається 1).

Починаючи від четвертого і аж до останнього стовпця містяться певні характеристики, котрі позначені від x1 до x83, це певні як фінансові, так і нефінансові характеристики визначений підприємств. Вони стосуються минулого часового періоду, котрий варто застосовувати для прогнозування власне того, буде підприємство зазнавати фінансових бід чи ні.

НД є незбалансованим. Варто зазначити, що 30% цього НД варто випадково визначити як набір тестових даних, а 70% котрі залишилися, застосовуються для визначення функцій і селекції моделі [1].

Також дані містять примітки для стовпців.

Перший – дані можна вважати проблемою класифікації.

Другий - дані також можуть вважатися проблемою регресії, а після цього результат буде трансформовано у класифікацію.

Третій – такі дані можуть вважатися багатовимірною класифікацією часових рядів.

Головні проблеми даних для їх дослідження:

– котрі особливості найкраще говорять про фінансові труднощі?

– які моделі ML щонайліпше працюють із саме цим НД?

Слід зазначити, що дані, які використовуються у дослідженні, незбалансовані.

1.2 Показники рентабельності компанії

У НД основними стовпцями для аналізу є набір ознак x1, x2, ... x79, x81, …, x83. Ці ознаки є деякими характеристиками стану підприємства за певний час. На жаль, розрахунок цих показників прихований, що накладає деякі обмеження для можливого масштабування програми ПФПК, що розробляється, тобто при потенційному розширенні функціоналу програми до калькулятора рентабельності компаній окремих користувачів, у аналітика буде недостатньо

інформації про аналізовані дані. Тому гостро постає питання вивчення відомих показників рентабельності компаній, які носять іншу назву коефіцієнтів рентабельності [4].

Вони є фінансовими показниками, котрі застосовують аналітики та інвестори для визначення та оцінювання можливості компанії продукувати дохід (прибуток) щодо виручки, балансових активів, операційних витрат та власного капіталу упродовж деякого часового діапазону. Ці коефіцієнти свідчать, як добре компанія застосовує свої активи для одержання доходу та підвищення цінності для акціонерів [4].

Майже всі компанії, як правило, досягають вищого коефіцієнта або вартості, такт як це, зазвичай, свідчить, що бізнес працює добре за рахунок генерування доходів та коштів. Коефіцієнти найбільш корисні, коли вони аналізуються у порівнянні з такими ж самими компаніями чи минулими часовими рамками.

Використовуються різноманітні коефіцієнти рентабельності, які застосовуються компаніями для отримання якісних даних щодо фінансового стану та ефективності бізнесу. Всі ці співвідношення можна поділити на дві категорії.

Коефіцієнти маржі – показують можливість конвертувати продаж у прибуток за різноманітних степенях виміру.

Коефіцієнти прибутковості – демонструють можливість компанії давати прибуток.

Варто приділити більше уваги розгляду тих показників рентабельності, котрі найбільше застосовуються.

Рентабельність капіталу. Він поділяється на:

– рентабельність сукупного капіталу. Показник, який особливо цікавий для біржових аналітиків та інвесторів. Цей сприятливо високий коефіцієнт часто згадується як причина на купівлю акцій компанії. Фірми з високою рентабельністю, зазвичай, більш здатні заробляти кошти в фірмі, а тому є слабо

залежними від боргового фінансування. Обчислюється за формулою (1.1):

$$R_{ROA} = \frac{\text{Прибуток до оподаткування}}{\text{Всього джерел засобів}} \quad (1.1)$$

– рентабельність власного капіталу. Дає остаточну картину прибутковості фірми після урахування всіх витрат, в т.ч. з відсотками та податками. Причина застосування чистого прибутку як міри рентабельності у тому, що тут враховується все. Недоліком цього показника є те, що він включає багато «шуму», такого як одноразові витрати і прибуток, що утруднює зіставлення підсумків діяльності фірми та конкурентів. Для розрахунку цього показника застосовується формула (1.2):

$$R_{ROE} = \frac{\text{Чистий прибуток}}{\text{Власний капітал}} \quad (1.2)$$

Рентабельність продажів. Її можна розділити на кілька коефіцієнтів, що вираховуються.

Валова рентабельність реалізованого товару. Зіставляє валовий прибуток із виручкою від продажів. Демонструє, скільки заробляє фірма із урахуванням необхідних витрат за виготовлення послуг чи певних товарів. Великий показник говорить про вищу ефективність проведення основних операцій, що означає, що він, як і раніше, може покривати операційні витрати, фіксовані витрати, дивіденди та амортизацію, а також забезпечувати отримання чистого прибутку для фірми. Але малий прибутку вказує на велику ціну проданих товарів, що може бути пов'язане з поганою політикою проведення власне закупівель, низькими продажними цінами, низькими продажами, жорсткою ринковою конкуренцією чи неграмотною політикою стимулювання властиво збуту. Для розрахунку застосовується формула (1.3):

$$R_{GPM} = \frac{\text{Валовий прибуток}}{\text{Виручка від реалізації}} \quad (1.3)$$

Операційна рентабельність реалізованого товару. Компанії з високим операційним прибутком, як правило, краще оснащені для оплати фіксованих витрат і % за зобов'язаннями, володіють значно кращими шансами перебути економічний спад і мають змогу призначати менші ціни, аніж їхні конкуренти, котрі мають нижчу маржу прибутку.

$$R_{OIM} = \frac{\text{Операційний прибуток}}{\text{Виручка від реалізації}} \quad (1.4)$$

Чиста рентабельність реалізованого товару. Дає остаточну версію прибутковості підприємства після врахування всіх витрат, разом із % та податками. Для розрахунку показника застосовується формула (1.5):

$$R_{NPM} = \frac{\text{Чистий прибуток}}{\text{Виручка від реалізації}} \quad (1.5)$$

1.3 Модель Дюпона

Крім всіх згаданих показників рентабельності компаній, також застосовується модель Дюпона — це глибше аналізування фінансового стану фірми. Береться для оцінювання ROE фірми. Це дає змогу інвестору зрозуміти, котрий вид фінансової діяльності є найкращим для зміни рентабельності власного капіталу. Він може застосувати подібний розгляд для співставлення операційної ефективності двох аналогічних фірм. Працівники можуть застосувати його для визначення сильних та слабких боків, котрих необхідно позбутися.

Загалом формула вимірює сукупний вплив розміру прибутку та

оборотності активів. Розраховується за формулою (1.6):

$$\mathrm{ROA} = \frac{\text{Net Income}}{\text{Revenue}} \times \frac{\text{Revenue}}{\text{Average Total Assets}} = \frac{\text{Net income}}{\text{Average Total Assets}} \quad (1.6)$$

де: Net Income - Чистий дохід компанії, Revenue - Виручка, Average Total Assets – Активи компанії.

Рентабельність власного капіталу, в порівнянні з ринковою вартістю, враховуючи власне структурні ризики та динаміку прибутку, що демонструє частку впливів ззовні, формує додаткову оцінку [5]. Слід також зазначити про потребу врахування специфіки окремої галузі.

Варто відзначити, що всі наведені вище коефіцієнти, при потенційному масштабуванні програми в систему розрахунку рентабельності компанії, можуть бути використані як коефіцієнти рентабельності. При цьому, звичайно, для коректного навчання слід вибрати один із способів розрахунку цього показника.

1.4 Технології та бібліотеки, що використовуються в роботі

Для основного дослідження даних застосовується мова Python. Мова мінімалістична і досить популярна за рахунок цього існує величезна кількість бібліотек, які допомагають у роботі.

Основною платформою для роботи визначено Jupyter Notebook. Це дуже зручний інструмент для створення зручних, візуально зрозумілих аналітичних звітів. Інтерфейс програми дозволяє компонувати документ, що поєднує у собі програмний код, зображення, коментарі, формули та графіки.

Notebook також включає велику базу бібліотек, як графічних, так і математичних, за допомогою цих бібліотек Jupiter дозволяє створювати інтерактивні графіки і докладні рішення.

Для веб-представлення моделей прогнозування використовувався інструмент Streamlit. Streamlit перетворює сценарії даних на веб- додатки для спільного використання за лічені хвилини, використовуючи мову Python.

Streamlit – це фреймворк з відкритим кодом, спеціально розроблений для інженерів ML, які працюють з Python. Він дозволяє створювати стильні програми завдяки буквально кільком рядкам коду. Streamlit дозволяє розробити дашборд для інтерактивного представлення моделей машинного чи нейронного навчання. Завдяки цій платформі розробка веб-інтерфейсу стає зручною та простою.

Нижче наведено список основних використовуваних бібліотек при роботі з даними:

– matplotlib.pyplot - заснований на стані інтерфейсу до matplotlib. Він забезпечує побудову графіків. pyplot в основному призначений для інтерактивних сюжетів та простих моментів генерації програмних сюжетів;

– plotly.offline – застосовується для побудови графіків, підтримує більше 40 неповторних діаграмних типів, котрі представляють велике коло статистичних, фінансових, географічних, наукових та тривимірних сценаріїв застосування;

– plotly.graph - бібліотека для графічного представлення даних;

– statsmodels.api - є пакетом Python, який забезпечує доповнення до scipy для статистичних обчислень, включаючи описову статистику та оцінку;

– scipy.stats - всі статистичні функції перебувають у підпакеті scipy.stats, можна отримати досить повний список цих функцій;

– sklearn.metrics - метрики оцінки для відбору моделей у scikit-learn;

– scikit-learn – безкоштовна бібліотека ML для Python;

– Keras – відкрита бібліотека на Python для роботи з нейромережами;

– TensorFlow – відкрита бібліотека для ML, розроблена Google.

1.5 Передобробка даних

У даних представлено 422 номери компаній, а фінансові дані представлені у вигляді коротких часових рядів для кожної компанії. У поточній ПОД інформація про час ігнорується, і кожен рядок розглядається як незалежна точка даних.

Першим кроком під час розгляду даних є вивчення кореляції [6]. Вивчення кореляції змінних досягається за допомогою виміру фінансового лиха. Для цього необхідно позбутися категоріальної змінної, що позначається як стовпець "80x", це таргет даних. На цьому етапі поточний віддалений стовпець буде єдино віддаленим, оскільки інші можуть бути відкинуті після кореляції як незначні.

Функції мають бути масштабовані за допомогою стандартного засобу масштабування перед використанням для навчання або тестування.

1.6 Особливості передобробки даних для методів прогнозування

ПОД у ML - це важливий крок, який допомагає підвищити якість даних та сприяти вилученню з них змістовної інформації. Фактично - це метод підготовки (очищення та систематизації) необроблених даних, щоб зробити їх придатними для побудови та навчання моделей ML. Простіше кажучи, ПОД у ML - це метод ІАД, котрий трансформує необроблені дані в зрозумілий формат.

Коли йдеться про побудову моделі ML, така ПОД є першим кроком, що знаменує початок процесу. Як правило, реальні дані є неповними, непослідовними, неточними (містять помилки чи викиди) та часто не мають конкретних значень/тенденцій атрибутів [7]. Саме тут у сценарій входить ПОД - вона допомагає очищати, форматувати і систематизувати необроблені дані, тим самим роблячи їх готовими до використання в моделях ML.

Можна схематично виділити етапи ПОД у ML [2]:

– отримання даних. Щоб побудувати та розробити моделі ML, необхідно спочатку отримати відповідну кількість даних. Вона буде складатися з даних, зібраних з кількох та розрізнених джерел, які потім будуть об'єднані у належному форматі для формування. Формати таких власне НД різняться залежно від застосування сценарію. Для прикладу, набір бізнес-даних повністю відрізнятиметься від НД для медицини;

– імпорт бібліотеки. Бібліотеки, котрі застосовуються, описані у використовуваних технологіях. Найбільш використовуваними бібліотеками при обробці даних є бібліотеки pandas та numpy;

– імпорт даних. На цьому етапі необхідно імпортувати НД, зібраних для поточного проекту ML. У процесі імпорту НД необхідно витягти залежні та незалежні змінні. Для кожної моделі ML необхідно розділити незалежні змінні (матрицю функцій) та залежні змінні НД. Щоб отримати незалежні змінні, необхідно використовувати функцію iloc[] бібліотеки Pandas;

– виявлення та обробка відсутніх значень. При ПОД дуже важливо ідентифікувати та правильно обробляти відсутні значення, в іншому випадку ви можете зробити неточні та помилкові висновки на основі даних. Зайве говорити, що це завадить вашому проекту ML. У БД немає пропущених значень, тому цей крок опускається;

– кодування категоріальних даних. Вони належать до інформації, що має певні категорії в НД. У згаданому вище НД є дві категоріальні змінні - країна та покупка. Моделі ML, в основному, ґрунтуються на математичних рівняннях. У НД категоріальної змінної є стовпець з інформацією про рентабельність підприємства;

– розділення НД. Будь-який НД для моделі ML повинен бути поділеним на два окремих НД - навчальний і тестовий. Перший – це підмножина НД, котра застосовується для навчання моделі ML [8]. Другий – це підмножина НД для тестування моделі ML. Зазвичай НД розбивається на співвідношення 70% та 30% або співвідношення 80% та 20%. Процес поділу

залежить від форми та розміру аналізованого НД;

– масштабування функцій. Означає кінець ПОД у ML. Це метод стандартизації незалежних змінних НД у межах певного діапазону.

Варто відзначити, що всі ці кроки ПОД виділено в окремий розділ модулю прогнозування рентабельності. Він містить не тільки опис кроків, але й прикріплений програмний код.

1.7 Проблема незбалансованості даних

Дуже часто безпосереднє використання стандартних моделей призводить до результатів низької якості, причиною таких результатів найчастіше стають дані, а саме їх незбалансованість. Тому основним і ключовим етапом у процесі ПОД було їхнє балансування.

Дані, що використовуються для роботи, є незбалансованими, що згодом може сильно вплинути на результат прогнозу моделей [1].

Незбалансовані дані можуть призвести до феномену точності. Цей парадокс для прогнозної аналітики свідчить, що прогнозні моделі з меншим рівнем точності можуть мати більшу прогностичну силу, ніж моделі з вищою точністю.

Точність часто є відправною точкою для аналізу якості моделі, що прогнозує, а також очевидним критерієм прогнозування. Точність вимірює ставлення правильних прогнозів загальної кількості оцінених випадків. Може здатися очевидним, що це співвідношення має бути ключовим показником. Однак, прогностична модель може мати високу точність, але бути марною.

На рис. 1.2 представлений відбір рентабельних та збиткових компаній.

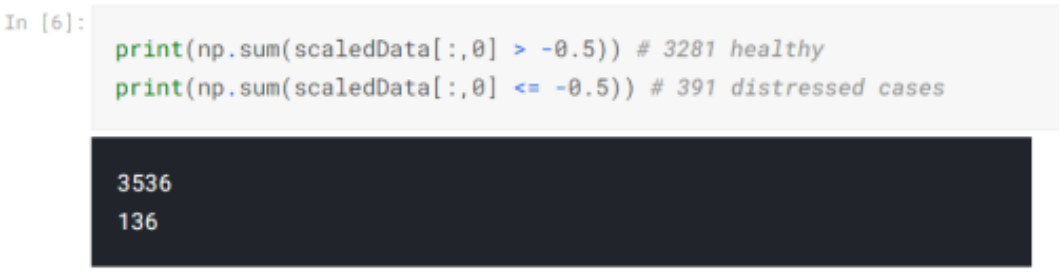

Рисунок 1.2 – Результат відбору рентабельних та збиткових компаній

Дані, що використовуються, сильно незбалансовані, дисбаланс у НБ значний. Лише близько 5% представлених компаній у вибірці збиткові. Цей факт може сильно вплинути на результати оцінки моделей, тому гостро постає питання балансування даних.

На практиці існує кілька способів усунення дисбалансу класів:

– перевищення вибірки класу меншості.

– зменшення вибірки класу більшості.

– використання готових параметрів типу dass_weight, наданих у бібліотеках ML.

Нижче будуть розглянуті кожен із варіантів.

1.7.1 Перевищення вибірки класу меншості

Найпростішим способом згенерувати синтетичні вибірки на вирішення вибірки класу меншини - це випадкова вибірка атрибутів з екземплярів у меншості. Необхідно вибрати їх емпірично в НБ або використовувати такий метод, як Наївний Байєс, який може вибирати кожен атрибут незалежно при зворотному запуску.

Таким чином згенеруються більше різних даних, але нелінійні відносини між атрибутами не можуть бути збережені.

Існують систематичні алгоритми [9], які можна використовувати для генерації синтетичних зразків. Найпопулярніший з таких алгоритмів називається SMOTE. Є методом передискретизації. Він працює, створюючи

синтетичні зразки із молодшого класу замість створення копій. Алгоритм вибирає два або більше однакових екземплярів (використовуючи міру відстані) і збуджує один атрибут екземпляра за раз на випадкову величину в межах різниці з сусідніми екземплярами.

1.7.2 Зменшення вибірки класу більшості

Найпростіший і найгрубіший спосіб балансування даних. При виборі цього рішення про незбалансованість даних використовується обрізка даних, тобто видаляються рядки даних - різниця між класами. Такий спосіб неможливий у розрізі великого дисбалансу, тому що доводиться видаляти велику кількість даних, і розмір вибірки стає занадто малим для подальшого аналізу.

1.7.3 Використання готових параметрів типу class_weight, наданих у бібліотеках машинного навчання

При виборі цього способу необхідно змінити не дані, а оцінку моделей. Тоді для аналізу моделей не використовується оцінка Mean accuracy (співвідношення правильної кількості класифікованих об'єктів до загального розміру навчальної вибірки), на яку сильно впливає дисбаланс класу.

Для аналізу моделей використовується точність, відгук (recall) та FScore, засновані на використання параметра class_weight у моделях прогнозування.

Іншими словами, можна вирішити проблему дисбалансу класів, змінивши саму функцію втрат, щоб надати більшої ваги класу меншості.

У роботі використовуються метод SMOTE. Це один з методів передискретизації, що найчастіше використовуються, для вирішення проблеми дисбалансу. Він спрямований на балансування розподілу класів шляхом випадкового збільшення прикладів меншин класів шляхом їх відтворення.

SMOTE синтезує нові екземпляри меншості між існуючими екземплярами меншини. Він генерує віртуальні навчальні записи шляхом лінійної інтерполяції

для меншості. Ці синтетичні навчальні записи генеруються шляхом випадкового вибору одного або кількох k -найближчих сусідів для кожного прикладу у меншості. Після процесу передискретизації дані відновлюються, і до оброблених даних можна застосувати кілька моделей класифікації.

Далі подано алгоритм роботи SMOTE.

Крок 1: встановлюється набір класів меншин A, для кожного x , що належить множині A, k -найближчі сусіди x виходять шляхом обчислення евклідової відстані між x і кожним іншим зразком у наборі A.

Крок 2: частота дискретизації N встановлюється відповідно до незбалансованої пропорції. Для кожного x множині A, N прикладів (тобто x_1 , x_2 ,..., x_n) випадковим чином вибираються з k- найближчих сусідів, і вони складають набір A_k .

Крок 3: для кожного прикладу x_k множини A_1 (k = 1, 2, 3… N), застосовується наступна формула для створення нового прикладу:

$$x' = x + rand\ (0,1) * |x - x_k| \quad \text{.....(1.1)}$$

де *rand* (0,1) - являє собою випадкове число від 0 до 1.

Існують готові реалізації алгоритму SMOTE на python мові , далі на рис. 1.3 представлено використання бібліотеки imbleam.over_sampling.

```
from imblearn.over_sampling import SMOTE

counter = Counter(y_train)
print('Before',counter)
# oversampling the train dataset using SMOTE
smt = SMOTE()
#X_train, y_train = smt.fit_resample(X_train, y_train)
X_train_sm, y_train_sm = smt.fit_resample(X_train, y_train)

counter = Counter(y_train_sm)
print('After',counter)
```

```
Before Counter({0: 18497, 1: 4208})
After Counter({0: 18497, 1: 18497})
```

Рисунок 1.3 – Застосування алгоритму SMOTE мовою python.

1.8 Висновки до першого розділу

У цьому розділі описано особливості даних, котрі застосовані при проведенні дослідження. Наведено основні коефіцієнти рентабельності. Також приведені основні програмні технології та бібліотеки, котрі використані в роботі.

Докладно описано поняття ПОД для ML та специфіку її використання для методів прогнозування.

Виконано огляд проблеми незбалансованості даних при проведенні ІАД та можливих шляхів її вирішення.

2 МЕТОДИ МАШИННОГО НАВЧАННЯ ТА МОДЕЛІ ШТУЧНИХ НЕЙРОМЕРЕЖ

ML – є класом методів ШІ, котрий характеризується не прямим вирішенням задачі, а власне навчанням, використовуючи рішення багатьох схожих задач [10]. Для формування таких методів застосовуються можливості математичної статистики, чисельних методів, математичного аналізу, методів оптимізації, теорії ймовірностей, теорії графів, різних технік роботи з даними в цифровій формі.

Навчання поділяють на типи:

– індуктивне (за прецедентами), базується на визначення певних емпіричних залежностей даних;

– дедуктивне, котре очікує формалізацію знань експертів та його переміщення на ПК як бази знань.

Для розробки системи ПФПК були використані такі методи ML та моделі ШНМ: Decision Tree; Random Forest; Naive Bayes; SGD; XGBoost; LightGBM; KNN; Logistic Regression; LinearRegression; SVM; ANN.

2.1 Decision Tree

Це засіб підтримки прийняття рішень, що застосовується в ML, аналізі даних і статистиці. На дереві є «листя» та «гілки». Ребра («гілки») містять ознаки. Від них залежить цільова функція. На «листках» наведені її значення, а інші вузли містять ознаки, котрими відрізняються випадки. Для класифікації нового випадку, потрібно опуститися по дереву до листка і видати певне значення.

В ІАД, дерева рішень можна застосувати як математичний апарат для проведення обчислень з метою опису, класифікації і узагальнення НД, котрі можна записати як (2.1):

$$(x,Y) = (x_1, x_2, x_3, \ldots x_k, Y), \quad (2.1)$$

де Y - залежна цільова змінна, котру треба проаналізувати, класифікувати та узагальнити; x - вектор x складається з вхідних змінних x_1,x_2,x_3 і т. д., які використовуються для виконання цього завдання.

Реалізація Decision Tree мовою python. Причиною вибору python та її перевагою є значна популярність у сфері ІАД. З цього факту випливає, що багато моделей прогнозування вже мають реалізацію в окремих бібліотеках, і вимагають лише налаштування. Далі на рис. 2.1 наведено приклад реалізації алгоритму дерева рішень.

```
clf3 = tree.DecisionTreeClassifier(min_samples_split = m_s,
                                   min_samples_leaf=m_l,
                                   ccp_alpha = c_a)
.fit(X_train_res, y_train_res)
y_pred3 = clf3.predict(X_test0)
```

Рисунок 2.1 – Реалізація алгоритму дерева рішень

Цей класифікатор реалізується в бібліотеці sklearn.tree. Для навчання потрібна наявність навчальної вибірки, виділеної під час ПОД.

Для навчання самого класифікатора використовуються додаткові параметри, їх можна опустити, тоді класифікатор використовує параметри за замовчуванням, але в програмі, що розробляється, ключовою частиною є ідея налаштування моделі - тобто встановлення параметрів класифікатора. Класифікатор skleam.tree.DecisionTreeQassifier має безліч параметрів для налаштування, для реалізації були використані такі:

– min_samples_split: int або float, default=2. Мінімальна кількість вибірок, що необхідні для розділення внутрішнього вузла;

– min_samples_leaf: int or float, default=1. Мінімальна кількість вибірок, яка потрібна для кінцевого вузла;

– ccp_alphanon_negative: float, default = 0.0. Параметр складності, що використовується для обрізки з мінімальними витратами та складністю.

Результати точності цієї моделі представлені у табл. 2.1.

Таблиця 2.1 – Результати точності моделі Decision Tree

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
Decision Tree	Mean accuracy	0.64	0.82

2.2 Random forest

Алгоритм застосовується для завдань класифікації, регресії та кластеризації [11]. В його основі – використання значного ансамблю обчислювальних дерев, власне кожне з яких дає низьку якість класифікації, проте за рахунок їх значної кількості отримується хороший результат.

На рис. 2.2 представлено схематичне порівняння архітектури моделей Випадкового лісу та Дерева рішень.

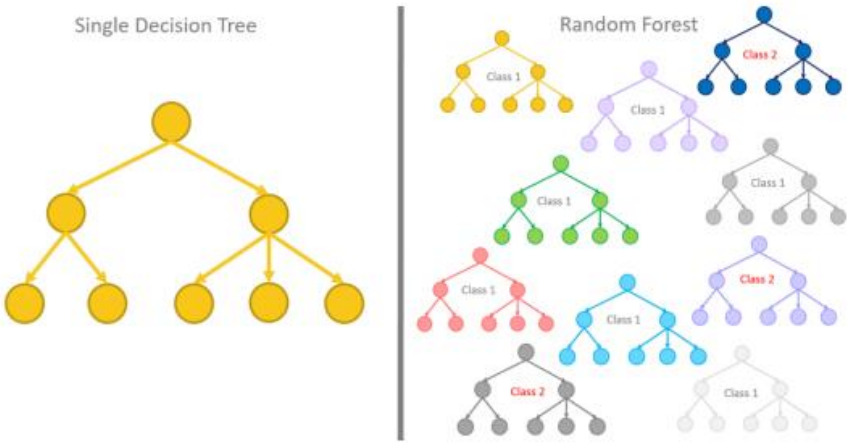

Рисунок 2.2 – Моделі Single Decision Tree та Random Forest

Алгоритм Random forest також реалізується на python із застосуванням

бібліотеки skleam.ensemble. Для навчання необхідна навчальна вибірка. Для навчання класифікатора використовуються додаткові параметри.

Для даної реалізації були використані наступні параметри класифікатора skleam.ensemble.RandomForestClassifier:

– max_depth: int, default = None. Максимальна глибина дерева;

– min_samples_split: int або float, default=1. Мінімальна кількість вибірок, необхідна для розділення внутрішнього вузла;

– min_samples_leaf: int or float, default=1. Мінімальна кількість вибірок, яка потрібна для кінцевого вузла;

Приклад інтерфейсу налаштування параметрів для користувача наведено на рис. 2.3.

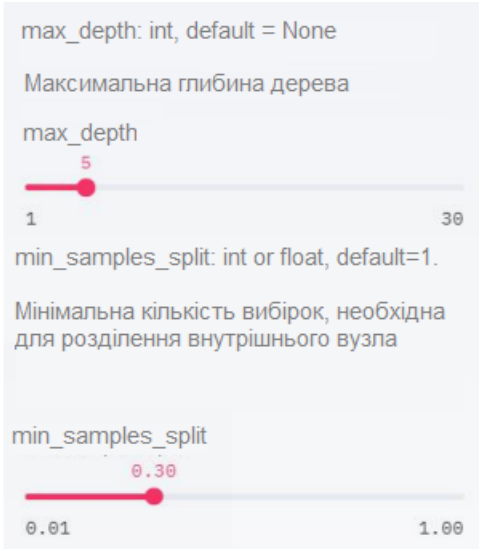

Рисунок 2.3 – Інтерфейс налаштування параметрів моделі

Результати моделі Random Forest представлені у табл. 2.2.

Таблиця 2.2 - Результати Random Forest

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
Random Forest	Mean accuracy	0.49	0.82

2.3 Naive Bayes

Простий імовірнісний класифікатор, котрий базується на застосуванні теореми Байєса зі строгими (наївними) припущеннями про незалежність.

Перевагою наївного класифікатора Байєса є мала кількість даних, необхідних для навчання, оцінки параметрів і класифікації.

Цей класифікатор базується на формулі Байєса:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}, \tag{2.2}$$

де $P(A)$ - апріорна ймовірність гіпотези A; $P(A \mid B)$ - ймовірність гіпотези A при настанні події (апостеріорна ймовірність); $P(B \mid A)$ - ймовірність настання події B при істинності гіпотези A; $P(B)$ – повна ймовірність настання події B.

Алгоритм Naive Bayes реалізується мовою python за допомогою бібліотеки skleam.naive_bayes. Для навчання необхідна навчальна вибірка. Реалізація алгоритму продемонстровано рис. 2.4.

```
clf2 = GaussianNB().fit(X_train, y_train)
y_pred2 = clf2.predict(X_test)
```

Рисунок 2.4 – Реалізація алгоритму Naive Bayes з використанням бібліотеки skleam.naive_bayes

Слід зазначити, що у інтерфейсі програми кожної моделі прикріплений код класифікатора - її реалізація. Для того щоб у користувача була можливість реалізувати цю модель самостійно.

Результати моделі Naive Bayes представлені у табл. 2.3.

Таблиця 2.3 – Результати моделі Naive Bayes

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
Naive Bayes	Mean accuracy	0.52	0.56

2.4 Stochastic Gradient Descent

Ітераційний метод для оптимізації цільової функції з відповідними властивостями гладкості (наприклад, диференційність або субдиференційність) [12]. Його можна розцінювати як стохастичну апроксимацію оптимізації методом градієнтного спуску, оскільки він замінює реальний градієнт (обчислений з повного НД його оцінкою (обчисленою з випадково обраної підмножини даних). Особливо в додатках обробки великих даних це скорочує обчислювальні ресурси і допомагає досягти більш швидкого обмін на нижчу швидкість збіжності.

І статистична оцінка, і ML розглядають завдання мінімізації цільової функції [10], що має форму суми:

$$Q(\omega) = \frac{1}{n}\sum_{i=1}^{n} Q_i(\omega), \tag{2.3}$$

де параметр ω, що мінімізує $Q(\omega)$, слід оцінити. Кожен член суми Q_i зазвичай асоціюється з спостереженням у НД.

При використанні для мінімізації наведеної вище функції, стандартний (або «пакетний») метод градієнтного спуску здійснює наступні ітерації:

$$\omega := \omega - \eta\nabla Q(\omega) = \omega - \eta\sum_{i=1}^{n}\nabla Q_i(\omega)/n, \tag{2.4}$$

де n – є розміром кроку чи іншими словами швидкістю навчання.

Алгоритм STG також реалізується мовою python за допомогою бібліотеки skleam.lmear_model. Для навчання необхідна навчальна вибірка, розділена на етапі п ередобробки на два набори: X_train (набір навчальних ознак), y_train (категоріальні ознаки набору навчальної вибірки).

Для реалізації використовуються додаткові параметри класифікатора sklearn.linear_model:

– alpha: float, default = 0.0001. Константа, що множить член регуляризації;

– Epsilon: float, default = 0.1. Епсилон для функції втрат;

– eta0: double, default = 0.0. Початкова швидкість навчання;

– n_iter_no_change: int, default=5. Кількість ітерацій без покращень, щоб дочекатися дострокової зупинки.

Код вихідної програми побудови моделі представлений рис. 2.5.

```
sgd_model=SGDClassifier(alpha=al,
                        epsilon=e,
                        eta0=eta,
                        n_iter_no_change=n)
sgd_model.fit(X_train,y_train)
```

Рисунок 2.5 – Реалізація моделі STG за допомогою бібліотеки sklearn.linear model

Приклад інтерфейсу виведення коду алгоритму моделі у програмі представлено на рис. 2.6.

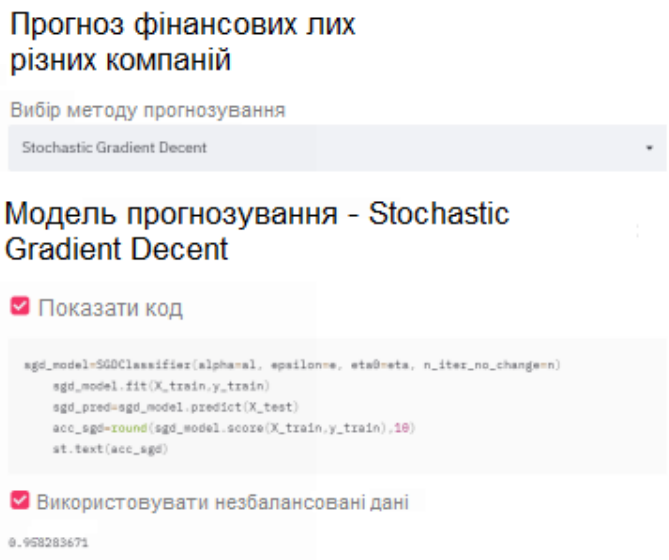

Рисунок 2.6 – Інтерфейс реалізованої програми ПФПК

Результати моделі STG представлені у табл.2.4.

Таблиця 2.4 – Результати моделі STG

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
STG	Mean accuracy	0.95	0.94

2.5 XGBoost

Бустинг (посилення) — композиційний метаалгоритм ML, головним чином застосовується для зменшення зміщення (похибки оцінки), і навіть дисперсії у навчанні з учителем. Також визначається як сімейство алгоритмів ML, що перетворюють слабкі навчальні алгоритми на сильні.

Бустинг грунтується на питанні, піднятому Кернсом і Веліантом (1988, 1989): «Чи може набір слабких навчальних алгоритмів створити сильний навчальний алгоритм?». Слабкий навчальний алгоритм визначається як класифікатор, який слабо корелює з правильною класифікацією (може помітити приклади краще, ніж випадкове вгадування). На відміну від слабкого

алгоритму, сильний навчальний алгоритм є класифікатором, що добре корелює з правильною класифікацією [13].

Переваги алгоритму:

– добре працює з різними даними: з невеликими даними, даними з підгрупами, великими даними та складними даними;

– більшість реального моделювання - це класифікація і регресія. Тобто метод XGBoost відмінний підхід при роботі зі структурованими НД.

Недоліки алгоритму:

– не так добре працює з розрідженими даними, а дуже розрізнені дані можуть створювати деякі проблеми;

– природа чорного ящика, саме прихована робота алгоритму, для докладної інформації роботи алгоритму доведеться реалізувати свої методи.

Реалізація алгоритму XGBoost показано рис. 2.7.

```
xgb_model=XGBClassifier()
xgb_model.fit(X_train,y_train)
```

Рисунок 2.7 – Реалізація алгоритму XGBoost за допомогою бібліотеки xgboost

Результати моделі XGBoost представлені у табл. 2.5.

Таблиця 2.5 – Результати моделі XGBoost

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
XGBoost	Mean accuracy	0.99	0.94

2.6 LightGBM

Алгоритм LightGBM має багато переваг XGBoost, включаючи розріджену

оптимізацію, паралельне навчання, множинні функції втрат, регуляризацію, упаковку та ранню зупинку. Основна відмінність між ними полягає у побудові дерев. LightGBM не ростить дерево поетапно - рядок за рядком - на відміну більшості інших реалізацій. Натомість він будує дерева, вкриті листям. Він обирає листок, який, на його думку, завдасть найбільшого зниження втрат.

Переваги алгоритму:

– швидкість навчання (порівняно зі схожим алгоритмом XGBoost) та більш висока ефективність: LightGBM використовує алгоритм на основі гістограми, тобто він поєднує безперервні значення функцій у дискретні комірки, які прискорюють процедуру навчання;

– нижче використання пам'яті: замінює безперервні значення дискретними комірками, що призводить до меншого використання пам'яті;

– краща точність, ніж будь-який інший алгоритм підвищення: він створює набагато складніші дерева, слідуючи підходу з поділом по листах, а не за рівневим підходом, який є основним фактором у досягненні вищої точності. Однак, іноді це може призвести до перенавчання, чого можна уникнути, встановивши параметр max_depth;

– сумісність із великими НД: він здатний однаково добре працювати з великими НД із значним скороченням часу навчання порівняно з XGBoost.

У реалізації алгоритму в програмі важливими параметрами, що налаштовуються, є:

– num_leaves: кількість листових вузлів, що використовуються. Велика кількість стулок підвищить точність, але також призведе до переоснащення;

– min_child_samples: мінімальна кількість вибірок (даних) для групування в аркуш. Параметр може значно допомогти при перенавченні: великі розміри вибірки на листок зменшать перенавчання;

– max_depth: явно контролює глибину дерева. Дрібніші дерева зменшують переоснащення, при цьому занадто великі дерева можуть вести до перенавчання моделі.

Результати моделі LightGBM представлені у табл. 2.6.

Таблиця 2.6 – Результати моделі LightGBM

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
LightGBM	AUC ROC	0.99	0.94

2.7 k-nearest Neighbors

Метричний алгоритм для автоматичної класифікації об'єктів чи регресії.

У разі використання методу для класифікації об'єкт присвоюється тому класу, який є найпоширенішим серед сусідів даного елемента, класи яких вже відомі. У разі використання методу для регресії об'єкту присвоюється середнє значення по k найближчим до нього об'єктам, значення яких вже відомі.

Алгоритм можна застосувати до вибірок з великою кількістю атрибутів (багатовимірних) [11]. І тому перед застосуванням потрібно визначити функцію відстані; Класичний варіант такої функції - евклідова метрика.

На рис. 2.8 наведено приклад роботи алгоритму.

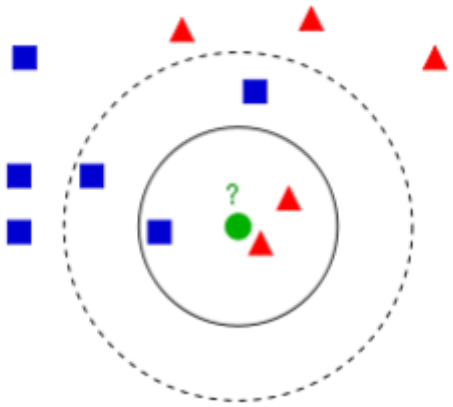

Рисунок 2.8 – Приклад роботи алгоритму KNN

При зваженому способі до уваги береться не тільки кількість певних класів, що потрапили в область, але і їх віддаленість від нового значення

$$Q_i = \sum_{i=1}^{n} \frac{1}{d(x,a_i)^2}, \tag{2.6}$$

де $d(x, a_i)$ - відстань від нового значення x до об'єкта a_i.. У якого класу вище значення близькості, той клас і надається новому об'єкту.

Алгоритм KNN також реалізується мовою python за допомогою бібліотеки sklearn.neighbors. Для навчання класифікатора використовуються додаткові параметри. Класифікатор sklearn.neighbors має безліч параметрів для налаштування, саме для даної реалізації були використані наступні параметри:

– n_neighbors: int, default=5. Кількість сусідів для використання.

– leaf_size: int, default = 30. Розмір листя, впливає швидкість побудови запиту, на обсяг пам'яті, необхідний зберігання дерева.

– p: int, default = 2. Степеневий параметр для метрики Мінковського

Налаштування цих параметрів, як і у всій програмі, відбувається динамічно, а параметри налаштовуються користувачем. На рис. 2.9 показано приклад роботи алгоритму з налаштованими параметрами, та висновок підсумкової оцінки моделі в інтерфейсі, реалізованим за допомогою streamlit.

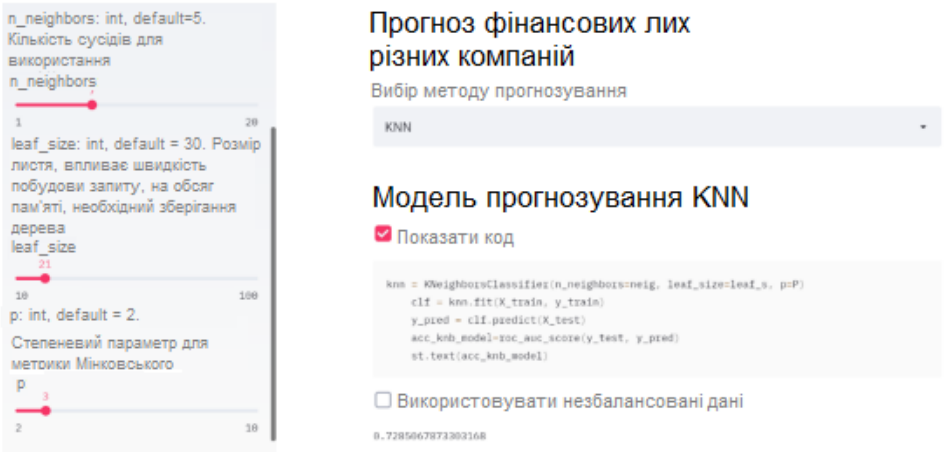

Рисунок 2.9 - Інтерфейс програми під час роботи з моделлю KNN

Результати моделі KNN представлені у табл 2.7.

Таблиця 2.7 – Результати моделі KNN

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
KNN	AUC ROC	0.50	0.76

2.8 Logistic Regression

Логіт-модель (англ. logit model) - це статистична модель, яка використовується для прогнозування ймовірності появи якоїсь події через її порівняння з логістичною кривою. Ця регресія видає у вигляді ймовірності бінарної події (1 чи 0).

Метод використовується для прогнозування імовірності появи якоїсь події за значенням множини ознак.

Для стислості функцію розподілу y при заданому x можна записати в такому вигляді:

$$P\{y \mid x\} = f(\theta^T x)^y (1 - f(\theta^T x))^{1-y}, y \in \{0,1\}. \quad (2.7)$$

Фактично, це розподіл Бернуллі з параметром $f(\theta^T,x)$. Застосовується дана модель на вирішення завдань класифікації - об'єкт x можна зарахувати до класу y = 1, якщо передбачена моделлю ймовірність $P\{y = 1|x\} > 0,5$, і класу y = 0 інакше. Отримуються при цьому лінійні класифікатори.

Реалізація Logistic Regression мовою python здійснювалася за допомогою готової бібліотеки skleam.lmear_model. Для навчання моделі необхідні як дані для аналізу, а й налаштування параметрів, слід зазначити, що за відсутності налаштування класифікатора - тобто вказівці параметрів класифікатора, модель

використовує базові налаштування на навчання. Далі представлені параметри моделі, які використовуються при реалізації алгоритму Logistic Regression:

– C: float, default = 1.0. Параметр регуляризації. інверсія сили регуляризації; має бути позитивним числом із плаваючою комою. Як і в машинах опорних векторів, менші значення вказують на сильнішу регуляризацію;

– max_iter: int, default = 100. Максимальна кількість ітерацій.

Використання функції skleam.lmear_model показано на рис. 2.10.

```
lr = LogisticRegression(C=c, max_iter = maxiter)
clf1 = lr.fit(X_train, y_train)
```

Рисунок 2.10 – Код алгоритму Logistic Regression із використанням бібліотеки skleam.linear_model

Результати моделі Logistic Regression представлені у табл. 2.8.

Таблиця 2.8 – Результати моделі Logistic Regression

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
Logistic Regression	Coefficient of determination	0.50	0.79

2.9 Linear Regression

Модель залежності змінної x від однієї чи декількох інших величин з лінійною функцією залежності[16].

Якщо розглядається залежність між однією вхідною та однією вихідною змінними, то має місце проста лінійна регресія. Для цього визначається рівняння регресії $y = ax + b$ і будується відповідна пряма, відома лінія

регресії, представлена рис. 2.11.

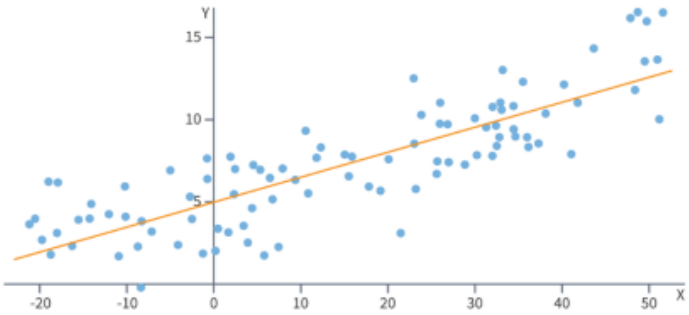

Рисунок 2.11 – Лінія регресії

Алгоритм лінійної регресії заснований на регресійній моделі представленою формулою:

$$y = f(x, b) + \varepsilon, \quad E(\varepsilon) = 0, \tag{2.8}$$

де b – параметри моделі; ε – випадкова помилка моделі.

При цьому регресійна модель є лінійною, якщо функція $f(x,b)$ набуває вигляду:

$$f(x, b) = b_0 + b_1 x_1 + b_2 x_2 + \cdots + b_k x_k, \tag{2.9}$$

де b_i - коефіцієнти регресії; x_i – регресори; k – кількість регресорів (факторів моделі).

Цей класифікатор реалізується в бібліотеці skleam.linear_model. Для навчання потрібна наявність навчальної вибірки, виділеної під час ПОД.

Для навчання самого класифікатора використовуються додаткові параметри, їх можна опустити, тоді класифікатор використовує параметри за замовчуванням, але в програмі, що розробляється, ключовою частиною є ідея налаштування моделі - тобто встановлення параметрів класифікатора.

Класифікатор sklearn.linear_model.Linear_Regression має безліч параметрів для налаштування, для даної реалізації були використані такі параметри:

– fit_intercept : bool, default = True. Чи слід розраховувати точку перетину цієї моделі. Якщо значення False, у розрахунках не береться перехоплення (тобто очікується, що дані будуть центровані).

– rank_: int. Ранг матриці X. Доступно, тільки якщо X є щільним.

Результати моделі Linear regression представлені у табл 2.9.

Таблиця 2.9 – Результати моделі Linear Regression

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
Linear Regression	Coefficien of determination	0.51	0.91

2.10 Support Vector Machines

Є набором подібних алгоритмів навчання з учителем, що використовуються для завдань класифікації та регресійного аналізу [12].

Обчислення (soft-margin) класифікатора SVM зводиться до мінімізації виразу:

$$\left[\frac{1}{n}\sum_{i=1}^{n}\max(0,1-\ y_i(w^T x_i - b))\right] + \ \lambda\|w\|^2, \qquad (2.10)$$

де w - набуває вигляду $w = \sum_{i=1}^{n} c_i y_i x_i;\ c_i = \max\left(0,1-\ y_i(w^T x_i - b)\right).$

Реалізацію алгоритму виконано за допомогою бібліотеки skleam.svm. Функція навчання класифікатора має численні налаштування, далі представлені параметри, що використовуються при параметризації моделі, тобто її

динамічному налаштуванні:

– C: float, default = 1.0. Параметр регуляризації. Сила регуляризації обернено пропорційна C. Повинна бути строго додатньою.

– degree: int, default=3. Ступінь поліноміальної функції ядра. Ігнорується рештою всіх ядр.

– cache_size: float, default = 200. Розмір кеша ядра (МБ).

Результати роботи алгоритму SVM представлені в табл. 2.10.

Таблиця 2.10 – Результати моделі SVM

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
Support Vector Machines	Coefficien of determination	0.50	0.60

2.11 Штучна нейронна мережа

Поняття ШНМ з’явилося для процесів, котрі присутні у мозку людини, і при спробі виконати їх моделювання [4]. ШНМ, використана у роботі, реалізована з допомогою бібліотеки Keras. У Keras необхідно зібрати шари для формування моделей. Стек шарів є найбільш поширеним видом моделі [5].

На рис. 2.12 показано схему нейрона.

Рисунок 2.12 – Схема нейрона

Стан нейрона можна визначити так:

$$S = \sum_{i=1}^{n} x_i w_i, \sum_{i=1}^{n} k^2, \quad (2.11)$$

де n – число входів нейрона; x_i - значення і-го входу нейрона; w_i - вага і-го синапсу.

Для реалізації ANN використовувалася бібліотека Keras. Keras – це API, розроблений для аналізу даних за допомогою ШНМ. Його перевагами є:

– містить у собі послідовні та прості API-інтерфейси;

– зводить до мінімуму кількість дій користувача, необхідних для поширених випадків використання, та надає чіткі та дієві повідомлення про помилки.

– містить велику документацію та посібники для розробників.

Keras – простий інструмент для побудови ШНМ. Це високорівневий фреймворк, заснований на бекендах tensorflow, theano.

Аналіз даних для ANN дещо відрізняється від відпрацювання методів ML:

– першим кроком, як і алгоритмах для ML, є попередня обробка і завантаження даних. Дані є ключем до роботи ANN, і, отже, необхідно обробити їх перед передачею до мережі. На цьому етапі також рекомендується візуалізувати дані, оскільки це допоможе краще їх зрозуміти;

– другим кроком є визначення моделі ШНМ, необхідно вказати кількість прихованих шарів у ній та їх розмір, розмір входу та виходу;

– третім кроком є визначення втрати та оптимізація моделі. Необхідно визначити функцію втрат відповідно до поставленого завдання. Також слід вказати оптимізатор для використання зі швидкістю навчання та іншими гіперпараметрами оптимізатора;

– четвертий та останній крок - це етап навчання ШНМ, в якому потрібно визначити кількість епох, для яких необхідно навчити мережу.

Після навчання моделі можна протестувати її на тестових даних, щоб перевірити, чи є переоснащення.

Для початку відбувається налаштування ANN, показане на рис. 2.13.

```
#Dependencies
import keras
from keras.models import Sequential
from keras.layers import Dense


# Neural network
model = Sequential()
model.add(Dense(16, input_dim=20, activation='relu'))
model.add(Dense(12, activation='relu'))
model.add(Dense(4, activation='softmax'))
```

Рисунок 2.13 – Налаштування параметрів ANN

Sequential показує keras, що модель створюється послідовно, і вихідні дані кожного шару, що додається, є вхідними даними для наступного шару, який вказує розробник.

model.add використовується для додавання шару до ANN. Потрібно вказати як аргумент, який тип шару необхідний роботи з даними. Dense використовується для вказівки пов'язаного шару. Аргументами Dense є вихідний вимір, що дорівнює 16 у першому випадку, вхідний вимір, що дорівнює 20 для вхідного виміру, і функція активації, яка буде використовуватися в даному випадку.

Другий рівень аналогічний, не потрібно вказувати вхідний вимір, оскільки дана модель визначається як послідовна, тому keras автоматично вважатиме вхідний вимір таким самим, як вихід останнього шару, тобто 16. У третьому шарі (вихідний шар) розмірність виводу - 4 (кількість класів). Вихідний шар виконує різні функції активації, а разі мультикласової класифікації це softmax[20].

Далі на рис. 2.14 представлено визначення функції оптимізації.

```
model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
```

Рисунок 2.14 – Приклад оптимізації моделі keras

У цьому прикладі loss - це втрата крос-ентропії. Category_crossentropy показує, що модель має кілька класів. Оптимізатор – Адам. Метрики використовують визначення способу оцінки продуктивності ШНМ.

Крок навчання простий у keras і наведений на рис. 2.15.

```
history = model.fit(X_train, y_train, epochs=100, batch_size=64)
```

Рисунок 2.15 – Навчання ШНМ у інструменті keras.

Тут потрібно вказати вхідні дані: X_train, y_train, кількість епох (ітерацій) та розмір пакету. Повертає історію навчання моделі. Історія складається з точності моделі та втрат після кожної епохи. Слід зазначити, що такі параметри, як epochs, batch_size, налаштовуються в програмі динамічно за допомогою інтерфейсу користувачем:

– epochs: int. Епоха – це одна ітерація у процесі навчання. Так як комп'ютера складно впоратися з усім навчанням відразу, його ділять на епохи;

– batch_size: int. Загальна кількість тренувальних об'єктів, які представлені в одному навчанні.

Навчання моделі зберігається в змінній, це необхідно для подальшого виведення даних про результати роботи моделі. Саме - інформація про роботу кожної епохи, і результати втрат і точності кожному етапі навчання, ці дані представлені на рис. 2.16.

Рисунок 2.16 – Результат роботи алгоритму ШНМ кожної епохи

Після досягнення значення accuracy понад 0.9 модель є навченою, і її можна випробовувати на тренувальній вибірці. Після випробування моделі можна візуалізувати і похибки при навчанні та перевірці. Для цього використовується бібліотека matplotlib.pyplot. Графіки, отримані внаслідок навчання представлені рис. 2.17 і 2.18 відповідно.

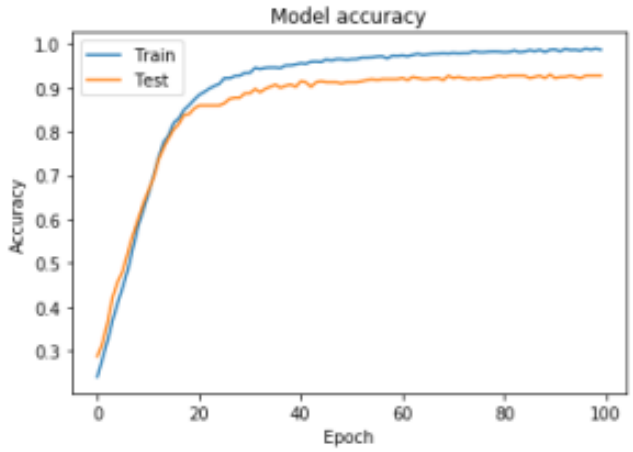

Рисунок 2.17 – Графік точності нейромережі

Результати роботи моделі нейронної мережі представлені у табл. 2.11.

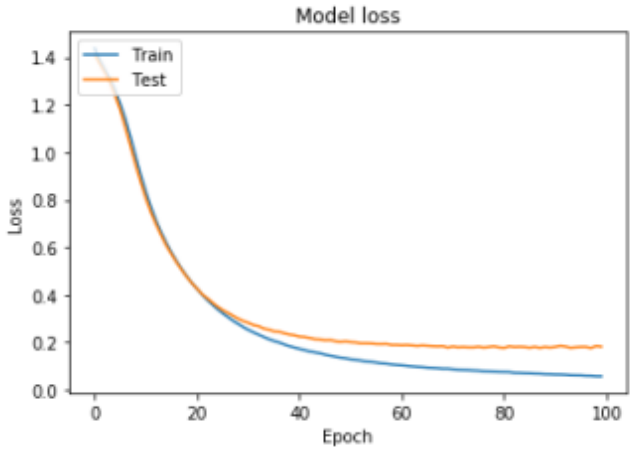

Рисунок 2.18 – Графік втрат нейромережі

Таблиця 2.11 – Результати моделі ANN

Модель	Оцінка	Значення оцінки на збалансованих даних
ANN	AUC ROC	0.96

Варто відзначити той факт, що в цій таблиці не вказано значення результатів роботи алгоритму нейронної мережі на незбалансованих даних, оскільки клас Keras при навчанні має параметр автоматичного балансування даних.

2.12 Висновки до другого розділу

У цьому розділі описані та проаналізовані існуючі методи ML та моделі ШНМ. Для кожного наведено математичний апарат, переваги використання їх.

Описано особливості застосування методів та моделей на мові pytnon, приведені параметри налаштування, значення оцінок на збалансованих та незбалансованих даних.

3 ПРАКТИЧНА ЧАСТИНА

3.1 Архітектура системи

Для виконання завдання створення системи ПФПК використовувався інструмент Streamlit. Це бібліотека мовою Python, що реалізує клієнт-серверний додаток.

Streamlit зручний інструмент, оскільки його використання на рівні розробки прирівнюється до використання стандартних бібліотек на python. Цей інструмент часто використовується в аналізі даних завдяки його перевагам:

– бібліотека дає можливість локально та на серверному рівні розгорнути дашборди, що дозволяють відобразити результати аналізу, реалізованих мовою python;

– має можливість розгортати кешовані моделі прогнозування, тим самим збільшуючи продуктивність дашборду, не використовуючи зайвих ресурсів для перенавчання моделей. За рахунок цього ж скорочується час відгуку до серверної програми;

– використання бібліотеки доступне в, оскільки цей інструмент має вичерпну документацію. Також за рахунок її популярності існують форуми, де проілюстровані приклади розгортання та створення клієнт-серверної програми, а також є готові відповіді на популярні питання під час роботи зі Streamlit;

– низький поріг входження. За допомогою кількох базових команд, представлених на рис. 3.1, програміст чи аналітик зможе розробити односторінкову програму.

```
#ВСТАНОВЛЕННЯ
pip install streamlit
#використання бібліотеки
import streamlit as st
#базові команди і функції
#параметри меню
st.sidebar.header('Назва панелі навігації ')
#Заголовки
st.header('Заголовок')
st.subheader('Підзаголовок')
#Вивід тексту, даних dataframe чи картинок
st.write(' Дані для аналізу ')
st.write(dataframe)
st.write('Приклад тексту')
st.text('Приклад тексту')
st.image('.../saved_figure.png')
```

Рисунок 3.1 – Основні команди для роботи зі Streamlit

Слід зазначити, що в програмі, трак реалізується, кешування не використовувалося, так як основною ідеєю додатку було створення інтерактивного дашборду. Інтерактивність дашборду здійснюється за рахунок параметризації моделей прогнозування, тобто можливість користувачем налаштовувати параметри моделей, тим самим отримувати моделі, що динамічно навчаються. Архітектура програми представлена на рис. 3.2.

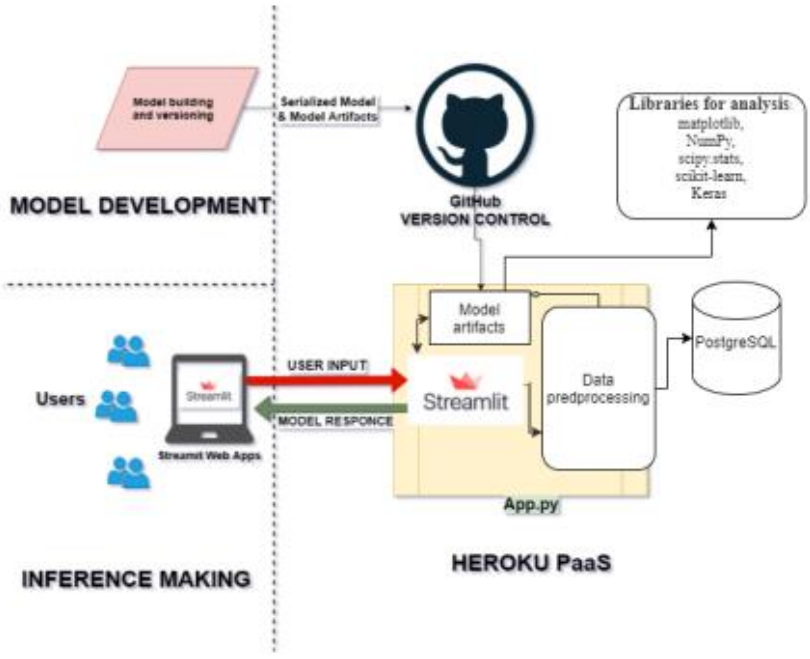

Рисунок 3.2 – Архітектура ПФПК

На рисунку App.py - система, заснована на файлах python (Model artifacts). У цих файлах реалізовані моделі прогнозування, які раніше описані в роботі, за допомогою сторонніх бібліотек для аналізу даних.

Дані для дослідження зберігаються в базі даних PostgreSql і викликаються за допомогою SQL - запитів з блоку App.py.

Всі файли та конфігурації програми завантажені в інструмент контролю версій GitHub, за допомогою якого відбувається розгортання програми на базі Heroku Paas.

Користувачі отримують доступ до моделей засобами передачі та отримання даних інструментом streamlit.

За допомогою програми користувач отримує доступ до моделей та має можливість змінювати параметри моделей. Інтерфейс програми представлений на рис. 3.3.

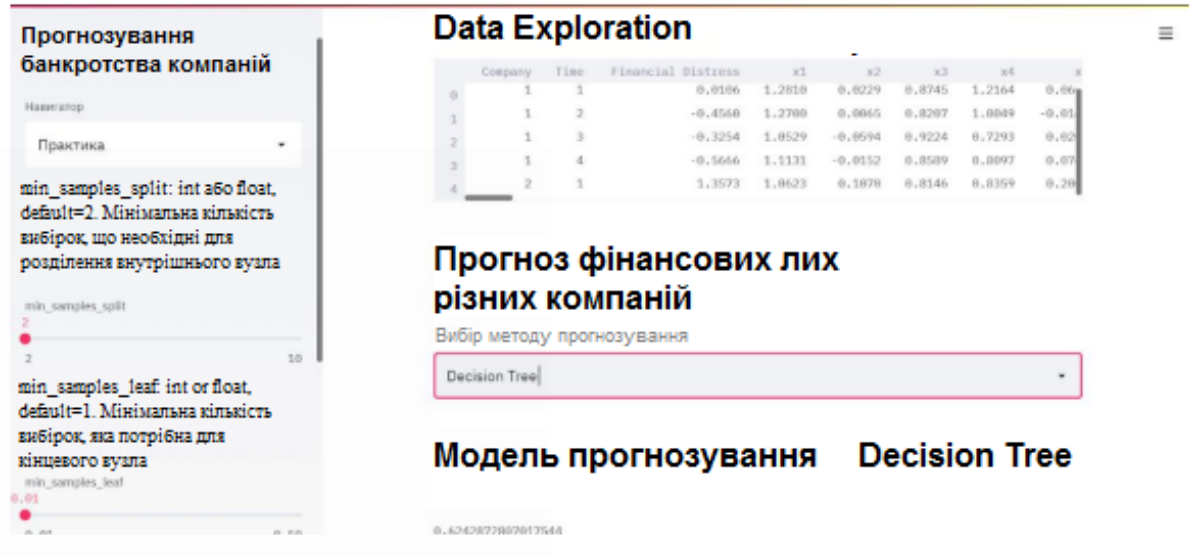

Рисунок 3.3 – Інтерфейс системи ПФПК

У верхній частині лівої панелі сторінки представлена навігація за програмою. Існує кілька розділів:

– Теорія - у цьому розділі представлена теоретична інформація щодо моделей прогнозування, що використовуються в програмі;

– Передобробка даних - у розділі наведено варіанти ПОД для моделей ANN та ML;

– Про дані - в даному розділі описані використовувані дані: інформація про стовпці, кількість вибірки та ін. Тут також з різними графіками розподілу щільності, матрицями кореляції, показані залежності між ознаками;

– Практика – цей розділ є основним, тут представлений вибір моделей прогнозування з інформацією про отримані дані, а також панелі зміни параметрів моделі.

Повний сценарій використання програми наведений на рис. 2.4. На схемі представлені в повному обсязі моделі, неповний список представлений задля економії місця та демонстрації самої можливості вибору способу прогнозування.

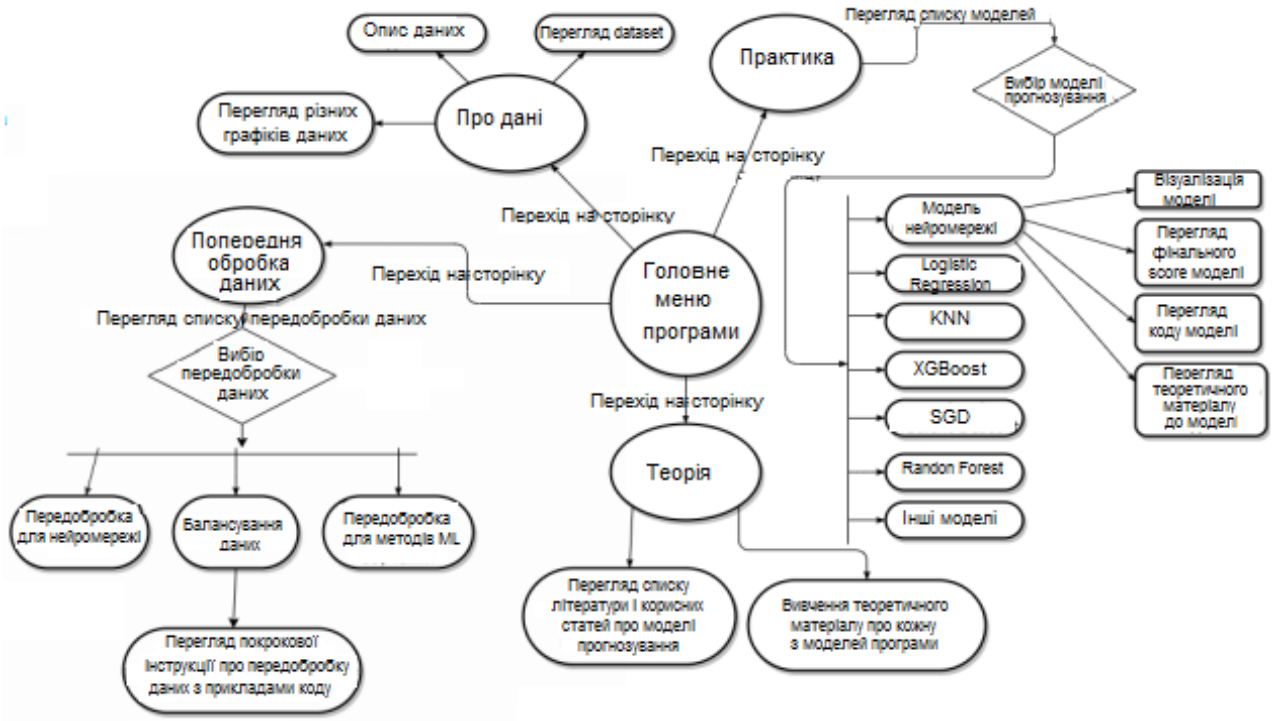

Рисунок 3.4 – Діаграма варіантів використання програми прогнозування

Такі функції, як візуалізація моделей, перегляд підсумкового score, перегляд коду моделі, перегляд теоретичного матеріалу, доступні для кожної з моделей прогнозування. Кожна з цих функцій окремих моделей може

відрізнятися у вигляді специфіки кожної з моделі, наприклад, до розрахунку точності регресійних моделей використовується оцінка “Coefficient of determination”, а частини методів - “ROC AUC”. А візуалізація моделей, а також код моделей та теоретична частина відрізняється для кожного з методів.

3.2 Аналіз отриманих моделей

У процесі дослідження та навчання моделей були отримані результати, подані у таблиці 3.1.

Таблиця 3.1 – Зведена таблиця результатів роботи моделей

Модель	Оцінка	Значення оцінки на незбалансованих даних	Значення оцінки на збалансованих даних
XGBoost	Mean accuracy	0.99	0.95
LightGBM	AUC ROC	0.99	0.94
ANN	AUC ROC	-	0.96
SGD	Mean accuracy	0.95	0.94
Decision Tree	Mean accuracy	0.64	0.82
Naive Bayes	Mean accuracy	0.52	0.56
SVM	Coefficient of determination	0.50	0.60
KNN	AUC ROC	0.50	0.76
Linear Regression	Coefficient of determination	0.51	0.91
Logistic Regression	Coefficient of determination	0.50	0.79
Random Forest	Mean accuracy	0.49	0.82

Моделі, представлені в таблиці, відсортовані в ній тільки для наочності, так як для оцінки точності використовувалися різні показники з огляду на належність моделей до різних класів. Для оцінки точності моделей використовувані дані були поділені на дві вибірки: навчальна та тестова у співвідношенні 70% та 30% відповідно.

Що стосується конкретних оцінок, то здебільшого це вбудовані функції у відповідних бібліотеках, що використовуються при побудові моделей [14].

Mean accuracy - співвідношення правильного числа класифікованих об'єктів до загального розміру навчальної вибірки

$$Accuracy = \frac{P}{N}, \tag{3.1}$$

де P – кількість правильно класифікованих об'єктів, N – об’єм вибірки.

Coefficent of determination є часткою дисперсії залежної змінної, котра пояснюється аналізованою моделлю залежності [15]. Реальний коефіцієнт детермінації моделі залежності випадкової величини від факторів *x* обчислюється так:

$$R^2 = 1 - \frac{D[y|x]}{D[y]} = 1 - \frac{\sigma^2}{\sigma_y^2}, \tag{3.2}$$

де D[y] = σ^2_y - дисперсія випадкової величини y, D [y|x] = σ^2 - умовна за факторами x дисперсія залежної змінної (дисперсія помилки моделі).

AUC ROC – площа під кривою помилок. Графік, котрий дає змогу провести оцінку якості бінарної класифікації, показує взаємозв’язок між часткою об'єктів від загального числа носіїв ознаки, вірно класифікованих як ознаки, так і часток об'єктів від загального числа тих, які не несуть ознаки, але

хибно класифікованих як ознаки (величина 1-FPR називається специфічністю алгоритму класифікації) при варіюванні порога вирішального правила.

Найвищий результат на незбалансованих даних показали алгоритми бустингу: XGBoost, LightGBM. Такий високий результат є помилкою навчання моделей, тобто моделі можуть перенавчитися за рахунок дисбалансу класу, як це було на незбалансованих даних, і за рахунок вказівки параметра занадто глибокого навчання. Частково причина такої похибки навчання криється у дуже великому розбалансуванні класів. Також при навчанні слід приділити більше уваги параметру max_depth, який явно контролює глибину дерева, і через який можуть виникнути проблеми з переоснащенням дерева бустингу.

На збалансованих даних ці два способи показали схожі результати, як і алгоритми ANN та STG.

Якщо порівнювати два методи LightGBM і XGBoost, перший може обробляти категоріальні функції, вводячи імена функцій. Він не конвертується в одноразове кодування і є набагато швидше, ніж таке кодування [16]. LightGBM використовує спеціальний алгоритм, щоб знайти значення поділу категоріальних ознак. З практичного боку деяким мінусом LightGBM є той факт, що перед навчанням необхідно перетворити свої категоріальні функції на тип int. Якщо говорити про переваги XGBoost, слід відзначити вбудовану в алгоритм обробку пропущених значень. Найнижчі результати на незбалансованих даних показали алгоритми: Naive Bayes, Support Vector Machines, KNN, Logistic Regression, Linear Regression, Random Forest.

Такий результат є цілком очікуваним на такій незбалансованій вибірці, тобто відношення кількості рядків одного класу до іншого - 95% вибірки проти 5%, при такому співвідношенні вибірки та використанні таких метрик як mean accuracy, результуючі підсумки навчання моделей стануть якоюсь мірою про помилковими .

Слід зазначити, що лінійна регресія все ж таки краще спрацювала, ніж логістична регресія - 0.50 проти 0.51 на незбалансованих даних. Загалом це

невелика різниця на незбалансованих даних, але на збалансованих даних різниця в результатах прогнозу вже відчутна: 0.91 у лінійної регресії проти 0.79 у логістичної регресії, варто виділити той факт, що зазвичай логістична регресія працює краще, ніж лінійна для завдань класифікації з наступних причин [14]:

– прогнозоване значення є безперервним, а не імовірнісним;

– чутливі до дисбалансу даних під час використання лінійної регресії для класифікації (дані вибірки є незбалансованими).

Лінійна регресія могла показати кращі результати, ніж алгоритм логістичної регресії за рахунок таких факторів:

– вона працює виключно добре для лінійних даних;

– досить добре справляється з перенавчанням, використовуючи методи зменшення розмірності, регуляризацію та перехресну перевірку.

У задачі бінарної класифікації нас цікавить можливість настання результату. Імовірність знаходиться в діапазоні від 0 до 1, де ймовірність того, що щось, напевно, відбудеться, дорівнює 1, а 0 - щось малоймовірне. Але в лінійній регресії ми пророкуємо абсолютне число, яке може виходити за межі 0 та 1 [12].

Random Forest показав значно менші результати, ніж Decision Tree. Можливою причиною є те, що існує можливість перенавчання. Застосування декількох дерев у випадковому лісі зменшує ризик переоснащення.

Загалом дерево рішень просте для розуміння, його легше пояснювати та візуалізувати. Не потребує великої підготовки даних, здатне опрацьовувати як числові, і категоріальні дані. Проте, шум даних може призвести до перенавчання. Крім того, модель також може стати нестабільною через незначні зміни.

Робота алгоритму Naive Bayes вважається швидшою, ніж суміжних алгоритмів ML, але різниця в роботі цього алгоритму на незбалансованих та збалансованих даних не суттєва – 0.52 проти 0.56. Тобто алгоритм однаково погано спрацював на розбалансованих та врівноважених даних, можливо такі

результати роботи моделі випливають із специфіки даних та недоліків алгоритму, описаних нижче:

– наївний Байєс припускає, що всі провісники (або характеристики) незалежні, що рідко зустрічається в реальному житті. Це обмежує застосування цього алгоритму в реальних випадках;

– алгоритм стикається з «проблемою нульової частоти», коли він надає нульову ймовірність категоріальної змінної, категорія якої в наборі тестових даних не була доступна в наборі навчальних даних. Для обходу цієї проблеми використовується згладжування.

Подібні результати з алгоритмом Наївного Байєса видав алгоритм SVM: 0.50 та 0.60 оцінки точності на незбалансованих та збалансованих даних відповідно. Такі результати також можуть бути наслідком таких недоліків алгоритму:

– не підходить для великих вибірок;

– погано відпрацьовує на наборах даних з шумом (сміття даних за рахунок випадкових викидів у даних).

Також на незбалансованих даних спрацював алгоритм найближчих сусідів: 0.50, але при цьому алгоритм KNN відпрацював краще ніж SVM на балансованих даних. Слід виділити недоліки алгоритму KNN:

– не працює з великим НД: там вартість обчислення відстані між новою точкою і кожною точкою величезна, що знижує продуктивність алгоритму;

– потрібно масштабувати функції: необхідно виконати масштабування функцій (стандартизацію та нормалізацію) перед застосуванням алгоритму KNN до будь-якого НД. Якщо цього не зробити, то KNN може давати невірні прогнози;

– як і алгоритм SVM, KNN чутливий до зашумлених даних, відсутніх значень та викидів.

Але він також має переваги перед деякими алгоритмами ML - у нього немає періоду навчання: KNN називається Lazy Learner (навчання на основі

екземплярів). Нічого не впізнає під час навчання. Він не виводить жодної відмінної функції з навчальних даних. Інакше кажучи, в нього немає тренувального періоду. Він зберігає набір навчальних даних та навчається на ньому лише під час складання прогнозів у реальному часі. Це надає алгоритму KNN значно більшу швидкість, аніж іншим алгоритми, котрі потребують навчання, наприклад, SVM або лінійна регресія. Також цей алгоритм легкий у реалізації. Для реалізації KNN потрібні лише два параметри: величина K та функція відстані (наприклад, евклідова чи манхеттенська тощо).

Якщо ж говорити про лідерів у результатах на збалансованих даних, то це алгоритми SGD, ANN, тому слід виділити їх особливості.

Алгоритм SGD швидко відпрацьовує для великих НД, оскільки викликає частіше оновлення параметрів. Через часті оновлення кроки, що робляться в напрямку мінімумів функції втрат, мають коливання, які можуть допомогти вийти з локальних мінімумів функції втрат (у випадку, якщо обчислене положення виявляється локальним мінімумом).

У випадку ж з алгоритмом ANN, ці алгоритми в якійсь мірі універсальні, так як використовуються для завдань, що мають цільову функцію, вихідні дані можуть бути дискретними, дійсними або вектором з декількох дійсних або дискретних атрибутів. Також алгоритм досить стійкий до шуму у навчальних даних. Приклади навчання можуть містити помилки, що не впливають на кінцевий результат.

Недоліками алгоритму є той факт, що ШНМ за своєю структурою потребують процесорів із паралельною обчислювальною потужністю. З цієї причини реалізація обладнання є залежною. Але з іншого боку для цієї проблеми є готові рішення, готові інструменти аналізу: tenserflow або keras.

Важливим фактом при побудові ШНМ є гарантія правильної мережевої структури: немає конкретного правила визначення її структури. Відповідна структура мережі досягається за рахунок досвіду, спроб та помилок. З цього випливає специфіка при роботі з ШНМ, для цього необхідні спеціальні навички

та вміння: вміння вибудовувати та підбирати шари для неї, вміння переконатися, що мережа навчена і тільки тоді приступати до тестування даних, не варто також забувати про окреме налаштування деяких параметрів. І в цілому з цієї причини алгоритм ANN так сильно виділяється серед моделей ML, тому що для цих алгоритмів існують готові бібліотеки, де моделі описуються кілька рядків.

Загалом можна дійти висновку, що багато моделей показали результати гірше на незбалансованих даних. На збалансованих даних 9 з 11 моделей показали результати точності прогнозування вище 0.70.

3.3 Висновки до третього розділу

В цьому розділі приведено і докладно описано загальну архітектуру програмного засобу для ПФПК. Показано інтерфейс системи ПФПК. Також наведено і описано діаграму варіантів використання.

Виконано експериментальний аналіз усіх моделей, описаних в розділі 2. Проведені експерименти доводять, що для більшості моделей отримані кращі результати для збалансованих даних. Лідерами є алгоритми SGD, ANN.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Закордонний досвід організації охорони праці в ІТ-компаніях

Стан справ з охороною праці у світі стає все більш актуальною проблемою як для профспілок, так і для міждержавних структур, насамперед Міжнародної організації праці (МОП). МОП розглядає цю тему як частину своєї Програми гідної праці. Підвищена увага до проблем безпеки праці пояснюється в першу чергу тим, що з кожним роком, незважаючи на заходи, що вживаються, у різних країнах зростає рівень виробничого травматизму, у тому числі зі смертельними наслідками, і кількість профзахворювань.

На перший погляд, робота за комп'ютером при дослідженні моделей та побудові системи ПФПК здається безпечною, але саме легковажність до неї може призвести до певних проблем у здоров'ї людини. Професія програміста та інших фахівців ІТ-технологій пов'язана з колосальним розумовим напруженням. Розробники – настільки захоплені люди, що навіть відволікаючись від роботи над проектом, продовжують думати про роботу. Нерідко відпочинком вони вважають паралельну заміну основної діяльності, наприклад, читання профільної літератури, верстку сайтів, вивчення нових мов програмування. Однак мозок не може до безкінечності приймати виключно корисну інформацію, яку розробник прагне направляти в русло особистісного та професійного зростання [21].

Зарубіжний досвід охорони праці при використанні новітніх інформаційних технологій та сучасного комп’ютерного обладнання передбачає з метою попередження наслідків монотонної праці, підвищення рівня рухової активності і покращення розумової працездатності фахівців ІТ-індустрії під час технологічних перерв участь у спеціальних облаштованих приміщеннях необхідним спортивним інвентарем та різними тренажерами відповідних фізичних вправ, індивідуальних тренінгових завдань відповідно до віку, статі та

категорії зорової роботи. Такий підхід дозволяє зняти надлишкове психофізіологічне перевантаження, підвищити працездатність центральної нервової системи, попередити перевтому зорового аналізатора. Показана ефективність проведення різноманітних за своєю спрямованістю вправ робітників цієї галузі (приблизно на 5-30%) [21].

Зараз багато IT-компаній обладнують свої офіси кімнатами відпочинку та лаунж-зонами, які забезпечують психофізіологічне розвантаження працівників. Адже окремим робочим столом з ноутбуком вже давно нікого не здивуєш. Тому, бажаючи підвищити продуктивність працівників, міжнародні компанії змагаються, перетворюючи нудні одноманітні офіси в креативні простори, де нові ідеї народжуються без титанічних зусиль. Наприклад, винахідники компанії Inventionland трудяться в казкових декораціях. Тут і гігантський гоночний трек, і пряниковий будиночок, і дуже реалістичний піратський корабель, на палубі якого розташувалися комп'ютерні столи, ніжками яких служать винні бочки. Робочі місця співробітників компанії Google в Цюріху нагадують гігантські вулики, а офіс шведського інтернет-провайдера Bahnhof розташувався в бомбосховищі часів холодної війни і походить на підземний притулок землян після глобальної катастрофи. А щоб співробітників не тягнуло додому, роботодавці створюють і можливість релаксувати, не відходячи від робочого місця, обладнавши басейни, ігрові кімнати та спортзали [21].

Корпорація Google піклується і про санітарно-гігієнічні умови праці та використовує систему очищення повітря, яка видаляє з атмосфери всі токсини та тяжкі домішки [22]. Також незвичайними на території офісу є спеціальні ізолюючі світло й звук капсули, де втомлений співробітник може відпочити, так само як і в спеціалізованих кімнатах у зоні відпочинку з приглушеним світлом та заспокійливою музикою, масажних кабінетах, ігрових кімнатах. Культивації активного відпочинку та спорту в Google відводиться особлива увага, оскільки одні з найпоширеніших хвороб програмістів пов’язані зі спиною. Саме тому в головному офісі міститься спортзал, в якому можуть одночасно займатися

велика частка співробітників, до того ж на території є плавальний басейн з черговим рятувальником, тому, фактично, людина може працювати прямо звідти (пам'ятаючи про дотримання правил техніки безпеки) [23]. Для переміщення по великій території Гуглмістечка «гуглівці» (так себе називають співробітники цієї компанії) користуються самокатами, які мають моторчики. Звичайно, усі ці зручності безкоштовні.

Намагається не відставати від Google і корпорація Facebook. Так, дана фірма пропонує своїм співробітникам чудові умови праці: якісне різноманітне харчування, зручне апаратне забезпечення, важливою складовою якого є широкі монітори ПКз високою роздільною здатністю [24]. Особливістю Facebook є те, що співробітники не мають ані своїх кабінетів, ані офісних комірок – усі сидять разом, оскільки це стимулює комунікацію та обмін ідеями, підвищує продуктивність роботи. Загалом у компанії царює атмосфера позитиву. Усі співробітники привітні та посміхаються. На стінах можна побачити карикатури на топменеджерів, а робітники одягають футболки зі смішними написами [24]. 16 Співробітники тут, на відміну від Google, пересуваються по офісу не на самокатах, а на скейтах. У Facebook також наявна велика кількість тематичних та ігрових зон, де кожен може розслабитися та відпочити, і це все також абсолютно безкоштовно.

Поява та впровадження нових інформаційно-комунікаційних технологій зумовлює необхідність подальшого вдосконалення охорони праці фахівців ІТ-індустрії. З метою належного правового забезпечення необхідно розширити ти доповнити перелік основних професій комп'ютерної галузі у національному класифікаторі ДК-003-2010 [25], а також підготувати відповідний випуск у кваліфікаційному довіднику посад фахівців ІТ-індустрії, що сприятиме вирішенню питань їх соціального захисту, пенсійного забезпечення, атестації робочих місць основних професій за умовами праці на предмет подальших певних видів пільг та компенсацій за важкі шкідливі і небезпечні умови праці. Важливим напрямом стосовно визначення професійної придатності фахівців з

інформаційних технологій є проведення психофізіологічної експертизи відповідно до 5 статті [26].

ІТ-фахівці, як і будь-які інші працівники, повинні проходити навчання і перевірку знань з охорони праці або в навчальному центрі, або в самій організації. Якщо в ній є комісія з перевірки знань з охорони праці, атестованих в спеціалізованому навчальному центрі. Навчання охорони праці в організації проводять по самостійно розробленими програмами. Їх складають, спираючись на типові програми, а також з огляду на особливості галузі, в якій працює організація.

Робота з комп'ютерами нового покоління характеризується певним психофізіологічними перенавантаженнями, втомою зорового аналізатора, гіпокінезією, відсутність диференційованих норм праці при роботі з новою комп'ютерною технікою в залежності від віку, статі, категорії зорової роботи, режимів праці і відпочинку (протягом робочого дня, тижня, щорічного режиму відпусток). Все це потребує розробки нових нормативно-правових актів з регламентації праці та відпочинку фахівців ІТ-індустрії і стандартів підприємств, центрів комп'ютерної техніки, центрів інформаційних технологій, сучасних комп'ютерних класів. Особлива роль з точки зору збереження та відновлення здоров'я працюючих в комп'ютерні галузі належить попереднім та періодичним наглядам з подальшої психофізіологічної експертизи і встановленням професійної придатності при роботі з комп'ютерами нового покоління, який супроводжується виникненням певних факторів професійного ризику електротравматизму при їх ремонті та обслуговуванні. В цьому зв'язку необхідне запровадження експертизи на предмет безпечної експлуатації ПЕОМ, тобто офіційне підтвердження фактичних параметрів електробезпеки, їх відповідності вимогам нормативної документації фахівців, які проводять таку експертизу повинні пройти навчання і перевірку знань відповідно до вимог [427. За результатами експертизи повинні прийматися рішення про відповідність ПЕОМ нормам безпеки, терміни чергової експертизи, оформлюються протоколи

вимірювань і випробувань, проведені у разі потреби розрахунки та експертний висновок. Для підвищення розумової працездатності то зорової роботи повинна здійснюватися ергономічна оптимізація в рамках системи «оператор-термінал», яка сприятиме результативній фізичній та інтелектуальний працездатності і відновленню психосоматичного здоров'я фахівців ІТ-індустрії.

Заслуговує на увагу зарубіжний досвід створення у приміщеннях та в зоні їх розміщення на територіях підприємств спеціальних візуальних комфортних умов та забезпечення вимог виробничої естетики, дотримання норм рівнів виробничого шуму та акустичної тиші за межами офісу. Також дуже важливим є використання в офісних приміщеннях та кабінетах психофізіологічного розвантаження функціональної музики, яка сприяє попередженню перевтоми і підтриманню необхідного рівня розумової працездатності фахівців комп'ютерної галузі. В цьому напрямі заслуговує на увагу створення при великих центрах інформаційних технологій кімнат (кабінетів) психофізіологічного розвантаження працівників галузі (на 5 місць) [21].

Всі наведені заходи щодо вдосконалення охорони праці фахівців ІТ-індустрії повинні контролюватися службою охорони праці та комісією з охорони праці підприємства. Особливе значення у соціальному захисті цієї категорії працівників належить прийняття комплексного договору, який може забезпечити фахівців додатковими пільгами та компенсаціями.

Таким чином, використання новітніх технологій вимагає від фахівців ІТ-індустрії додержання певних правил та вимог з точки зору безпеки праці, її нормування з урахуванням віку працюючих та загального інформаційного навантаження, розробки та впровадження індивідуальних, щотижневих та щорічних режимів праці та відпочинку, які сприятимуть профілактиці перевтомлення і підвищенню розумової працездатності працюючих. Особливу роль в цьому напрямі повинні відігравати ергономічні заходи стосовно створення робочих місць, оптимізації взаємодії людини в рамках системи «оператор-термінал». Всі ці вимоги повинні бути втілені у відповідних

нормативно-правових актах (стандартах підприємств), що регламентують різноманітні питання охорони та психології безпеки праці фахівців ІТ-індустрії.

4.2 Оцінка дії електромагнітного імпульсу на елементи комп'ютерної системи

У воєнний час при застосуванні ядерної зброї проти України на електронно-обчислювальне обладнання в першу чергу буде впливати електромагнітний імпульс **(**EMI) ядерного вибуху у вигляді короткого імпульсу, який вражає головним чином електричну та електронну апаратуру [28].

EMI виникають в основному в результаті взаємодії гамма-випромінювання з атомами навколишнього середовища. На утворення EMI йде невелика кількість ядерної енергії, але він здатен викликати високі імпульси струмів та напруг в кабелях повітряних і підземних ліній зв'язку, сигналізації, управління, електропередачі, в антенах радіостанцій. Вплив EMI може привести до згортання чутливих електронних та електричних елементів, зв'язаних з великими антенами чи відкритими дротами, а також до порушень в обчислювальних пристроях. Вплив EMI необхідно враховувати для всіх електричних та електронних систем. Для найбільш важливих приладів треба використовувати засоби захисту і підвищувати їх стійкість до EMI.

Особливістю EMI, як вражаючого фактору є його здатність розповсюджуватись на десятки і сотні кілометрів в оточуючому середовищі. Тому EMI може вплинути своєю дією на об'єкти, там де вибухова хвиля, світлове випромінювання, проникаюча радіація втрачають своє значення, як вражаючі фактори. При наземних та низьких повітряних вибухах в лініях зв'язку та електрозабезпечення виникають напруги, які можуть викликати пробій ізоляції провідників та кабелів відносно землі, пробій ізоляції елементів приладів підключених до повітряних і підземних ліній. Степінь враження залежить від наведеного імпульсу напруги чи струму і також електричної міцності обладнання.

Найбільш піддані впливу ЕМІ системи зв'язку, сигналізації, управління. Використані в цих системах кабелі та апаратура мають обмежену електричну міцність не більше 10кВ імпульсної напруги, тоді як наведені імпульси напруги від ЕМІ можуть перевищувати ці значення. Найбільш піддана впливу ЕМІ апаратура виконана на напівпровідниках та інтегральних схемах, працюючих на малих струмах і напругах, і значить відчутних до впливу зовнішніх електричних і магнітних кіл, в тому числі і елементи програмного засобу для управління процесом міграції віртуальних машин в обчислювальній хмарі. ЕМІ пробиває ізоляцію, спалює елементи електричних схем радіоапаратури, викликає коротке замикання в радіопристроях, іонізацію діелектриків, змінює або повністю стирає магнітний запис. Встановлено, що при дії ЕМІ на апаратуру найбільша напруга наводиться на вході. В транзисторах відбувається така залежність: чим більший коефіцієнт підсилення транзистора, тим менша його електрична міцність.

ЕМІ пошкоджує також резистори, викликає іскріння в їх міжконтактних з'єднаннях і деяких областях провідної поверхні. Найбільшу небезпеку ЕМІ представляє для апаратури, яка встановлена в особливо міцних спорудах, які витримують великі тиски ударної хвилі. В цих спорудах апаратура не виходить з ладу від механічних пошкоджень, але ЕМІ може вивести з ладу всю незахищену апаратуру системи зв'язку, сигналізації і керування. Найбільших значень досягають напруги, які наводяться між кабелем і землею. Напруженість електромагнітного поля всередині споруди в деяких випадках недостатня для того, щоб вивести з ладу апаратуру, але такі поля в змозі викликати короткочасний збій роботи радіотехнічних пристроїв.

Розглянемо можливі шляхи рішення задачі захисту від ЕМІ сервісу для адміністрування і обліку роботи автомобільної пар. Ідеальним захистом від ЕМІ виявилося б повне укриття приміщення, в якому розміщена радіоелектронна апаратура, металевим екраном. Водночас зрозуміло, що практично забезпечити такий захист у ряді випадків неможливо, тому що для роботи апаратури часто потрібно забезпечити її електричний зв'язок із

зовнішніми пристроями. Тому використовуються менш надійні засоби захисту, такі, як струмопровідні сітки, або плівкові покриття для вікон, щільникові металеві конструкції для повітрезабірників і вентиляційних отворів і контактні пружинні прокладки, розміщувані по периметру дверей і люків.

Більш складною технічною проблемою рахується захист від проникнення ЕМІ в апаратуру через різноманітні кабельні входи. Радикальним рішенням даної проблеми міг би стати перехід від електричних мереж зв'язку до практично не схильних до впливу ЕМІ волоконно-оптичних. Проте заміна напівпровідникових приладів у всьому спектрі виконуваних ними функцій електронно-оптичними пристроями можлива тільки у віддаленому майбутньому. Тому в даний час в якості засобів захисту кабельних входів найбільші широко використовуються фільтри, у тому числі волоконні, а також іскрові розрядники, металлоокисні варистори та ін. [28].

Всі ці засоби мають як переваги, так і недоліки. Так, ємнісно-індуктивні фільтри достатньо ефективні для захисту від ЕМІ малої інтенсивності, волоконні фільтри захищають у відносно вузькому діапазоні надвисоких частот. Іскрові розрядники мають значну інерційність й в основному придатні для захисту від перевантажень, що виникають під впливом напруг і струмів, що наводяться в обшивці літака, кожусі апаратури й оплетенні кабеля.

Металоокисні варистори є напівпровідниковими приладами, що різко підвищують свою провідність при високій напрузі. Проте, при застосуванні цих приладів у якості засобів захисту від ЕМІ варто враховувати їх недостатньо високу швидкодію і погіршення характеристик при кількаразовому впливі навантажень. Ці недоліки відсутні у високошвидкісних зенеровських діодах, дія яких заснована на різкій лавиноподібній зміні опору від високого значення практично до нуля, при перевищенні прикладеної до них напруги граничного розміру. Крім того на відміну від варисторів характеристики зенеровських діодів після багатократних впливів високих напруг і переключень режимів не погіршуються.

Найбільш раціональним підходом до проектування засобів захисту від ЕМІ кабельних входів є створення таких роз'ємів у конструкції яких передбачені спеціальні заходи, що забезпечують формування елементів фільтрів і установку вмонтованих зенеровських діодів. Подібне рішення сприяє одержанню дуже малих значень ємності й індуктивності, що необхідно для забезпечення захисту від імпульсів, що мають незначну тривалість і, отже, потужну високочастотну складову.

Складність рішення задачі захисту від ЕМІ і висока вартість розроблених для цих цілей засобів і методів змушують піти по шляху їхнього вибіркового застосування в особливо важливих системах зброї і військової техніки. Такий же шлях обраний і для захисту систем, що мають велику протяжність, керування і зв'язку. Проте основним методом рішення даної проблеми спеціалісти вважають створення так званих розподілених мереж зв'язку.

4.3 Висновки до четвертого розділу

В цьому розділі розглянуто важливі питання охорони праці та безпеки в надзвичайних ситуаціях, зокрема описано закордонний досвід організації охорони праці в ІТ-компаніях та виконано оцінку дії на елементи комп’ютерної системи.

ВИСНОВКИ

При проведенні дослідження моделей та розробки системи ПФПК були виконані такі завдання:

– вивчено існуючі технології прогнозування в економічній сфері;

– досліджено існуючу БД - інформації про компанії;

– виконано ПОД;

– створено різні моделі прогнозування, у тому числі за допомогою ШНМ із застосуванням методів ML;

– проведено порівняльний аналіз реалізованих методів прогнозування, складено зведену таблицю результатів точності моделей.

Крім перерахованих вище завдань, додатково отримано такі результати:

– вивчено алгоритм балансування SMOTE і впроваджено в систему, що розробляється;

– складено порівняльний аналіз результатів точності моделей на різних даних: збалансованих та незбалансованих;

– вивчено існуючі показники рентабельності компаній;

– виявлено переваги та недоліки окремих методів прогнозування;

– складено методологічний посібник із впровадження реалізованих моделей в інші проекти.

При вирішенні завдань у роботі довелося зіткнутися з різними труднощами, наприклад, великий дисбаланс класів, специфіка реалізації та роботи з ЩНМ, і навіть реалізація динамічного навчання моделей із засобів налаштування параметрів моделі. Варто відзначити, що всі ці проблеми були вирішені за рахунок популярності аналізу даних на мові python, тому що була можливість вивчити схожі, в технічному плані, системи, у тому числі роботи зі streamlit, та з бібліотекою sklearn.

Реалізована система ПФПК дозволить користувачам глибше дізнатися про роботу алгоритмів ML, ШНМ, дозволить вивчити специфіку роботи з

фінансовими даними, а також внесе свій внесок у популяризацію аналізу даних, у тому числі використання нейромереж.

Підсумки роботи кожного алгоритму з ПФПК дозволяють дійти висновку, що у навчанні моделі вирішальну роль грають самі дані, у цьому можна переконатися, якщо звернутися до таблиці 3.1, де відображені результати точності моделей на балансованих і незбалансованих даних. Також важлива і специфіка прогнозованих даних, оскільки конкретні методи, навіть у балансованих даних, можуть показати результати точності набагато гірше, проти іншими моделями.

Варто відзначити, що система, котра розробляється, при потенційній масштабованості, може бути використана в якості просунутого бізнес калькулятора з можливістю розрахунку конкретних показників користувачів.

ПЕРЕЛІК ДЖЕРЕЛ

1. Ravula S. Bankruptcy prediction using disclosure text features. // Annals of Applied Probability, 2020. Vol. 22, no. 3, 347-394.
2. Кучеренко О.А., Кучеренко О.О. Особливості передобробки даних для методів прогнозування // Інформаційні моделі, системи та технології: Праці XI наук.-техн. конф. - Тернопіль, 2023. - С. 72.
3. Kaggle: Your Machine Learning and Data Science Community [Електронний ресурс] – Режим доступу: https://www.kaggle.com/ (дата звертання 01.12.2023).
4. Yuichi I. An Interacting Agent Model of Economic Crisis. Complexity, Heterogeneity and the Methods of Statistical Physics in Economics. // Computational Systems Engineering, 2020. Vol. 11, no. 1, 30 – 39.
5. Falco J., Niederreiter J., Massimo R. Supervised learning for the prediction of firm dynamics. // Computational Systems Engineering, 2020. Vol. 1, no. 4, 1-10.
6. Бідюк, П. І. Прикладна статистика / П. І. Бідюк, О. М. Терентьєв, Т. І. Просянкіна-Жарова. Вінниця : ПП "ТД"Едельвейс і К", 2013. 304 с.
7. White C. Unkind cuts at statisticians. // The American Statistician Journal, 1964. Vol. 18, no. 5, 15-17.
8. Ian Goodfellow, Yoshua Bengio, Aaron Courville. Deep Learning (Adaptive Computation and Machine Learning series), MIT Press book, 2016
9. Soumyadip G., Yingdong, L., Nowicki, T. HMC, Algorithms in Data Mining, the Functional Analysis approach. // Int. J. Computational Systems Engineering, 2021. Vol. 1, no. 2, 124-129.
10. Kent B. Machine learning. // Computational Systems Engineering, 2018. Vol. 1, no. 1, 5-10.
11. Ibragimov R., Pedersen R,. Skrobotov A. New Approaches to Robust Inference on Market Efficiency, Volatility Clustering and Nonlinear Dependence. // Annals of Applied Probability, 2020. Vol. 2, no. 4, 12-50.

12. Lancelot F. James. Analysis of a Class of Likelihood Based Continuous Time Stochastic Volatility Models including Ornstein-Uhlenbeck Models in Financial Economics. // ACM Computing Surveys, 2005. Vol. 3, no. 3, 36—45.

13. Wei J. Research on Machine Learning and Its Algorithms and Development. // Journal of Physics: Conference Series, 2020. Vol. 10544, no. 5, 25-35.

14. Paolo Dai Pra S., Wolfgang J. R., Sartori E., Tolotti M. Large portfolio losses: A dynamic contagion model. // Annals of Applied Probability, 2009. Vol. 19, no. 1, 347-394.

15. Greenland S. There are natural scores: Full comment on Shafer, "Testing by betting: A strategy for statistical and scientific communication". // Annals of Applied Probability, 2020. Vol. 100, no. 6, 300-324.

16. Zhi-Qiang J., Wen-Jie X., Wei-Xing Z. Multifractal analysis of financial markets: a review. // Int. J. Computational Systems Engineering, 2019. Vol. 82, no. 12, 150-172.

17. Licong L., Edgar D. What causes the test error? Going beyond biasvariance via ANOVA. // Cambridge University Press, 2021. Vol. 2, no. 1, 20- 50.

18. Caldarelli G. A perspective on complexity and networks science. // Journal of Physics: Complexity, 2020. Vol. 1, no. 2, 20-26.

19. Lee C., Miaoyan W. Beyond the Signs: Nonparametric Tensor Completion via Sign Series. // ACM Computing Surveys, 2021. Vol. 1, no. 10, 150-151.

20. Subhrajit S., Umesh V., Enoch Y. On Few Shot Learning of Dynamical Systems: A Koopman Operator Theoretic Approach. // ACM Computing Surveys, 2021. Vol. 1, no. 1, 10-20.

21. Сьогодні UA [Електронний ресурс] – Режим доступу: https://www.segodnya.ua/lifestyle/fun/pochti-kak-u-googlechemudivlyayut-ofisy-ukrainskih-it-kompaniy--764025.html (дата звертання 07.12.2023).

22. Як працюють в Google? Умови, в яких хочеться трудитися. [Електронний ресурс] – Режим доступу: http://www.clevers.com.ua/articles-cleveradvertising-agency/success-stories/245-google2 (дата звертання 07.12.2023).

23. Офіс мрії: Робота в компанії Google. [Електронний ресурс] – Режим доступу:http://bigpicture.ru/?p=187580 (дата звертання 07.12.2023).

24. Офіс Facebook: Репортаж із RMA SiliconTrip. [Електронний ресурс] – Режим доступу:https://habrahabr.ru/company/rma/blog/103800/ (дата звертання 08.12.2023).

25. Класифікатор професій ДК 003:2010. [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/rada/show/va327609-10 - (дата звертання 07.12.2023).

26. Закон України «Про охорону праці». [Електронний ресурс] – Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12 - (дата звертання 06.12.2023).

27. ДНАОП 0.00-8.20-99. Порядок проведення експертизи електроустановок споживачів/ [Електронний ресурс] – Режим доступу: https://dnaop.com/html/43255/doc-%D0%94%D0%9D%D0%90%D0%9E%D0 %9F _ 0.00-8.20-99 - (дата звертання 07.12.2023).

28. Зеркалов Д.В. Безпека життєдіяльності та основи охорони праці. Навч. посібник. К.: «Основа». 2016. 267 с.

ДОДАТКИ

ДОДАТОК А
Тези
конференції

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

XI НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

13-14 грудня 2023 року

ТЕРНОПІЛЬ
2023

УДК 004.4:004.05
О.А. Кучеренко[1], О.О. Кучеренко[2]
[1]Тернопільський національний технічний університет імені Івана Пулюя, Україна
[2]Національний університет «Львівська політехніка», Україна

ОСОБЛИВОСТІ ПЕРЕДОБРОБКИ ДАНИХ ДЛЯ МЕТОДІВ ПРОГНОЗУВАННЯ

O.A.Kucherenko, O.O.Kucherenko

FEATURES DATA PREPROCESSING FOR FORECASTING METHODS

Попередня обробка даних (ПОД) у машинному навчанні (МН) - це важливий крок, який допомагає підвищити якість даних та сприяти вилученню з них змістовної інформації. Фактично - це метод підготовки (очищення та систематизації) необроблених даних, щоб зробити їх придатними для побудови та навчання моделей МН. Простіше кажучи, ПОД у МН - це метод інтелектуального аналізу даних, котрий трансформує необроблені дані в зрозумілий формат.

Коли йдеться про побудову моделі МН, така ПОД є першим кроком, що знаменує початок процесу. Як правило, реальні дані є неповними, непослідовними, неточними (містять помилки чи викиди) та часто не мають конкретних значень/тенденцій атрибутів. Саме тут у сценарій входить ПОД - вона допомагає очищати, форматувати і систематизувати необроблені дані, тим самим роблячи їх готовими до використання в моделях МН.

Можна схематично виділити етапи ПОД у МН:

– отримання даних. Щоб побудувати та розробити моделі МН, необхідно спочатку отримати відповідну кількість даних. Вона буде складатися з даних, зібраних з кількох та розрізнених джерел, які потім будуть об'єднані у належному форматі для формування. Формати таких наборів даних (НД) різняться залежно від застосування сценарію. Для прикладу, набір бізнес-даних абсолютно інший, ніж НД для медицини;

– імпорт бібліотеки. Найбільш використовуваними бібліотеками при обробці даних є бібліотеки pandas та numpy;

– імпорт даних. На цьому етапі необхідно імпортувати НД, зібраних для поточного проекту МН. У процесі імпорту НД необхідно витягти залежні та незалежні змінні. Для кожної моделі МН необхідно розділити незалежні змінні (матрицю функцій) та залежні змінні НД. Щоб отримати незалежні змінні, необхідно використовувати функцію iloc[] бібліотеки Pandas;

– виявлення та обробка відсутніх значень. При ПОД дуже важливо ідентифікувати та правильно обробляти відсутні значення, в іншому випадку ви можете зробити неточні та помилкові висновки на основі даних. Зайве говорити, що це завадить вашому проекту МН. У БД немає пропущених значень, тому цей крок опускається;

– кодування категоріальних даних. Вони належать до інформації, що має певні категорії в НД. Моделі МН, в основному, грунтуються на математичних рівняннях. У НД категоріальної змінної є стовпець з інформацією про рентабельність підприємства;

– розділення НД. Будь-який НД для моделі МН повинен бути поділеним на два окремих НД - навчальний і тестовий. Перший – це підмножина НД, котра застосовується для навчання моделі МН. Другий – це підмножина НД для тестування моделі МН. Зазвичай НД розбивається на співвідношення 70% та 30% або співвідношення 80% та 20%. Процес поділу залежить від форми та розміру аналізованого НД;

– масштабування функцій. Означає кінець ПОД у МН. Це метод стандартизації незалежних змінних НД у межах певного діапазону.