МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «магістр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)
на тему:

«Розробка веб-сервісу для прогнозного моделювання та аналітики успішності студентів з використанням алгоритмів машинного навчання»

Виконав студент групи ІПЗм-21-1
ЗУБРИЦЬКИЙ Василь Вікторович

Керівник роботи:
ВАКАЛЮК Тетяна Анатоліївна

Рецензент:
ЄФРЕМОВ Юрій Миколайович

Житомир – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»
В.о. зав. кафедри інженерії
програмного забезпечення
Андрій МОРОЗОВ
«12» вересня 2022 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ЗУБРИЦЬКИЙ Василь Вікторович**
Керівник роботи: **ВАКАЛЮК Тетяна Анатоліївна**
Тема роботи: **«Розробка веб-сервісу для прогнозного моделювання та аналітики успішності студентів з використанням алгоритмів машинного навчання»**,
затверджена Наказом закладу вищої освіти від **«12» вересня 2022 р. № 356/с**
Вихідні дані для роботи: **Предметом дослідження є методи та моделі для прогнозування успішного завершення курсу студентами та ймовірність його закінчення. Об'єктом дослідження є прогнозування успішного завершення курсу студентами та ймовірність його закінчення.**
Консультанти з магістерської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	**Консультант**	Завдання видав	Завдання прийняв
1	Вакалюк Т.А.	01.05.2023	01.05.2023
2	Вакалюк Т.А.	01.05.2023	01.05.2023
3	Вакалюк Т.А.	01.05.2023	01.05.2023
4	Вакалюк Т.А.	01.05.2023	01.05.2023

РЕФЕРАТ

Кваліфікаційна робота присвячена науковому дослідженню використання методів та моделей інтелектуального аналізу даних в електронному навчанні. Пояснювальна записка складається з вступу, чотирьох розділів, висновків та списку використаної літератури.

Кваліфікаційна робота складається з програмного додатку, який включає в себе серверне API та веб-інтерфейс, а також з пояснювальної записки. Текстова частина викладена на 60 сторінках друкованого тексту. Пояснювальна записка включає список використаних джерел, що містить 19 найменувань і займає 2 сторінки. В роботі наведено 36 рисунків.

Кваліфікаційна робота складається з чотирьох розділів. В першому розділі було визначено основні вимоги, проведено аналіз задачі, а також досліджено методи та алгоритми для її вирішення.

В другому розділі було проведено аналіз аналогів програмного забезпечення та здійснено постановку завдання.

В третьому розділі о проведено здійснено практичну реалізацію алгоритму, проаналізовано отримані метрики, та проведено приклад використання отриманих даних.

В четвертому розділі було розглянуто серверне API, веб-інтерфейс додатку та проведено його тестування.

Висновок містить в собі підсумки проведеного дослідження.

Ключові слова: LogisticRegression, RandomForest, XGBoost, СУН, ЛОГІСТИЧНА РЕГРЕСІЯ, ДЕРЕВО РІШЕНЬ, ВИПАДКОВИЙ ЛІС, ЕЛЕКТРОННІ КУРСИ, СТУДЕНТ

ABSTRACT

Qualification work devoted to the scientific study of the use of methods and models of intellectual data analysis in electronic learning. The explanatory note consists of an introduction, four chapters, conclusions and a list of used literature.

The qualification work consists of a software application, which includes a server API and a web interface, as well as an explanatory note. The text part is laid out on 60 pages of printed text. The explanatory note contains a list of used sources, which contains 19 names and occupies 2 pages. There are 36 drawings in the work.

The qualification work consists of four chapters. In the first chapter, the main requirements were defined, the tasks were analyzed, and the methods and algorithms for solving them were investigated.

In the second chapter, an analysis of software analogues was carried out and the task was set.

In the third chapter, a real practical implementation of the algorithm was carried out, the received metrics were analyzed, and an example of using the received data was provided.

In the fourth chapter, the server API, web interface application was observed and tested.

The conclusion contains results of the conducted research.

Keywords: LogisticRegression, RandomForest, XGBoost, LMS, LOGISTIC REGRESSION, DECISION TREE, RANDOM FOREST, ELECTRONIC COURSES, STUDENT

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 6
ВСТУП 7
РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ 9
1.1 Аналіз задачі, засобів та методів її вирішення 9
1.2 Визначення алгоритму машинного навчання 10
1.3 Методи та алгоритми інтелектуального аналізу даних для машинного навчання 14
РОЗДІЛ 2. ПОРІВНЯННЯ НАЯВНИХ ПРОГРАМНИХ ПРОДУКТІВ ТА ПОСТАНОВКА ЗАВДАННЯ 19
2.1 Порівняння наявних програмних продуктів 19
2.2 Постановка завдання 21
РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ 24
3.1 Характеристика джерел даних для проведення аналізу 24
3.2 Побудова алгоритму машинного навчання та його практична реалізація 26
3.3 Аналіз результатів та визначення метрик для оцінки моделей 33
РОЗДІЛ 4. КОМПОНУВАННЯ АЛГОРИТМУ МАШИННОГО НАВЧАННЯ, НАПИСАННЯ СЕРВЕРНОГО API ТА ВЕБ-ІНТЕРФЕЙСУ 39
4.1 Розробка серверного API для алгоритму машинного навчання 39
4.2 Розробка веб-інтерфейсу 44
ВИСНОВКИ 54
ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ 56

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

XGBoost – extreme gradient boosting

ML – Machine learning

СУН – система управління навчанням

API – Application Programming Interface

ВСТУП

Оскільки ви ніколи не можете на сто відсотків передбачити, що може принести майбутнє, деякі практики майже допоможуть вам з планами на майбутнє. Прогнозна аналітика є однією з таких практик.

Прогнозна аналітика стосується використання алгоритмів машинного навчання та статистики для прогнозування майбутніх результатів і продуктивності. Такі методи, як інтелектуальний аналіз даних і прогнозне моделювання, оцінюють ймовірність майбутніх результатів і сповіщають вас про майбутні події, щоб допомогти вам прийняти рішення.

Прогнозна аналітика допомагає передбачити майбутні тенденції та закономірності, використовуючи історичні дані. Він використовує різні шаблони даних і визначає кореляції між змінними. Це допомагає зменшити бізнес-ризики та витрати, прогнозуючи майбутні значення певних змінних.

Актуальність теми обумовлена необхідністю оптимізації рішень для бізнесу.

Подальшого дослідження потребує проблема правильного вибору моделей для прогнозного аналізу.

Метою роботи є аналіз моделей, методів прогнозного аналізу та вибір сучасних технологій для тренування даних моделей. Для досягнення поставленої мети необхідно вирішити такі основні завдання:

- визначити основні вимоги та бажані кінцеві результати;
- обрати підходящу модель та методи попередньої обробки даних;
- протестувати роботу моделі;
- створити API для взаємодії з моделями;
- створити веб-інтерфейс для проведення тестування та загального використання моделей.

Об’єктом дослідження є прогнозування успішного завершення курсу студентами та ймовірність його закінчення.

Предметом дослідження є методи та моделі для прогнозування успішного завершення курсу студентами та ймовірність його закінчення.

Наукова новизна полягає в застосуванні машинного навчання (ML) в даній галузі, що відкриває нові можливості для аналізу великих обсягів даних, виявлення складних залежностей і розробки персоналізованих підходів до навчання. Сучасні системи управління навчанням та електронні платформи зберігають значну кількість інформації про навчальні досягнення студентів, таку як оцінки, відвідуваність, здані іспити тощо.

Практичне значення полягає у використанні цих даних з систем управління навчанням та навчальних закладів з метою виявлення корисних закономірностей та прогнозування майбутніх успіхів студентів.

Матеріали дослідження за тематикою кваліфікаційної роботи доповідалася на II Міжнародній студентській науковій конференції (Т. 2), м. Одеса, 17 грудня, 2021 рік / ГО «Молодіжна наукова ліга» [2], а також на XXIV Міжнародна науково-практична інтернет-конференція «Сучасні виклики і актуальні проблеми науки, освіти та виробництва: міжгалузеві диспути». [1]

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

1.1 Аналіз задачі, засобів та методів її вирішення

Сьогодні аналіз даних перебуває в центрі уваги всіх галузей. Якщо підприємство не використовує аналіз накопичених даних, тоді можна говорити про обраний хибний та некомпетентний підхід у веденні бізнесу на відміну від конкурентів, які займаються даним процесом. Аналіз даних дає змогу підприємству вживати активних заходів і створювати конкурентні переваги у своїй галузі.

Систематичне застосування статистичних і логічних методів для опису обсягу даних, модульності структури даних, стиснення представлення даних, ілюстрації за допомогою зображень, таблиць і графіків, а також оцінки статистичних нахилів, даних про ймовірність і отримання значущих висновків, відомі як аналіз даних. Найбільш зацікавлені у використанні методів та інструментів аналізу даних є комерційні підприємства, що розгортають свої проекти на основі інформаційних сховищ даних [4]. Забезпечення цілісності даних є одним із найважливіших компонентів аналізу даних. Розглянемо деякі методи інтелектуального аналізу даних.

Регресійний аналіз — метод, який працює шляхом моделювання зв'язку між залежною змінною та однією або кількома незалежними змінними. Модель регресії може бути лінійною, множинною, логістичною, нелінійною, життєвими даними тощо [3]. Часто особливий інтерес приділяється оцінці впливу кожної незалежної змінної на залежну змінну, і такий ефект розглядається як середнє значення ефекту для всіх суб’єктів у вибірці. Наприклад, якщо бали з математики 500 студентів регресують за показником їхньої мотивації, значення нахилу або коефіцієнт регресії кількісно визначає середню зміну досягнень з математики для всіх 500 студентів на одну одиницю зміни мотивації. Це означає, що ці 500 студентів розглядаються як одна однорідна група щодо впливу мотивації на

досягнення з математики, а неявне припущення полягає в тому, що ці студенти є з однієї групи населення зі схожими характеристиками.

Інтерес до вивчення аналітики навчання, пов’язаної із залученням студентів, останнім часом значно зростає. Це ще більше розширило дослідницьке поле для освіти. Вищі навчальні заклади виявили свою зацікавленість у використанні аналітики для підтримки їхньої участі. Це може діяти як інструмент, який допоможе в посередництві обміну інформацією між студентами та викладача-ми, що призведе до ефективного навчання, підвищення обізнаності та способу вирішення поточних складних ситуацій

У цьому прогнозному аналізі ми розглянемо данні студентів які проходять курс на електронній системі управління навчанням, яка має такий самий набір даних із студентами, які проходили цей курс в минулому. Ця організація розглядає успішне завершення курсу студентами та ймовірність його закінчення, тобто іншими словами, зацікавленість курсу студентом.

Прогнозний аналіз тут дозволяє нам визначити студентів, які, найімовірніше, закінчать цей курс успішно. Також для успішного прогнозування потрібно мати набір попередньо оброблених даних.

Визначивши основні цілі, можна приступити до розгляду методів для їхнього досягнення.

1.2 Визначення алгоритму машинного навчання

Експериментальне дослідження — це, по суті, дослідження однієї або кількох змінних (залежних змінних), якими маніпулюють для оцінки впливу на одну або більше змінних, відомих як незалежні змінні. Він базується на причинно-наслідкових зв’язках у вибраному предметі, щоб зробити висновок про різні зв’язки, які може створити продукт, теорія чи ідея [12]. Відношення серед змінних встановлюється за допомогою точної та систематичної маніпуляції. Ця методика підходить, коли в основі дослідження лежить теорія тестування та оцінка методів. Крім того, ту саму настройку та протокол можна відтворити за допомогою тих самих змінних. Це може підтвердити достовірність

продуктів, ідей і теорій. [9]. Крім того, цей тип наукового підходу може надати набір інструкцій для оцінювання та звітування інформації для дослідження [7]. На рисунку 1.3 нижче показаний популярний загальний перелік етапів того, як проводяться експерименти до досягнення оцінки моделі.

Машинне навчання має власний життєвий цикл, тобто процес, який проходять дані для розробки та розгортання системи прогнозування. Порівняно з життєвим циклом розробки програмного забезпечення, розробка моделей машинного навчання передбачає експериментування з наборами даних для досягнення цілей і завдань, визначених під час застосування свіжих даних після навчання. Розглянемо кожен з етапів більш детально.

1. **Збір даних.** Як відомо, машини спочатку навчаються на даних, які ви їм надаєте. Надзвичайно важливо збирати надійні дані, щоб ваша модель машинного навчання могла знайти правильні шаблони. Якість даних, які ви подаєте на машину, визначатиме, наскільки точна ваша модель. Якщо у вас є неправильні або застарілі дані, ви матимете неправильні результати або прогнози, які не є актуальними.Переконайтеся, що ви використовуєте дані з надійного джерела, оскільки це безпосередньо вплине на результат вашої моделі. Хороші дані є релевантними, містять дуже мало пропущених і повторюваних значень і добре представляють різні наявні підкатегорії/класи.
2. **Попередня обробка даних.** Отримавши дані, їх потрібно підготувати. Ви можете зробити це за допомогою:
    a. Об’єднання всіх наявних у вас даних і їх рандомізація. Це допомагає переконатися, що дані розподіляються рівномірно, а порядок не впливає на процес навчання.
    b. Очищення даних для видалення непотрібних даних, відсутніх значень, рядків і стовпців, повторюваних значень, перетворення типів даних тощо. Можливо, вам навіть доведеться реструктурувати набір даних і змінити рядки та стовпці або індекси рядків і стовпців.
    c. Візуалізуйте дані, щоб зрозуміти, як вони структуровані, і зрозуміти взаємозв’язок між різними змінними та присутніми класами.

d. Розбиття очищених даних на два набори - набір для навчання та набір для тестування. Навчальний набір – це набір, з якого вчиться ваша модель. Тестовий набір використовується для перевірки точності вашої моделі після навчання.

3. **Вибір моделі машинного навчання.** Модель машинного навчання визначає результати, які ви отримуєте після запуску алгоритму машинного навчання на зібраних даних. Важливо підібрати модель, яка відповідає поставленим завданням. Протягом багатьох років вчені та інженери розробляли різні моделі, які підходять для різних завдань, таких як розпізнавання мовлення, розпізнавання зображень, прогнозування тощо. Окрім цього, ви також повинні перевірити, чи підходить ваша модель для числових чи категоріальних даних, і вибрати відповідно.
4. **Навчання моделі** є найважливішим кроком у машинному навчанні. Під час навчання ви передаєте підготовлені дані своїй моделі машинного навчання, щоб знаходити закономірності та робити прогнози. Це призводить до того, що модель навчається на даних, щоб вона могла виконати поставлене завдання. З часом, під час навчання, модель стає краще прогнозувати.
5. **Покращення параметрів моделі.** Після того, як ви створили та оцінили свою модель, подивіться, чи можна якимось чином підвищити її точність. Це робиться шляхом налаштування параметрів, наявних у вашій моделі. Параметри - це змінні в моделі, які зазвичай вибирає програміст. При конкретному значенні вашого параметра точність буде максимальною. Налаштування параметрів стосується пошуку цих значень.
6. **Використання попередньо натренованої моделі на абсолютно нових даних.**

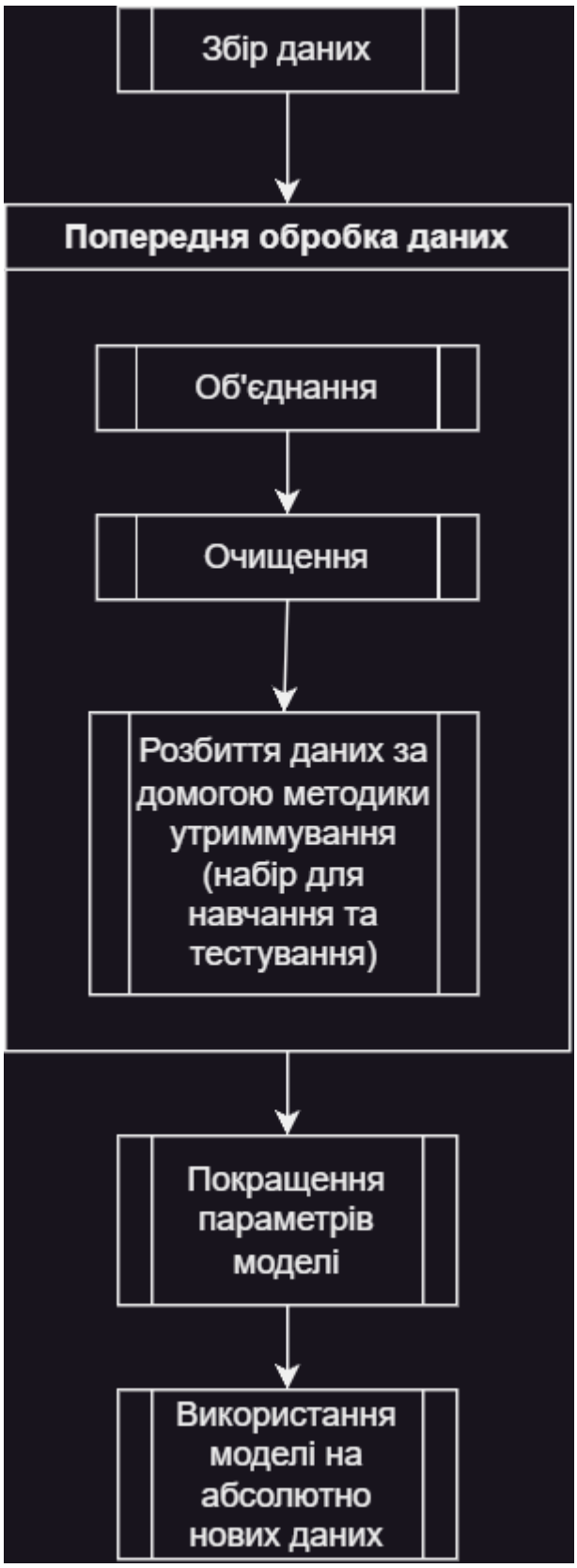


Рис. 1.1 Перелік етапів експериментів з моделями та методами, що були описані раніше

Оцінка прогнозної моделі є важливою частиною для визначення точності успішності студента. Для цього важливо кількісно оцінити якість прогнозів

системи [4]. Розглянемо деякі важливі показники ефективності для оцінки методів машинного навчання:

1. Точність - це кількість правильних позитивних результатів, поділена на всі зразки, позначені алгоритмом як позитивні.

2. Правильність - визначається як відношення правильних прогнозів до загальної кількості введених зразків. Це часто використовувана метрика для оцінки якості рішень класифікатора. Це найбільш використовувана метрика оцінки як для бінарної, так і для багатокласової класифікації. Це визначальне значення для оцінки можливостей алгоритму.

3. Відкликання (чутливість) - це кількість правильних позитивних результатів, поділена на всі зразки, які алгоритм повинен був позначити як позитивні.

1.3 Методи та алгоритми інтелектуального аналізу даних для машинного навчання

На відміну від будь-яких інших показників, прогнозна аналітика навчання є більш ефективною, оскільки зосереджена на окремому студенті, а не на системі керування навчанням в цілому. Це робить прогнозну аналітику надзвичайно впливовою у вирішенні проблем, які спричиняють неефективне навчання. Це дозволяє керівникам визначати прогрес, досягнутий співробітниками на своїх курсах. Ця технологія допомагає зрозуміти, наскільки добре студенти отримали вигоду від курсів і чи збираються вони застосовувати отримані знання на практиці.

Ефективність рішення машинного навчання залежить від природи набору даних і продуктивності алгоритмів. Вибір правильного алгоритму навчання, який підходить для застосування в конкретній області, є не простою справою. Причина цього полягає в тому, що призначення алгоритмів ML є різними. Навіть результати різних алгоритмів навчання в подібній категорії можуть відрізнятися залежно від характеристик даних [10]. Багато алгоритмів машинного навчання

було реалізовано в дослідницькому співтоваристві. Розглянемо найважливіші і відомі методи, які фігурують у науковій літературі.

Логістична регресія[1] (або логіт-регресія) — це оцінка параметрів логістичної моделі (коефіцієнтів у лінійній комбінації). Формально в бінарній логістичній регресії існує одна двійкова залежна змінна, кодована змінною-індикатором, де два значення позначені «0» і «1», тоді як кожна з незалежних змінних може бути двійковою змінною (два класи, кодовані як індикаторна змінна) або безперервна змінна (будь-яке реальне значення). Відповідна ймовірність значення, позначеного як «1», може коливатися від 0 (звичайно, значення «0») до 1 (безумовно, значення «1»), отже, позначення; [2] функція, яка перетворює логарифмічні шанси на ймовірність, є логістична функція, звідси і назва. Одиниця вимірювання для логарифмічної шкали шансів називається логіт, від logistic unit, звідси альтернативні назви. Формула логістичної регресії, як незалежні змінні впливають на залежну змінну [2]:

$$P(X) = \frac{1}{1} + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n)},$$

де P – у логістичній моделі p(x) інтерпретується як ймовірність того, що залежна змінна Y дорівнює успіху або підпадає під класифікацію випадку, або не підпадає під класифікацію випадку. Це важливо, оскільки воно показує, що значення виразу лінійної регресії може змінюватися від негативної до позитивної нескінченності, і все ж після перетворення результуючий вираз для ймовірності P(X) коливається між 0 і 1;

B1,β2,…… βn є невідомими параметрами (коефіцієнтами);

β0 є β0 є константою для створення лінії найкращого підходу. Метою логістичної регресії є зіставлення функції з характеристик набору даних на цілі для обчислення ймовірності того, що новий запис належить до одного з цільових класів.

Дерево рішень має деревоподібну структуру, де кожен вузол показує атрибут, кожне посилання показує рішення (правило), а кожен лист показує результат. Його можна використовувати як для безперервних, так і для

дискретних наборів даних [3]. Дерево рішень починається з кореневого вузла. З цього вузла користувачі рекурсивно розділяють кожен вузол відповідно до алгоритму навчання дерева рішень на основі запитань «якщо» [8]. Результатом є дерево рішень, у якому кожна гілка представляє можливий сценарій рішення та його результат. Приклад дерева рішень показано на рис. 1 нижче.

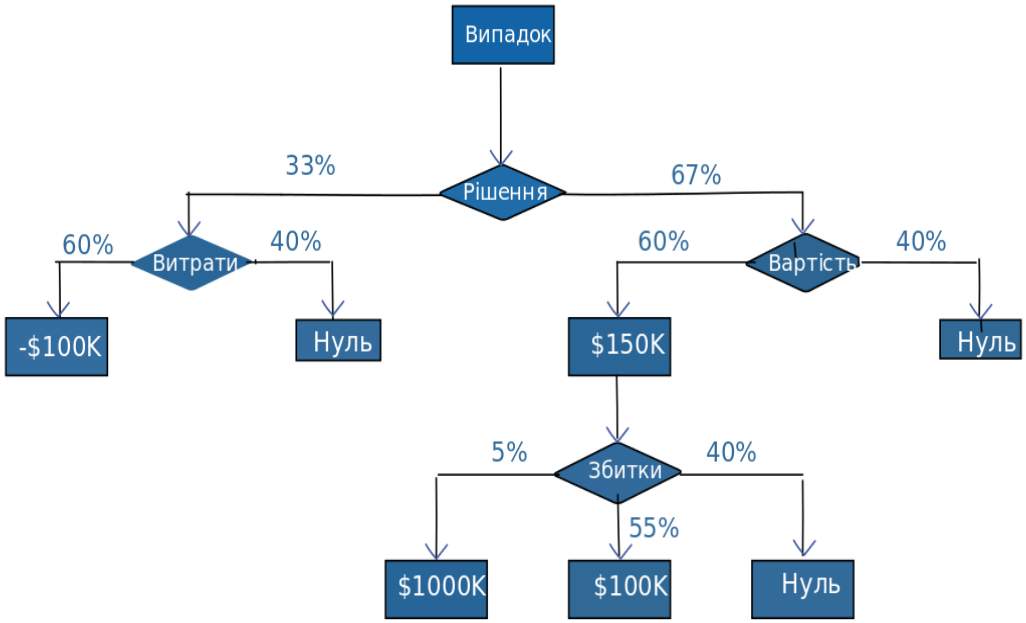


Рис. 1.2 Структура алгоритму дерева рішень

Випадковий ліс (RandomForest) - вважається експертним рішенням для більшості проблем і підпадає під класифікатори ансамблевого навчання, за допомогою яких слабкі моделі поєднуються для створення потужної. Ансамблеві методи є одними з найбільш перспективних напрямів дослідження. Він визначається як набір класифікаторів, прогнози яких об’єднуються для прогнозування нових випадків. Ансамблеві алгоритми навчання показали себе як ефективний метод для підвищення точності прогнозування та послаблення складності проблем навчання у підпроблеми. Численні дерева рішень створюються у випадкових лісах. Щоб класифікувати об’єкт, що має атрибути, кожне з дерев дає класифікацію, яка також вважається голосом. Тоді лісу надається можливість вибрати класифікацію з максимальною кількістю голосів. Це показано на рис. 1.2 нижче.

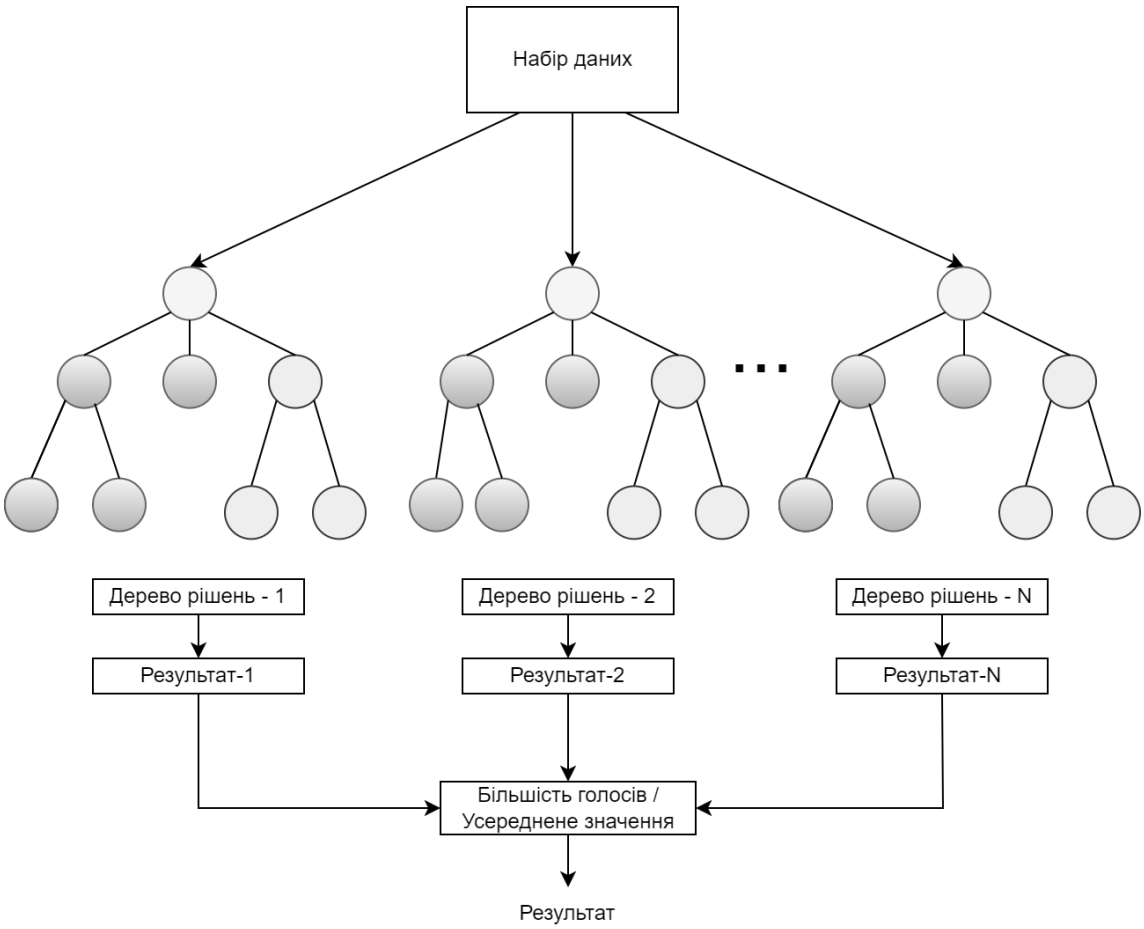


Рис. 1.3 Приклад випадкової лісової структури з урахуванням кількох результатів дерева рішень

Збільшення градієнта – це техніка машинного навчання, яка використовується, зокрема, у завданнях регресії та класифікації. Він дає модель прогнозування у формі ансамблю слабких моделей прогнозування, які зазвичай є деревами рішень. Коли дерево рішень є слабким навчальним елементом, отриманий алгоритм називається деревом із посиленням градієнта; зазвичай він перевершує випадковий ліс. Модель дерев із посиленням градієнта будується поетапно, як і в інших методах посилення, але вона узагальнює інші методи, дозволяючи оптимізувати довільну диференційовану функцію втрат.

XGBoost є скороченням від «Extreme Gradient Boosting». «Екстремальний» стосується підвищення швидкості, наприклад паралельних обчислень і усвідомлення кешу, що робить XGBoost приблизно в 10 разів швидшим, ніж традиційний Gradient Boosting. Крім того, XGBoost включає унікальний алгоритм пошуку поділу для оптимізації дерев разом із вбудованою

регуляризацією, яка зменшує переобладнання. Загалом, XGBoost — це швидша та точніша версія Gradient Boosting.

XGBoost тренується на залишках. Замість агрегування дерев, підсилені градієнтом дерева навчаються на помилках під час кожного раунду посилення.

Висновки до розділу 1

В першому розділі було розглянуто основну проблематику кваліфікаційної роботи. На основі цього було визначено напрямок подальшого дослідження можливих варіантів вирішення поставленої проблеми.

Найкращим рішенням може бути використання алгоритмів машинного навчання для прогнозування успішності студентів на основі різних факторів. Ці моделі спрямовані на виявлення закономірностей і зв’язків у даних, щоб спрогнозувати, наскільки добре студент, ймовірно, успішно навчатиметься. Варто зазначити, що вибір моделі прогнозування залежить від наявних даних, характеру завдання прогнозування та конкретних вимог освітнього контексту.

Опираючись на матеріал розглянутий в даному розділі можна приступити до пошуку наявних аналогів програмного забезпечення та постановки конкретних задач, які повинні будуть виконані в кваліфікаційній роботі.

РОЗДІЛ 2. ПОРІВНЯННЯ НАЯВНИХ ПРОГРАМНИХ ПРОДУКТІВ ТА ПОСТАНОВКА ЗАВДАННЯ

2.1 Порівняння наявних програмних продуктів

Програмне забезпечення для прогнозування успішності студентів використовує розширені методи аналізу даних і машинного навчання для прогнозування та визначення успішності студентів і потенційних проблем. Аналізуючи історичні дані студентів, ці програмні рішення створюють прогнози, які допомагають викладачам і навчальним закладам приймати зважені рішення та вчасно втручатися. Ці інструменти допомагають виявити студентів групи ризику, підвищити рівень утримання та покращити загальну успішність студентів. Використовуючи прогнозну аналітику, викладачі отримують цінну інформацію про моделі успішності студентів, що дозволяє їм впроваджувати цільові стратегії підтримки та персоналізовані підходи до навчання. Програмне забезпечення для прогнозування успішності студентів є потужним інструментом для оптимізації результатів навчання, сприяння залученню студентів і створення проактивного навчального середовища.

Розглянемо деякі плюси та мінуси використання платформ прогнозування успішності студентів.

Переваги:

1. Раннє виявлення студентів групи ризику: ці платформи використовують прогнозну аналітику для виявлення студентів, які можуть мати труднощі в навчанні, що дозволяє викладачам втрутитися на ранній стадії та надати своєчасну підтримку.

2. Персоналізовані втручання: інформація, надана цими платформами, допомагає педагогам адаптувати втручання та стратегії підтримки для задоволення конкретних потреб окремих студентів, збільшуючи їхні шанси на успіх.

3. Покращення результатів студентів: за допомогою аналізу даних і прогнозного моделювання ці платформи спрямовані на покращення загальних

результатів студентів, включаючи академічну успішність, рівень випуску та рівень утримання.

4. Прийняття рішень на основі даних: платформи надають викладачам, адміністраторам і консультантам цінну інформацію на основі даних, що дає їм змогу приймати обґрунтовані рішення щодо навчальних програм, ресурсів та ініціатив щодо підтримки студентів.

Недоліки:

1. Довіра до точних і актуальних даних. Точність і актуальність прогнозів значною мірою залежить від якості та доступності даних. Неповні або застарілі дані можуть вплинути на ефективність прогнозів.

2. Вимоги до технічної експертизи та інфраструктури: впровадження та підтримка цих платформ може вимагати технічної експертизи та відповідної інфраструктури для зберігання, інтеграції та аналізу даних.

3. Обмежений діапазон прогнозів: прогнозні моделі мають властиві обмеження і можуть не охоплювати всі фактори, які впливають на успішність студентів. Неакадемічні чинники, такі як особисті обставини чи зовнішні події, можуть бути недостатньо враховані.

4. Можливість неправильного тлумачення: Прогностична аналітика повинна використовуватися як інструмент підтримки прийняття рішень, але не повинна замінювати професійне судження педагогів. Існує ризик неправильно витлумачити прогнози та покладатися виключно на них без урахування інших факторів.

Найбільш поширеними в світі є:

1. **OpenEduCat** (https://www.openeducat.org/) - це інформаційна система для студентів з відкритим кодом, яка містить функції для прогнозування успішності студентів. Він використовує аналітику даних і алгоритми машинного навчання для створення прогнозів на основі історичних даних студентів. Основні особливості:

- Відкритий кодом (open-source).
- Використовує аналіз даних і алгоритми машинного навчання.

- Прогнозує успішність студентів на основі історичних даних.
- Надає повний набір функцій для керування інформацією про студентів.
- Пропонує додаткове налаштування, бо має відкритий код.
- Потрібні технічні знання для налаштування та обслуговування.

2. Predictive Student Analytics (https://www.civitaslearning.com/) - це програмне забезпечення для прогнозування успішності студентів, розроблене Civitas Learning. Він використовує моделі машинного навчання для виявлення студентів групи ризику та надає корисну інформацію для покращення їхніх результатів. Він також пропонує інструменти для академічних консультантів і адміністраторів, щоб втручатися та підтримувати студентів, які мають труднощі. Основні особливості:

- Розроблено Civitas Learning.
- Використовує моделі машинного навчання для виявлення студентів групи ризику.
- Надає практичну інформацію для підтримки та втручання студентів.
- Призначений для покращення успішності студентів і показників утримання.
- Пропонує інструменти для наукових радників і адміністраторів.
- Потрібна інтеграція з існуючими інформаційними системами студентів.

3. Blackboard Predict (https://www.blackboard.com/teaching-learning/data-and-analytics) - це програмне забезпечення для аналізу навчання, яке пропонує Blackboard Inc. Основні особливості:

- Прогнозує успішність і утримання студентів.
- Використовує дані з системи управління навчанням закладу.
- Надає раннє сповіщення та рекомендації щодо втручання.
- Призначений для покращення результатів і залучення студентів.

2.2 Постановка завдання

Проведений аналіз дозволив визначити основні методи та ресурси, які необхідні для вирішення проблеми прогнозного моделювання та аналітики успішності студентів. Тепер можна перерахувати коло задач, які необхідно виконати на теоретичному та практичному рівнях в межах кваліфікаційної роботи:

1) Визначити та дослідити набори даних на основі яких будуть тренуватися моделі.

2) Виокремити етапи обробки даних.

3) Розбиття історичних даних за допомогою методики утримування «утримування» - це коли ви розділяєте свій набір даних на набір «навчальний» і «тестовий»;.

4) Покращення параметрів моделі використовуючи попередньо оброблені данні в першому та другому пункті.

5) Використання моделі на абсолютно нових даних.

6) Вимірювання показників ефективності.

7) Розробити серверне API додатку .

8) Розробити веб-інтерфейс додатку.

Щодо веб-сервісу, то він міститиме лише основний функціонал, а саме:

1) Система повинна мати можливість прийняти файл csv з усіма даними.

2) Система повинна дозволяти вибір алгоритмів ML для прогнозування рівня залучення та успішності студентів.

3) Система повинна мати можливість застосовувати обробку даних.

4) Система повинна мати можливість відкидати нерелевантні атрибути в завантаженому наборі даних.

5) Система повинна мати можливість виведення даних про важливість кожного релевантного атрибуту в попередньо завантаженому наборі даних.

6) Система повинна оцінювати точність моделі, прогнозування, запам’ятовування та оцінку F1.

7) Система повинна надавати змогу виведення результати в файл формату csv.

Натомість всі зусилля та час будуть спрямовані на реалізацію алгоритму, який буде чітко виконувати своє основне завдання.

Написання серверного API, який буде відповідати за роботу з даними та реалізація алгоритму машинного навчання буде здійснюватися за допомогою мови програмування Python [21], адже вона містить безліч вбудованих методів та бібліотек, використання яких набагато спрощує процес розробки. Початкова реалізація алгоритму буде здійснюватися в хмарному середовищі розробки Google Colaboratory [22]. Для розробки веб-інтерфейсу та інфраструктурної частини додатку було обрано ASP.NET Core MVC. ASP.NET Core MVC — це потужна структура для створення веб-додатків за допомогою архітектурного шаблону Model-View-Controller (MVC). Це кросплатформна структура з відкритим кодом, розроблена Microsoft. Загалом ASP.NET Core MVC пропонує надійну та гнучку структуру для розробки веб-додатків. Він поєднує потужність шаблону MVC із продуктивністю та крос-платформними можливостями ASP.NET Core, що робить його популярним вибором для створення сучасних, масштабованих і високопродуктивних веб-додатків.

Висновок до розділу 2

В даному розділі було розглянуто три програмні продукти, які використовуються для аналітики та прогнозного моделювання успішності студентів. Кожен з перелічених додатків має свої плюси та недоліки, які було описано вище.

На основі проведеного дослідження та аналізу аналогів було здійснено чітку постановку завдання для проведення розробки власного додатку. Було обрано інструментарій для реалізації алгоритму та обрано інструментарій для розробки веб-інтерфейсу та інфраструктурної частини додатку.

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ

3.1 Характеристика джерел даних для проведення аналізу

Для прогнозної аналітики додатку слід використати прогнозне моделювання за допомогою мови програмування Python.[18]

Для реалізації пропонується використати наступні модулі:

- scikit-learn — це модуль Python для машинного навчання, створений на основі SciPy і поширюється за ліцензією BSD із 3 пунктами.
- pandas — програмна бібліотека, написана для мови програмування Python для маніпулювання даними та їхнього аналізу. Вона, зокрема, пропонує структури даних та операції для маніпулювання чисельними таблицями та часовими рядами.
- Numpy — розширення мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами.

Написання коду відбуватиметься в хмарному середовищі розробки Google Colaboratory [14].

Важливо знати, як правильно орієнтуватися в даних, з якими ви працюєте, щоб знати, як побудувати свою прогнозну модель. З цієї причини Python має кілька функцій, які допоможуть вам у ваших дослідженнях:

- info() – Функція, що показує нам тип даних кожного стовпця, кількість стовпців, використання пам’яті та кількість записів у наборі даних (рис. 2.1);

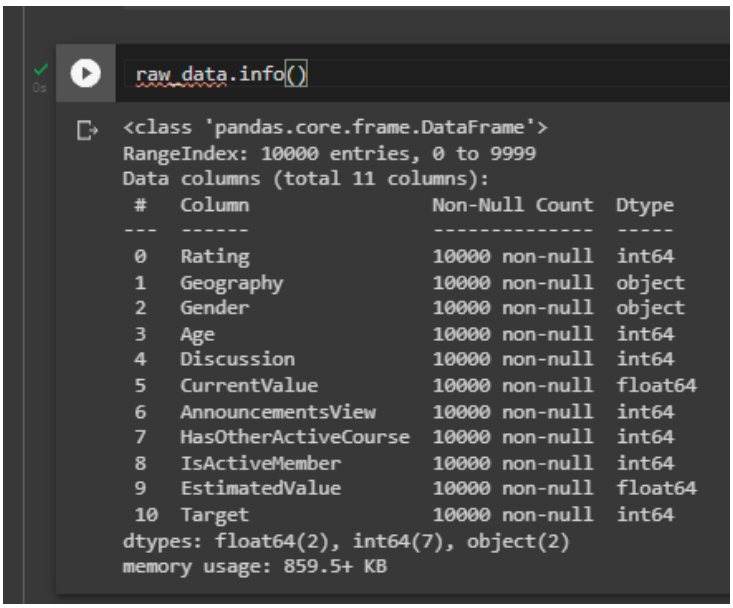


Рис. 3.1 Структура набору даних

- shape – властивість, що відображає кількість записів і стовпців (рис. 2.2);

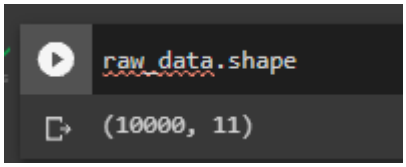
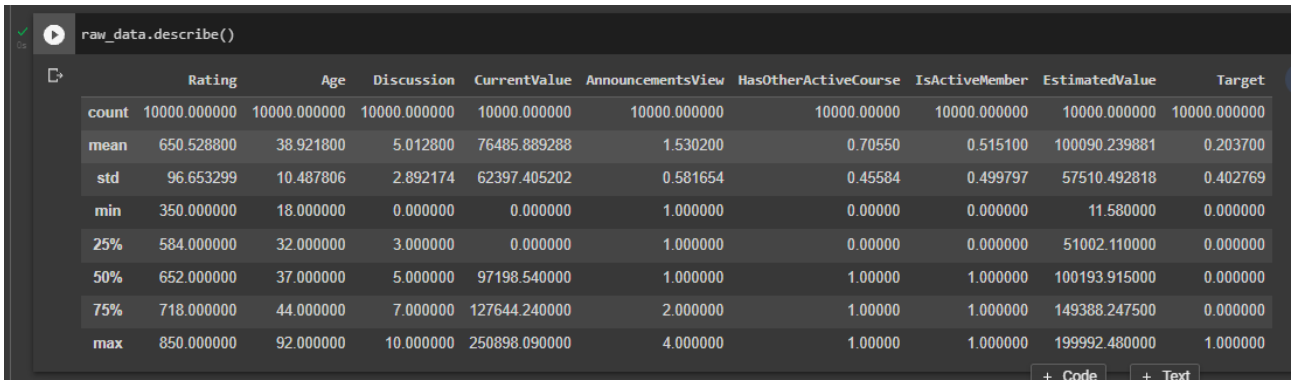


Рис. 3.2 Кількість записів і стовпців

- describe() – функція, що узагальнює статистичні властивості набору даних, такі як кількість, середнє, мінімальне та максимальне (рис. 2.3);

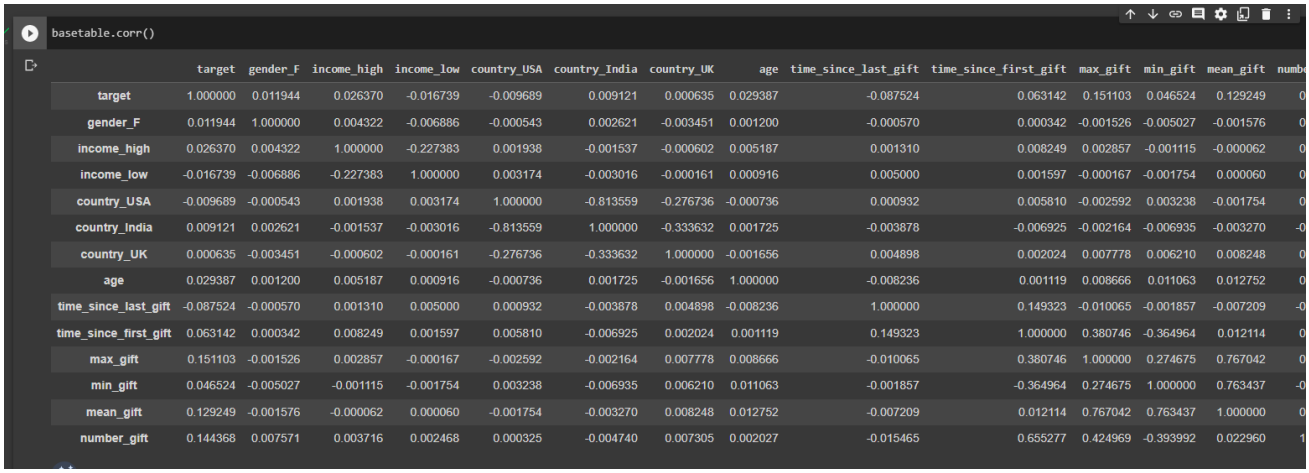


	Rating	Age	Discussion	CurrentValue	AnnouncementsView	HasOtherActiveCourse	IsActiveMember	EstimatedValue	Target
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10000.000000	10000.000000
mean	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881	0.203700
std	96.653299	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57510.492818	0.402769
min	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000	0.000000
25%	584.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000	0.000000
50%	652.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000	0.000000
75%	718.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500	0.000000
max	850.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000	1.000000

Рис. 3.3 Виклик функції describe

- corr() – функція, що відображає кореляцію між різними змінними в нашому наборі даних Чим ближче до 1, тим сильніша кореляція між цими змінними. Знак мінус означає, що ці 2 змінні негативно

корельовані, тобто одна зменшується зі збільшенням іншої і навпаки (рис. 3.4).



```
basetable.corr()
```

	target	gender_F	income_high	income_low	country_USA	country_India	country_UK	age	time_since_last_gift	time_since_first_gift	max_gift	min_gift	mean_gift	numbe
target	1.000000	0.011944	0.026370	-0.016739	-0.009689	0.009121	0.000635	0.029387	-0.087524	0.063142	0.151103	0.046524	0.129249	0
gender_F	0.011944	1.000000	0.004322	-0.006886	-0.000543	0.002621	-0.003451	0.001200	-0.000570	0.000342	-0.001526	-0.005027	-0.001576	0
income_high	0.026370	0.004322	1.000000	-0.227383	0.001938	-0.001537	-0.000602	0.005187	0.001310	0.008249	0.002857	-0.001115	-0.000062	0
income_low	-0.016739	-0.006886	-0.227383	1.000000	0.003174	-0.003016	-0.000161	0.000916	0.005000	0.001597	-0.000167	-0.001754	0.000060	0
country_USA	-0.009689	-0.000543	0.001938	0.003174	1.000000	-0.813559	-0.276736	-0.000736	0.000932	0.005810	-0.002592	0.003238	-0.001754	0
country_India	0.009121	0.002621	-0.001537	-0.003016	-0.813559	1.000000	-0.333632	0.001725	-0.003878	-0.006925	-0.002164	-0.006935	-0.003270	-0
country_UK	0.000635	-0.003451	-0.000602	-0.000161	-0.276736	-0.333632	1.000000	-0.001656	0.004898	0.002024	0.007778	0.006210	0.008248	0
age	0.029387	0.001200	0.005187	0.000916	-0.000736	0.001725	-0.001656	1.000000	-0.008236	0.001119	0.008666	0.011063	0.012752	0
time_since_last_gift	-0.087524	-0.000570	0.001310	0.005000	0.000932	-0.003878	0.004898	-0.008236	1.000000	0.149323	-0.010065	-0.001857	-0.007209	-0
time_since_first_gift	0.063142	0.000342	0.008249	0.001597	0.005810	-0.006925	0.002024	0.001119	0.149323	1.000000	0.380746	-0.364964	0.012114	0
max_gift	0.151103	-0.001526	0.002857	-0.000167	-0.002592	-0.002164	0.007778	0.008666	-0.010065	0.380746	1.000000	0.274675	0.767042	0
min_gift	0.046524	-0.005027	-0.001115	-0.001754	0.003238	-0.006935	0.006210	0.011063	-0.001857	-0.364964	0.274675	1.000000	0.763437	-0
mean_gift	0.129249	-0.001576	-0.000062	0.000060	-0.001754	-0.003270	0.008248	0.012752	-0.007209	0.012114	0.767042	0.763437	1.000000	0
number_gift	0.144368	0.007571	0.003716	0.002468	0.000325	-0.004740	0.007305	0.002027	-0.015465	0.655277	0.424969	-0.393992	0.022960	1

Рис. 3.4 Кореляція змінних в наборі даних

3.2 Побудова алгоритму машинного навчання та його практична реалізація

Почнемо з попередньої обробки даних на цьому етапі нам потрібно:

1. Переглянути кількість значень для кожного атрибуту в нашому наборі даних (рис. 3.5).

2. Перетворення категоріальних змінних на фіктивні/індикаторні змінні, тому що описані вище моделі працюють на основі математичних моделей, яким потрібні числові значення. Таким чином властивість набору даних «Gender» яка мала лише два значення Female, Male, автоматично трансформується в дві властивості: Gender_Female, Gender_Male.

Властивість набору «Geography» трансформується в три властивості: Geography_France, Geography_Germany, Geography_Spain.

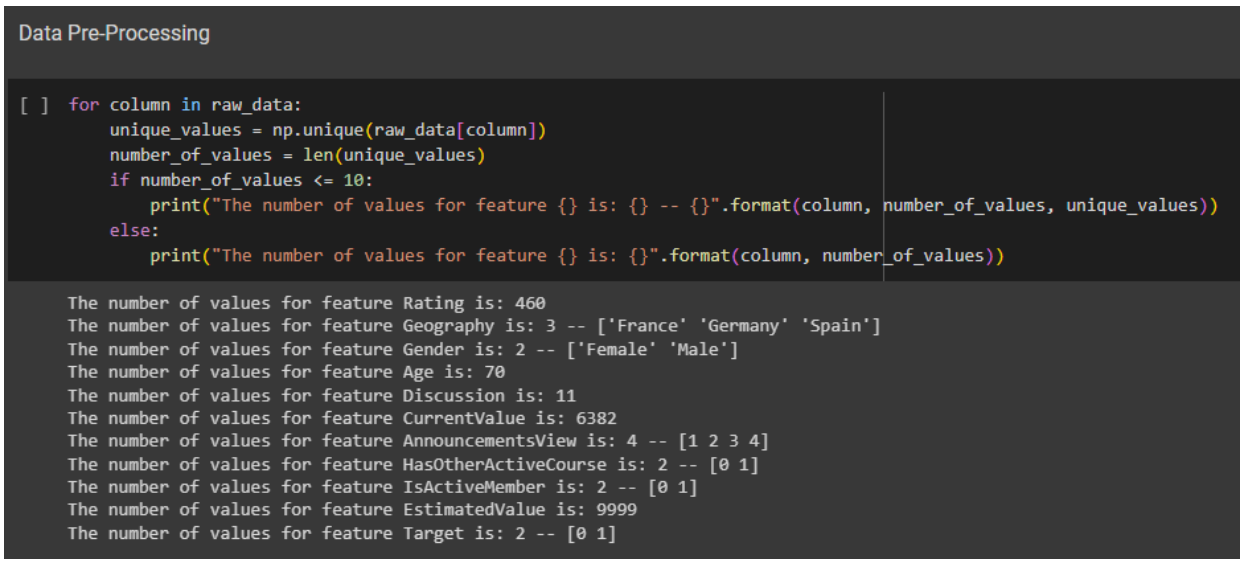


Рис. 3.5 Огляд властивостей дата сету

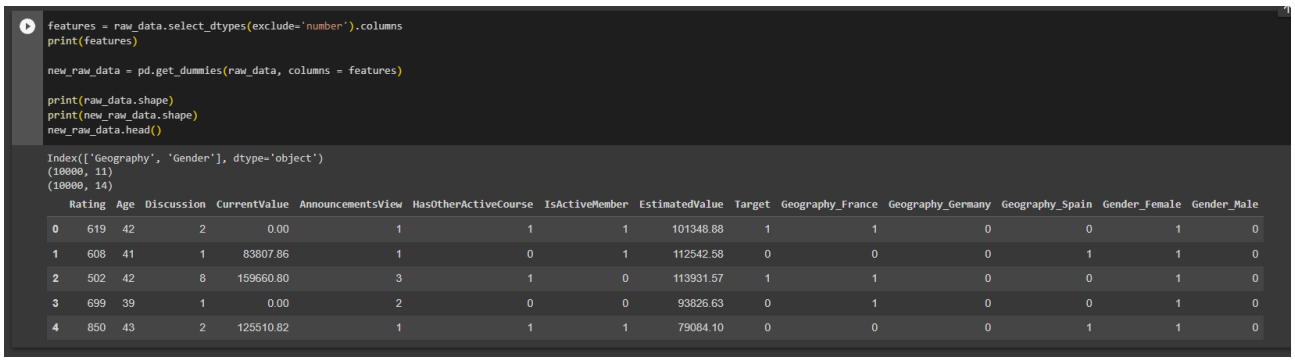


Рис. 3.6 Перетворення категоріальних змінних на фіктивні/індикаторні змінні

3. Масштабування числових властивостей за допомогою MinMaxScaler для підвищення швидкості роботи моделей та їх точності (рис. 2.3). Дві найпопулярніші техніки для масштабування числових даних перед моделюванням – нормалізація та стандартизація. Нормалізація масштабує кожну вхідну змінну окремо до діапазону 0-1, який є діапазоном для значень з плаваючою комою, де ми маємо найбільшу точність. Стандартизація масштабує кожну вхідну змінну окремо шляхом віднімання середнього (це називається центруванням) і ділення на стандартне відхилення, щоб зсунути розподіл, щоб отримати середнє значення нуль і стандартне відхилення одиниці.

```
scale_variables = raw_data.select_dtypes(include='number').columns
print(scale_variables)

scaler = MinMaxScaler()
new_raw_data[scale_variables] = scaler.fit_transform(new_raw_data[scale_variables])
```

```
Index(['Rating', 'Age', 'Discussion', 'CurrentValue', 'AnnouncementsView',
       'HasOtherActiveCourse', 'IsActiveMember', 'EstimatedValue', 'Target'],
      dtype='object')
```

Рис. 3.7 Масштабування числових даних

Наступний етап це розбиття даних за допомогою методики «утримування» - це коли ви розділяєте свій набір даних на набір «навчальний» і «тестовий». Навчальний набір — це те, на чому модель навчається, а тестовий набір використовується, щоб побачити, наскільки добре ця модель працює на нових даних. Звичайним поділом під час використання методу очікування є використання 90% даних для навчання, а решта 10% даних для тестування (рис. 3.8).

Splitting the Historical Data - Hold-out validation

```
X = new_raw_data.drop('target', axis='columns').values
y = new_raw_data['target'].values
print('X shape: {}'.format(np.shape(X)))
print('y shape: {}'.format(np.shape(y)))

X_train, X_test, y_train, y_test = train_test_split(X, y, train_size = 0.9, test_size=0.1, random_state=0)
```

```
X shape: (649, 56)
y shape: (649,)
```

Рис. 3.8 Розбиття даних

Після цього отриманні данні можна використовувати в моделях, на наступному рисунку зображено побудову дерева рішень (рис. 3.9).

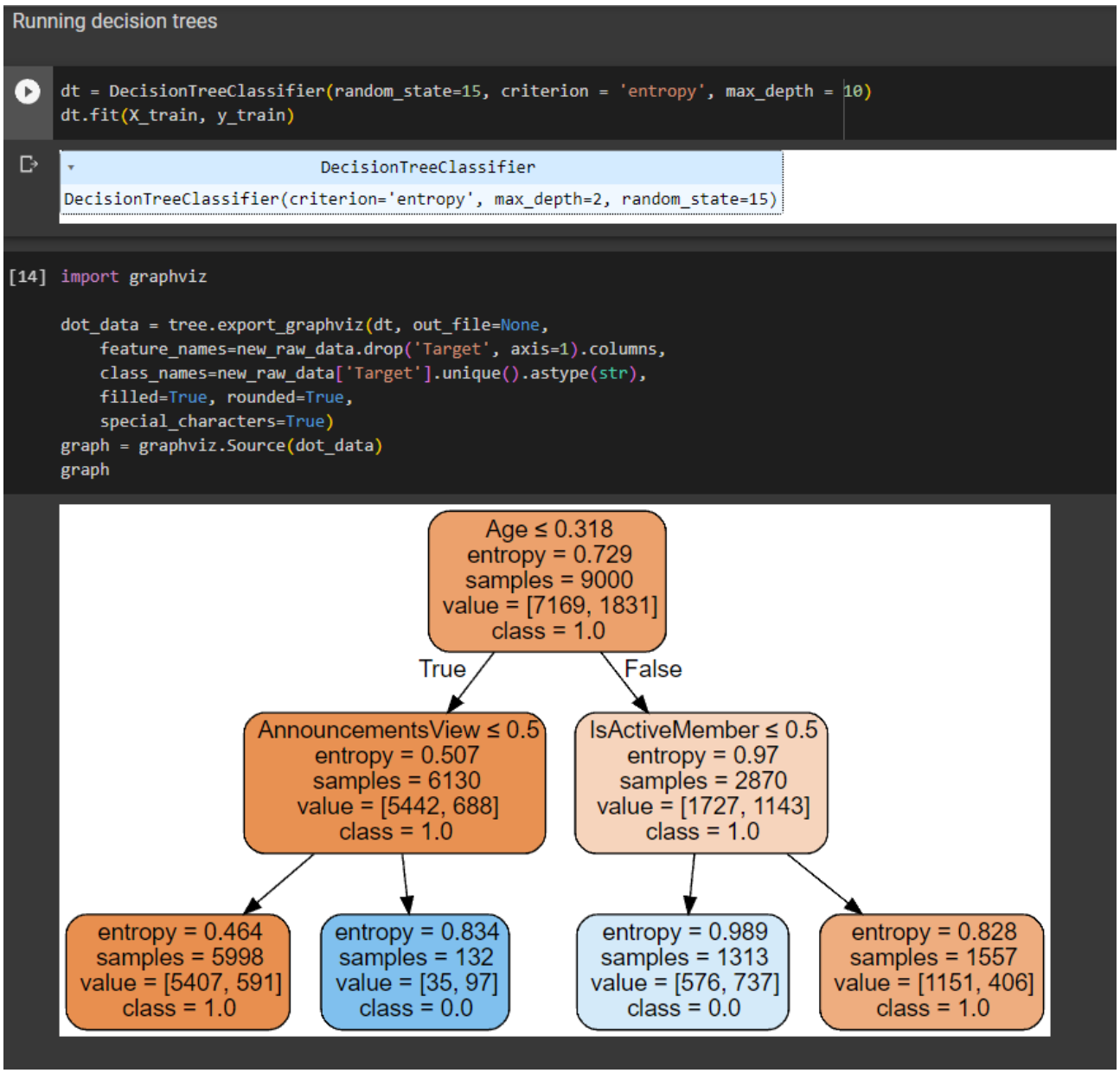


Рис. 3.9 Побудова дерева рішень.

Яке в свою чергу можна використати для знаходження найбільш релевантних властивостей набору даних. Наприклад, якщо перед нами стоїть мета покращити результативність лінійної регресії чи отримати показники властивостей які найбільше впливають на кінцевий результат DecisionTreeClassifier має повністю справитися з цією задачею. Дерево рішень отримує знання у формі дерева, яке також можна переписати як набір окремих правил, щоб полегшити його розуміння. Основною перевагою класифікатора дерева рішень є його здатність використовувати різні підмножини ознак і правила прийняття рішень на різних етапах класифікації. Як показано на рисунку

2.5, загальне дерево рішень складається з одного кореневого вузла, ряду внутрішніх і кінцевих вузлів і гілок. Листові вузли вказують на клас, який буде присвоєно зразку. Кожен внутрішній вузол дерева відповідає ознаці, а гілки представляють об'єднання ознак, які призводять до цих класифікацій. Як ми бачимо з рисунку 2.6 кожна властивість має значення важливості яке відмінне від нуля, тому нам нічого не потрібно видаляти з набору даних.

```
fi = pd.DataFrame(columns = ['Variable', 'Feature Importance Score'])
for i, column in enumerate(new_raw_data.drop('Target', axis=1)):
    print('Importance of {}: {:.3f}'.format(column, dt.feature_importances_[i]))
    fi = fi.append({'Variable': column, 'Feature Importance Score': dt.feature_importances_[i]},  ignore_index = True)

fi.sort_values('Feature Importance Score', ascending = False).reset_index()
print(fi)
```

```
Importance of Rating: 0.090
Importance of Age: 0.318
Importance of Discussion: 0.030
Importance of CurrentValue: 0.125
Importance of AnnouncementsView: 0.217
Importance of HasOtherActiveCourse: 0.004
Importance of IsActiveMember: 0.068
Importance of EstimatedValue: 0.091
Importance of Geography_France: 0.002
Importance of Geography_Germany: 0.032
Importance of Geography_Spain: 0.004
Importance of Gender_Female: 0.014
Importance of Gender_Male: 0.005
                Variable  Feature Importance Score
0                 Rating                  0.090341
1                    Age                  0.317576
2             Discussion                  0.029849
3          CurrentValue                   0.124537
4      AnnouncementsView                  0.217418
5   HasOtherActiveCourse                  0.003787
6         IsActiveMember                  0.067886
7         EstimatedValue                  0.090899
8       Geography_France                  0.001891
9      Geography_Germany                  0.032289
10       Geography_Spain                  0.003758
11         Gender_Female                  0.014356
12           Gender_Male                  0.005415
```

Рис. 3.10 Знаходження найбільш важливих властивостей набору даних

Розглянемо використання моделі RandomForest та покращення її результатів шляхом підбору параметрів для роботи з набором даних (рис 3.11), також побудуємо матрицю помилок. У сфері машинного навчання та, зокрема, у проблемі статистичної класифікації, матриця плутанини, також відома як матриця помилок [10], є певним макетом таблиці, який дозволяє візуалізувати продуктивність алгоритму, як правило, навчання під наглядом (у неконтрольоване навчання зазвичай називають матрицею відповідності). Кожен рядок матриці представляє екземпляри в фактичному класі, тоді як кожен стовпець представляє екземпляри в прогнозованому класі, або навпаки – обидва варіанти зустрічаються в літературі. Назва походить від того факту, що вона

дозволяє легко побачити, чи система плутає два класи (тобто, як правило, неправильно позначає один як інший).



Рис. 3.11 Використання моделі RandomForest на розбитих даних за допомогою методики утримування.

Судячи з отриманої матриці (1 - успішне завершення курсу, 0 – студент не зміг завершити курс успішно), можна встановити, що ми отримали досить непогані результати класифікації, адже коефіцієнт правильного встановлення того що студент не завершить курс 0.96, а коефіцієнт правильності встановлення успішного завершення становить 0.47 .

Наступний етап полягає в виборі найкращих параметрів моделі при використанні з навчальним набором даних, лише після цього можемо добитися

більш кращого результату виконання прогнозного аналізу на абсолютно новому наборі даних (рис. 3.13, рис.3.14).

```
from itertools import product
n_estimators = 100
max_features = [1, 'sqrt', 'log2']
max_depths = [None, 2, 3, 4, 5]
for f, d in product(max_features, max_depths):
    rf = RandomForestClassifier(n_estimators=n_estimators,
                                criterion='entropy',
                                max_features=f,
                                max_depth=d,
                                n_jobs=2,
                                random_state=1337)
    rf.fit(X_train, y_train)
    prediction_test = rf.predict(X=X_test)
    print('Classification accuracy on test set with max features = {} and max_depth = {}: {:.3f}'.format(f, d, accuracy_score(y_test,prediction_test)))
```

```
Classification accuracy on test set with max features = 1 and max_depth = None: 0.859
Classification accuracy on test set with max features = 1 and max_depth = 2: 0.794
Classification accuracy on test set with max features = 1 and max_depth = 3: 0.794
Classification accuracy on test set with max features = 1 and max_depth = 4: 0.796
Classification accuracy on test set with max features = 1 and max_depth = 5: 0.806
Classification accuracy on test set with max features = sqrt and max_depth = None: 0.864
Classification accuracy on test set with max features = sqrt and max_depth = 2: 0.803
Classification accuracy on test set with max features = sqrt and max_depth = 3: 0.812
Classification accuracy on test set with max features = sqrt and max_depth = 4: 0.838
Classification accuracy on test set with max features = sqrt and max_depth = 5: 0.852
Classification accuracy on test set with max features = log2 and max_depth = None: 0.864
Classification accuracy on test set with max features = log2 and max_depth = 2: 0.803
Classification accuracy on test set with max features = log2 and max_depth = 3: 0.812
Classification accuracy on test set with max features = log2 and max_depth = 4: 0.838
Classification accuracy on test set with max features = log2 and max_depth = 5: 0.852
```

Рис. 3.12 Підбір параметрів моделі RandomForest

```
classifier = xgboost.XGBClassifier(tree_method='gpu_hist')

params = {
    "learning_rate":[0.05,0.10,0.15,0.20,0.25,0.30],
    "max_depth":[2,3,4,5,6,8,10,12,15],
    "min_child_weight":[1,3,5,7],
    "gamma":[0.0,0.1,0.2,0.3,0.4],
    "colsample_bytree":[0.3,0.4,0.5,0.7]}

searchCV = RandomizedSearchCV(classifier,param_distributions=params,n_iter=5,scoring='roc_auc',cv=5,verbose=3)

searchCV.fit(X,y)
```

```
Fitting 5 folds for each of 5 candidates, totalling 25 fits
[CV 1/5] END colsample_bytree=0.3, gamma=0.2, learning_rate=0.2, max_depth=10, min_child_weight=5;, score=0.849 total time=   2.6s
[CV 2/5] END colsample_bytree=0.3, gamma=0.2, learning_rate=0.2, max_depth=10, min_child_weight=5;, score=0.854 total time=   1.3s
[CV 3/5] END colsample_bytree=0.3, gamma=0.2, learning_rate=0.2, max_depth=10, min_child_weight=5;, score=0.853 total time=   1.2s
[CV 4/5] END colsample_bytree=0.3, gamma=0.2, learning_rate=0.2, max_depth=10, min_child_weight=5;, score=0.854 total time=   1.1s
[CV 5/5] END colsample_bytree=0.3, gamma=0.2, learning_rate=0.2, max_depth=10, min_child_weight=5;, score=0.845 total time=   1.1s
[CV 1/5] END colsample_bytree=0.3, gamma=0.1, learning_rate=0.3, max_depth=3, min_child_weight=3;, score=0.868 total time=   0.6s
[CV 2/5] END colsample_bytree=0.3, gamma=0.1, learning_rate=0.3, max_depth=3, min_child_weight=3;, score=0.863 total time=   0.6s
[CV 3/5] END colsample_bytree=0.3, gamma=0.1, learning_rate=0.3, max_depth=3, min_child_weight=3;, score=0.860 total time=   0.6s
[CV 4/5] END colsample_bytree=0.3, gamma=0.1, learning_rate=0.3, max_depth=3, min_child_weight=3;, score=0.867 total time=   0.5s
[CV 5/5] END colsample_bytree=0.3, gamma=0.1, learning_rate=0.3, max_depth=3, min_child_weight=3;, score=0.861 total time=   0.5s
[CV 1/5] END colsample_bytree=0.3, gamma=0.0, learning_rate=0.3, max_depth=10, min_child_weight=7;, score=0.848 total time=   0.7s
[CV 2/5] END colsample_bytree=0.3, gamma=0.0, learning_rate=0.3, max_depth=10, min_child_weight=7;, score=0.845 total time=   0.7s
[CV 3/5] END colsample_bytree=0.3, gamma=0.0, learning_rate=0.3, max_depth=10, min_child_weight=7;, score=0.845 total time=   0.7s
[CV 4/5] END colsample_bytree=0.3, gamma=0.0, learning_rate=0.3, max_depth=10, min_child_weight=7;, score=0.853 total time=   0.7s
[CV 5/5] END colsample_bytree=0.3, gamma=0.0, learning_rate=0.3, max_depth=10, min_child_weight=7;, score=0.839 total time=   0.7s
[CV 1/5] END colsample_bytree=0.4, gamma=0.0, learning_rate=0.25, max_depth=10, min_child_weight=5;, score=0.838 total time=   0.9s
[CV 2/5] END colsample_bytree=0.4, gamma=0.0, learning_rate=0.25, max_depth=10, min_child_weight=5;, score=0.845 total time=   0.9s
[CV 3/5] END colsample_bytree=0.4, gamma=0.0, learning_rate=0.25, max_depth=10, min_child_weight=5;, score=0.844 total time=   0.9s
[CV 4/5] END colsample_bytree=0.4, gamma=0.0, learning_rate=0.25, max_depth=10, min_child_weight=5;, score=0.843 total time=   0.9s
[CV 5/5] END colsample_bytree=0.4, gamma=0.0, learning_rate=0.25, max_depth=10, min_child_weight=5;, score=0.832 total time=   0.9s
[CV 1/5] END colsample_bytree=0.3, gamma=0.4, learning_rate=0.1, max_depth=10, min_child_weight=1;, score=0.858 total time=   1.0s
[CV 2/5] END colsample_bytree=0.3, gamma=0.4, learning_rate=0.1, max_depth=10, min_child_weight=1;, score=0.851 total time=   1.0s
[CV 3/5] END colsample_bytree=0.3, gamma=0.4, learning_rate=0.1, max_depth=10, min_child_weight=1;, score=0.856 total time=   1.0s
[CV 4/5] END colsample_bytree=0.3, gamma=0.4, learning_rate=0.1, max_depth=10, min_child_weight=1;, score=0.856 total time=   1.1s
[CV 5/5] END colsample_bytree=0.3, gamma=0.4, learning_rate=0.1, max_depth=10, min_child_weight=1;, score=0.845 total time=   1.0s
RandomizedSearchCV(cv=5, estimator=XGBClassifier(tree_method='gpu_hist'),
                   n_iter=5,
                   param_distributions={'colsample_bytree': [0.3, 0.4, 0.5,
                                                             0.7],
                                        'gamma': [0.0, 0.1, 0.2, 0.3, 0.4],
                                        'learning_rate': [0.05, 0.1, 0.15, 0.2,
                                                          0.25, 0.3],
                                        'max_depth': [2, 3, 4, 5, 6, 8, 10, 12,
                                                      15],
                                        'min_child_weight': [1, 3, 5, 7]},
                   scoring='roc_auc', verbose=3)
```

```
searchCV.best_params_
```

```
{'min_child_weight': 3,
 'max_depth': 3,
 'learning_rate': 0.3,
 'gamma': 0.1,
 'colsample_bytree': 0.3}
```

```
searchCV.best_estimator
```

Рис. 3.13 Підбір параметрів моделі XGBoost

Наступний крок, це використання моделі на абсолютно новому наборі даних, для визначення того чи завершить студент курс та з якою імовірністю (рис. 2.10).

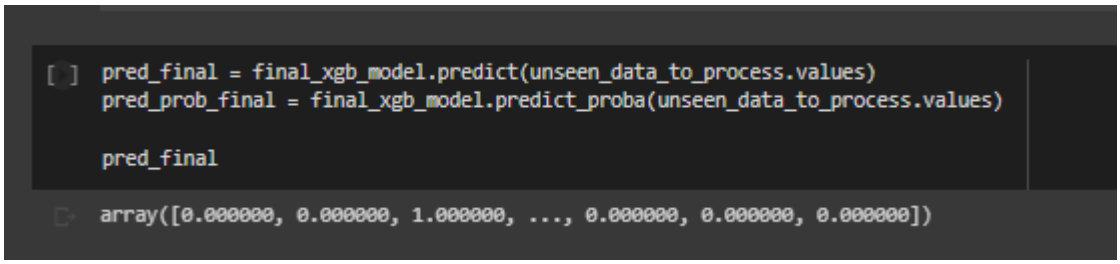


Рис. 3.14 Попередня обробка нових даних та використання фінальної моделі з обраними параметрами для прогнозування.

Отримані результати прогнозування збережемо в окремі властивості набору даних, які будуть розглянуті трохи пізніше.

3.3 Аналіз результатів та визначення метрик для оцінки моделей

Точність моделі є основним індикативним фактором для оцінки моделі. Результати, отримані за алгоритмом, були представлені відповідно до найвищої спостережуваної точності. Для оцінки моделі були використані такі характеристики:

- Точність (Accuracy) є найосновнішим показником продуктивності, який відображає загальну правильність передбачень моделі. Він обчислюється шляхом ділення кількості правильно передбачених випадків на загальну кількість випадків. Однак точність може ввести в оману, якщо набір даних незбалансований, тобто один клас має значно більше екземплярів, ніж інші.
- Презиція (Precision) — вимірює частку правильно передбачених позитивних випадків серед усіх випадків, передбачених як позитивні. Він акцентує увагу на правильності позитивних прогнозів. Точність обчислюється шляхом ділення кількості істинних позитивних результатів (правильно передбачених позитивних випадків) на суму

істиних позитивних і хибних позитивних результатів (неправильно передбачених позитивних випадків).

- Запам’ятовування (Recall) — також відоме як чутливість (Sensitivity) або справжній позитивний показник, вимірює частку правильно передбачених позитивних випадків серед усіх фактичних позитивних випадків. Він зосереджується на здатності моделі визначати позитивні випадки. Відкликання обчислюється діленням кількості істинно позитивних результатів на суму істинно позитивних результатів і хибно-негативних результатів (позитивні випадки, неправильно передбачені як негативні).
- Помилка класифікації — також відома як помилка неправильної класифікації або частота помилок, є показником, який використовується для оцінки ефективності моделі класифікації. Він кількісно визначає ступінь, до якого модель неправильно класифікує спостереження або робить неправильні прогнози. У контексті машинного навчання помилка класифікації є важливим показником для оцінки точності та надійності класифікатора.
- F-measure (F1) — це збалансована метрика, яка поєднує точність і запам’ятовування в один єдиний показник. Він забезпечує гармонійне середнє значення точності та запам’ятовування, надаючи однакову важливість обом показникам. F1-оцінка широко використовується, коли існує нерівномірний розподіл класів у наборі даних. Він розраховується як середньозважене значення точності та запам’ятовування за формулою:

$$F1 = 2 * (precision * recall)/(precision + recal)$$

Наступим кроком для більш кращого розуміння роботи нашої моделі є побудова матриці помилок, що дасть нам більш детальні характеристик того наскільки точним було наше прогнозування. Матриця помилок зазвичай використовується для оцінки ефективності методів, використаних після

класифікації. Переглянемо матриці помилок після покращення параметрів моделей (1 - успішне завершення курсу, 0 – студент не зміг завершити курс успішно).

Матриця плутанини, також відома як матриця помилок, — це таблиця, яка підсумовує продуктивність моделі класифікації. Це особливо корисно під час оцінювання моделей, які класифікують дані на два класи. Матриця плутанини дає уявлення про кількість істинно позитивних, істинно негативних, хибно-позитивних і хибно-негативних результатів, створених моделлю.

Ось розбивка компонентів у матриці плутанини для задачі бінарної класифікації з двома класами, які часто позначаються як «позитивний» і «негативний»:

- Істинно позитивні результати (TP - True Positives): це кількість випадків, які модель правильно класифікувала як позитивні.
- Істинно негативні результати (TN - True Negatives): Це вказує на кількість екземплярів, які модель правильно класифікувала як негативні.
- Хибно-позитивні спрацьовування (False FP): також відомі як помилки типу I, це випадки, які модель неправильно класифікувала як позитивні, хоча насправді вони належать до негативного класу.
- Хибно-негативні (FN): також звані помилками типу II, це випадки, які були неправильно класифіковані як негативні моделлю, хоча вони насправді належать до позитивного класу.

Матриця помилок пропонує стислий підсумок результатів класифікації моделі, що дає нам змогу оцінити її ефективність і визначити потенційні області для вдосконалення.

```python
def evaluateMetrics(cm):
  FP = cm.sum(axis=0) - np.diag(cm)
  FN = cm.sum(axis=1) - np.diag(cm)
  TP = np.diag(cm)
  TN = cm.sum() - (FP + FN + TP)
  recal = TP/(TP+FN)
  print ("The Recall (Sensitivity) per class is: ", recal)
  print ("The average Recall (Sensitivity) is: ", recal.sum()/2)
  print("")

  accuracy = (TP+TN)/(TP+FP+FN+TN)
  print ("The Accuracy of each class is", accuracy)
  print ("The average Accuracy is: ",accuracy.sum()/2)
  print("")

  precision = TP/(TP+FP)
  print ("The Precision per class is: ",precision)
  print ("The average Precision is: ", precision.sum()/2)
  print("")

  clasification_error = (FP+FN)/(TP+FP+FN+TN)
  print ("The Classification error of each class is", clasification_error)
  print ("The average Classification error is", clasification_error.sum()/2)
```

Рис. 3.15 Визначення метрик ефективності моделей

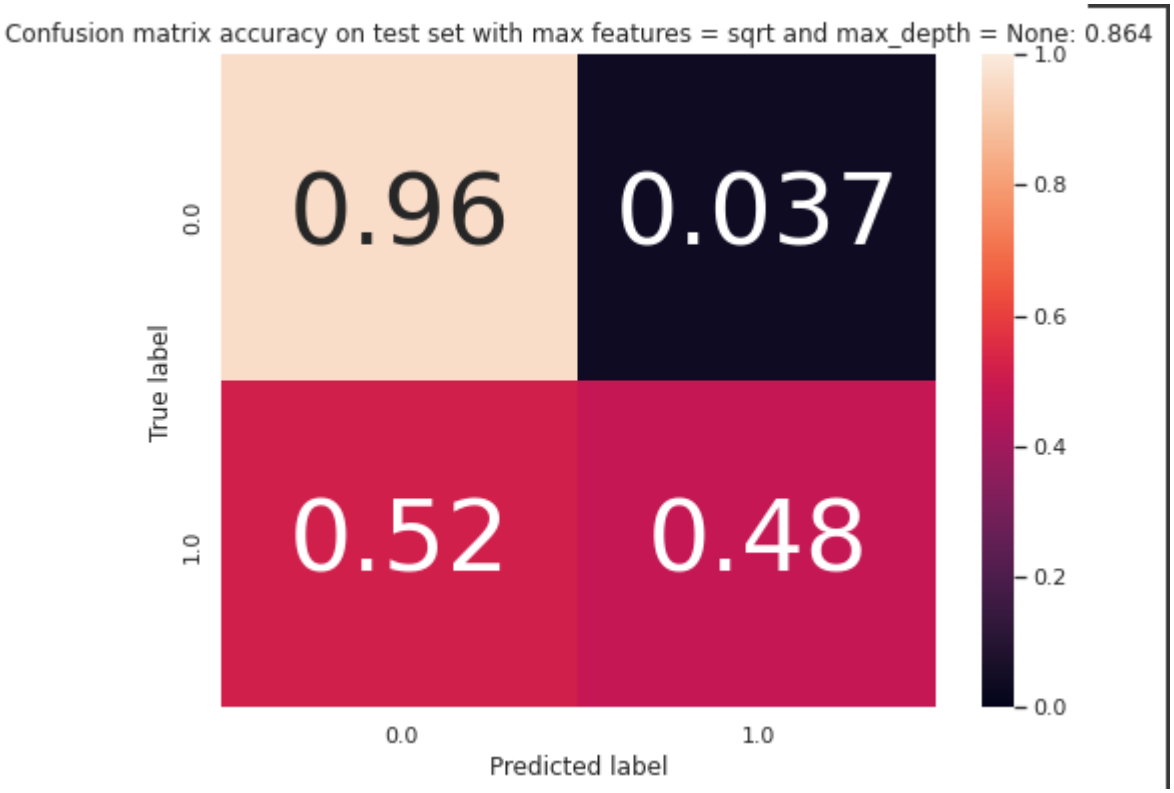


Рис. 3.16 Матриця помилок для RandomForest після покращення параметрів

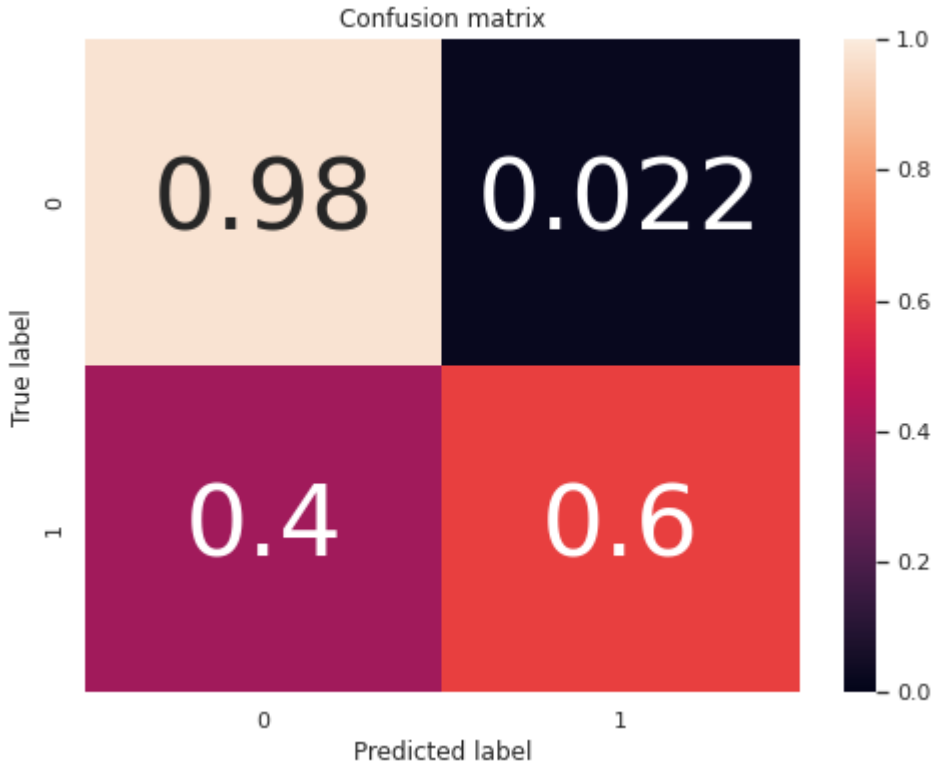


Рис. 3.17 Матриця помилок для XGBoost після покращення параметрів

Таблиця 3.1 Опис характеристик моделей

Характеристика	RandomForest	XGBoost
Точність	85%	90%
Чутливість	65%	78%
Правильність	87%	89%
Помилкова класифікація	14%	9%

Спираючись на отримані метрики можна зробити висновок що XGBoost набагато краще справляється з проблемою прогнозної класифікації, але також не можна відкидати модель RandomForest для даної задачі, тому що вона також показує гарні результати.

Отримані данні в свою чергу викладачі можуть використовувати для роботи зі студентами опираючись на результати роботи моделі, наприклад після класифікаційної обробки ми отримали відповідь на те чи завершить студент курс та з якою ймовірністю це відбудеться (рис. 3.18).

Predictions - Succeed or Not	Predictions - Probability to Succeed education	Predictions - Succeed or Fail Desc
1.0	0.56	Succeed
0.0	0.27	Fail
1.0	0.96	Succeed
0.0	0.02	Fail
0.0	0.10	Fail

Рис. 3.18 Результати прогнозного аналізу використовуючи модель з обраними попередньо параметрами.

Висновки до розділу 3

У другому розділі звіту, визначено інструментарій для реалізації алгоритму, проведено здійснено практичну реалізацію алгоритму. В даному розділі було проведено аналіз метрик та результатів натренованої моделі. З матриці помилок можна побачити, що ми отримали досить непогані результати після підбору параметрів для моделей.

В основному моделі показали досить гарні результати. Також проведено приклад використання отриманих даних.

РОЗДІЛ 4. КОМПОНУВАННЯ АЛГОРИТМУ МАШИННОГО НАВЧАННЯ, НАПИСАННЯ СЕРВЕРНОГО API ТА ВЕБ-ІНТЕРФЕЙСУ

4.1 Розробка серверного API для алгоритму машинного навчання

Після побудови алгоритму, було прийнято рішення винести його реалізацію в окремий сервіс, за допомогою Python та бібліотеки Flask написано спеціальне API для їх його використання. Flask — це легкий і популярний веб-фреймворк, написаний на Python. Це дозволяє розробникам швидко та легко створювати веб-додатки. Flask дотримується підходу мікрофреймворку, забезпечуючи мінімалістичну, але розширювану основу для веб-розробки. Розроблений Armin Ronacher, Flask був випущений у 2010 році і з тих пір отримав широке поширення в спільноті Python. На рисунку 4.1 зображено головні ендпоінти. Посилання на весь код серверної частини міститься у Додатку А.

```python
@app.route('/regression', methods=['POST'])
def execute_request():
    modelType = request.json['type']
    test_file_path = request.json['testFilePath']
    source_file_path = request.json['sourceFilePath']
    target_field = request.json['targetField']
    request_id = request.json['requestId']

    try:
        if modelType == "XGBoost":
            helper = XGBoostHelper(
                test_file_path, source_file_path, target_field, TRAIN_SIZE, TEST_SIZE, request_id)
            return json.dumps(helper.startProcess())
        elif modelType == "RandomForest":
            helper = RandomForestHelper(
                test_file_path, source_file_path, target_field, TRAIN_SIZE, TEST_SIZE, request_id)
            return json.dumps(helper.startProcess())
        elif modelType == "LogisticRegression":
            helper = LogisticRegressionHelper(
                test_file_path, source_file_path, target_field, TRAIN_SIZE, TEST_SIZE, request_id)
            return json.dumps(helper.startProcess())

        return jsonify({'error': 'model type is not support'}), 500
    except Exception as e:
        return jsonify({'error': str(e)}), 500


@app.route("/loadModel", methods=['POST'])
def execute_request_with_existing():
    predictionId = request.json['predictionId']
    source_file_path = request.json['sourceFilePath']
    columns = request.json['columns']
    modelType = request.json['type']
```

Рис. 4.1. Ендроінти API

Перший ендпоінт буде використовуватися для навчання та використання їх на нових даних (рис. 4.2), а другий надає доступ до попередньо натренованих моделей, за допомогою яких можна здійснити здійснити прогнозний аналіз використовуючи нові дані в тому самому форматі що й історичні дані, які були використані для навчання моделі.

Розглянемо більш детально параметри, які приймає перший ендпоінт серверного API:

- type – тип моделі для навчання. Можливі значення: «LogisticRegression», «RandomForest», «XGBoost»;
- testFilePath – шлях до тестового файлу, або до файлу з історичними даними для навчання моделі;
- sourceFilePath – шлях до файлу з абсолютно новими даними для використання моделі;
- targetField – назва поля з файлу з історичного набору даних для навчання моделі;
- requestId – унікальний id запиту у формі unique identifier, який генерується на стороні користувача API.

Після відправки запиту відбувається навчання моделі на основі отриманого набору даних, а потім використовує абсолютно новий набір даних для виконання прогнозного аналізу. В результаті користувач API отримує наступну відповідь:

- метрики для оцінки виконання моделі, такі поля як: “accuracy”, “precision”, “clasification_error”, “recal”;
- масив з інформацією про важливість кожного релевантного значення атрибутів в попередньо завантаженому наборі даних;
- шлях до файлу з прогнозним набором даних;
- масив з переліком релевантних атрибутів в попередньо завантаженому наборі даних.

Також важливо зазначити, що після навчання натренована модель зберігається на диску серверу для повторного використання. Для збереження та завантадення в пам’ять натренованої моделі XGBoost використовується внутрішнє API пекеджу “xgboost”, а для моделей LogisticRegression та Random Forest використовується бібліотека joblib. Joblib — це бібліотека Python, яка забезпечує зручний спосіб збереження та завантаження навчених моделей машинного навчання. Він пропонує простий і ефективний підхід до збереження навчених моделей, що дозволяє повторно використовувати їх пізніше без необхідності повторного навчання.

```
joblib.dump(final_model, os.path.join(
    directory_path, Constants.SKLEARN_MODEL_NAME))
```

Рис. 4.2 Збереження моделі

Функція збереження Joblib особливо корисна в ситуаціях, коли ви навчили модель, для навчання якої потрібен значний проміжок часу або велика кількість навчальних даних. Зберігаючи модель на диску, ви можете уникнути необхідності повторювати процес навчання та швидко отримувати доступ до навченої моделі, коли вам це потрібно.

```
model = joblib.load(self.__get_model_path())
pred_final = model.predict(X)
pred_prob_final = model.predict_proba(X)
```

Рис. 4.3 Приклад завантаження моделі з пам’яті

Розглянемо більш детально параметри, які приймає другий ендпоінт серверного API:

- type – тип моделі для навчання. Можливі значення: «LogisticRegression», «RandomForest», «XGBoost»;
- sourceFilePath – шлях до файлу з абсолютно новими даними для використання моделі;
- columns – масив з переліком релевантних атрибутів в історичному наборі даних, який було використано для навчання моделі;

- predictionId – унікальний ідентифікатор який було згенеровано під час навчання моделі, використовується системою для правильного визначення моделі яку потрібно завантажити в пам’ять.

Після отримання запиту на використання попередньо натренованої моделі, на сервері відбувається завантаження моделі безпосередньо в пам’ять для прогнозного аналізу з використанням абсолютно нового набору даних. Після виконання алгоритму в відповідь користувач API отримує шлях до файлу з прогнозним набором даних, який попередньо було збережено на сервері.

В отриманому файлі з прогнозним набором даних, в свою чергу викладачі можуть використовувати для роботи зі студентами опираючись на результати роботи моделі, наприклад після класифікаційної обробки ми отримали відповідь на те чи завершить студент курс та з якою ймовірністю це відбудеться (рис. 4.2).

AG	AH
Predictions - Probability to Succeed e	Predictions - Succeed or F
1	Succeed
0.987375	Succeed
0.748	Succeed
0.977458333	Succeed
0.976041667	Succeed
0.997990196	Succeed
0.975125	Succeed
0.975652546	Succeed
0.950184524	Succeed
0.946058824	Succeed
0.978362573	Succeed
0.98	Succeed
0.986850013	Succeed
0.99	Succeed
0.94447619	Succeed
0.978666667	Succeed
0.988888889	Succeed

Рис. 4.4 Результати прогнозного аналізу використовуючи модель з обраними попередньо параметрами.

Взаємодію з побудованим API можна підсумувати наступною схемою (рис.4.5).



Рис. 4.5. Схема взаємодії з API

На рисунках 4.6-4.7 зображено тестування серверного API. Всі запити виконувались за допомогою програми Postman.

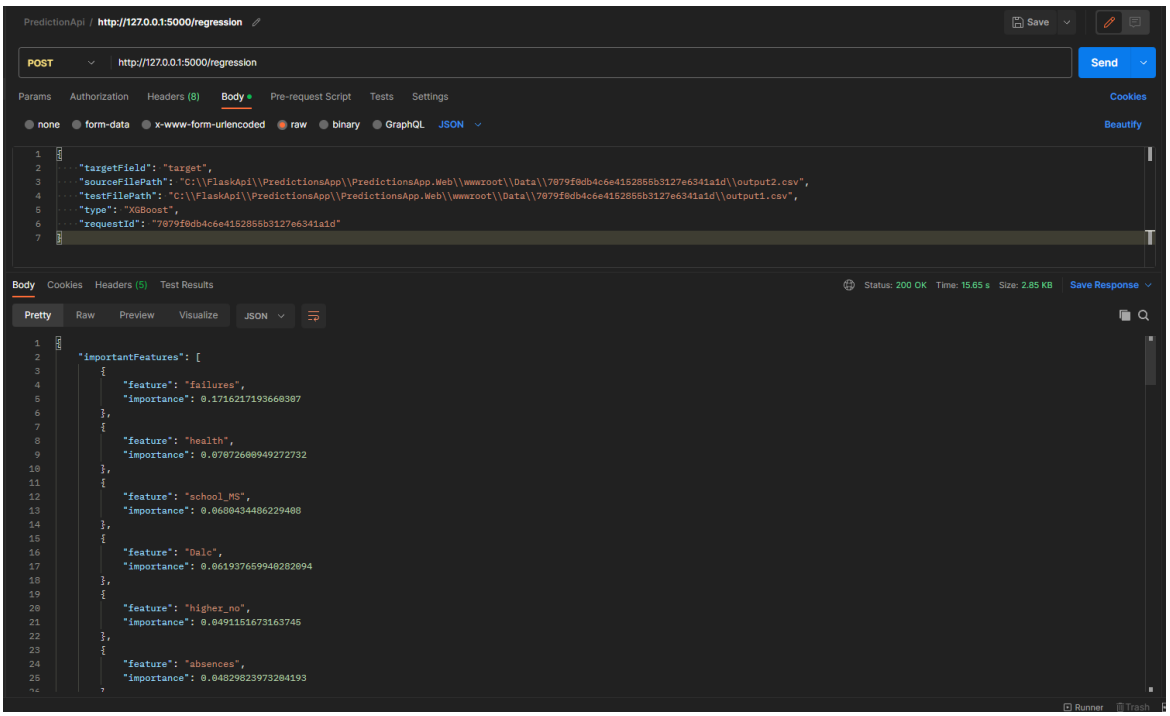


Рис. 4.6 Тестування першого ендпоінту за допомогою програми Postman.

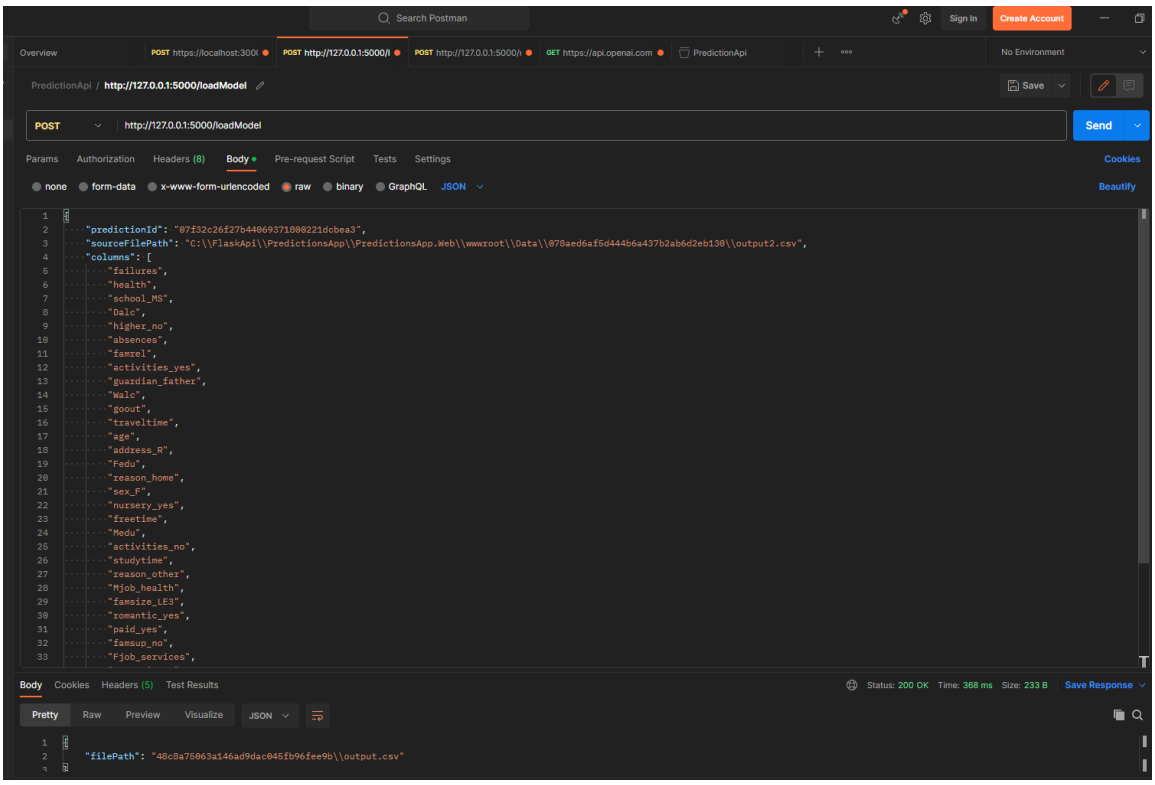


Рис. 4.7 Тестуваня другого ендпоінту за допомогою програми Postman.

Для розробки API було використано наступні інструменти Visual Studio Code – засіб для створення, редагування та зневадження сучасних веб-застосунків і програм для хмарних систем. Visual Studio Code розповсюджується безкоштовно і доступний у версіях для платформ Windows, Linux і OS X. а основу для Visual Studio Code використовуються напрацювання вільного проекту Atom, що розвивається компанією GitHub. VS Code набирає популярності серед спеціалістів із обробки даних та практиків машинного навчання. Він підтримує такі мови, як Python і R, а також розширення для маніпулювання даними, візуалізації та фреймворків машинного навчання.

4.2 Розробка веб-інтерфейсу

Як згадувалося раніше, для побудови веб інтерфейсу яке буде взаємодіяти з Python сервер API було розроблено веб-додаток за допомогою фреймворку ASP.NET Core MVC. Для розробки веб-додатку та його інтерфейсу існує безліч засобів, але в даній кваліфікаційній роботі використані наступні:

1) MS SQL було обрано через надійність та захищеність та вбудовану підтримку .NET Framework. Завдяки цьому, процедури бази даних, що зберігаються, можуть бути написані на будь-якій мові платформи .NET з використанням повного набору бібліотек, доступних для .NET Framework.

2) ASP.NET Core – вільно-розповсюджуваний крос-платформний фреймворк для створення веб-додатків з відкритим вихідним кодом. Дана

платформа розробляється компанією Майкрософт спільно з спільнотою і має велику продуктивність в порівнянні з ASP.NET. Має модульну структуру і сумісна з такими операційними системами як Windows, Linux і macOS.

3) Entity Framework Core — це кросплатформна об’єктно-реляційна система відображення (ORM) із відкритим кодом для програм .NET. Він надає набір інструментів і бібліотек, які дозволяють розробникам взаємодіяти з реляційними базами даних за допомогою концепцій об’єктно-орієнтованого програмування.

4) HTML (англ. HyperText Markup Language, укр. Мова розмітки гіпертекстових документів) — стандартна мова розмітки веб-сторінок в Інтернеті.

5) JavaScript (JS) — динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Використовується для надання сторінкам додатку інтерактивності.

6) CSS – каскадні таблиці стилів, спеціальна мова, що використовується для відображення сторінок, написаних мовами розмітки даних. Найбільш часто CSS використовують для візуальної презентації сторінок, написаних HTML та XHTML.

Для розробки веб-інтерфейсу було використано наступні інструменти, що відповідають стеку технологій описаному раніше:

1) Visual Studio — це комплексне інтегроване середовище розробки (IDE), створене Microsoft. Він надає низку інструментів і функцій, які дозволяють

розробникам створювати різноманітні типи додатків, включаючи настільні, веб, мобільні, хмарні та розробку ігор. Visual Studio підтримує кілька мов програмування та платформ, що робить його універсальним інструментом для розробників. Visual Studio організовує код у проекти та рішення. Рішення може містити кілька проектів, що полегшує керування складними програмами. Він надає функції для конфігурації проекту, параметрів збірки та шаблонів проектів для швидкого налаштування нових проектів.

2) SQL Server Management Studio (SSMS) — це графічне інтегроване середовище розробки (IDE), надане Microsoft для керування та адміністрування Microsoft SQL Server. Це потужний інструмент, який пропонує широкий спектр функцій для розробки баз даних, керування та налаштування продуктивності. SSMS надає зручний інтерфейс і розширені інструменти для спрощення завдань, пов’язаних із базою даних.

Головним інструментом SQL Server Management Studio є Object Explorer, який дозволяє користувачеві переглядати, отримувати об'єкти сервера, а також повністю ними управляти. Використовується для проектування та БД та внесення поправок до існуючої структури даних БД.

Розглянемо його структуру та можливості. На рисунку 4.8 зображено головну сторінку зі списком попередньо натренованих моделей.

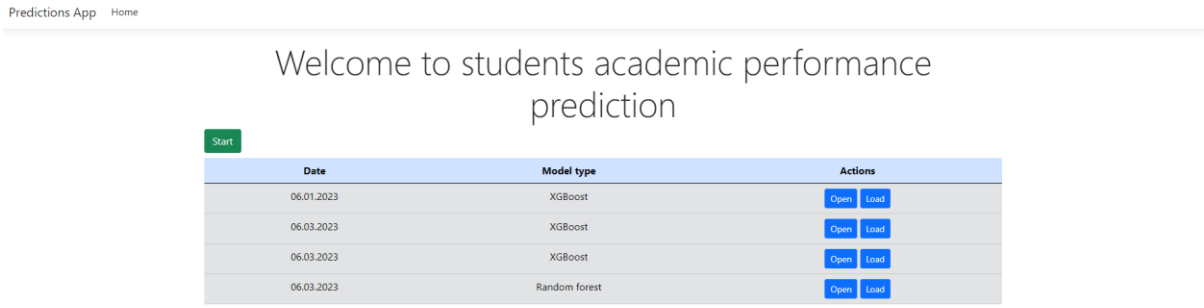


Рис. 4.8. Головна сторінка веб-додатку

Щоб розпочати створення тесту необхідно натиснути кнопку «Start». Після цього буде здійснено перенаправлення на сторінку з формою для генерації, яку зображено на рисунках 4.9 – 4.10. Після того як користувач обере файл з

історичними даним, йому стане доступним для вибору класифікаційне поле “Target” зі списку полів обраного дата сету.

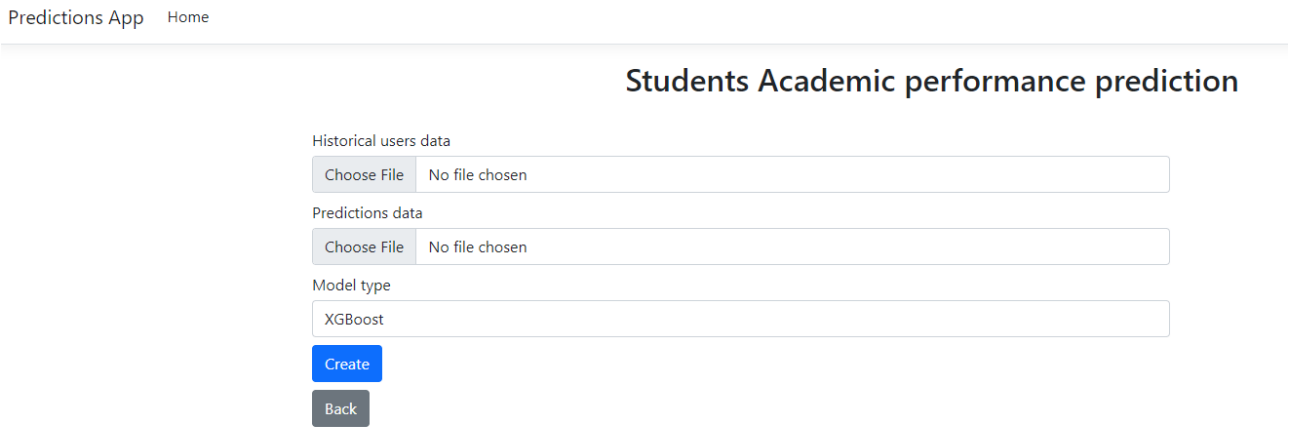


Рис. 4.9. Сторінка для початку тренування моделі

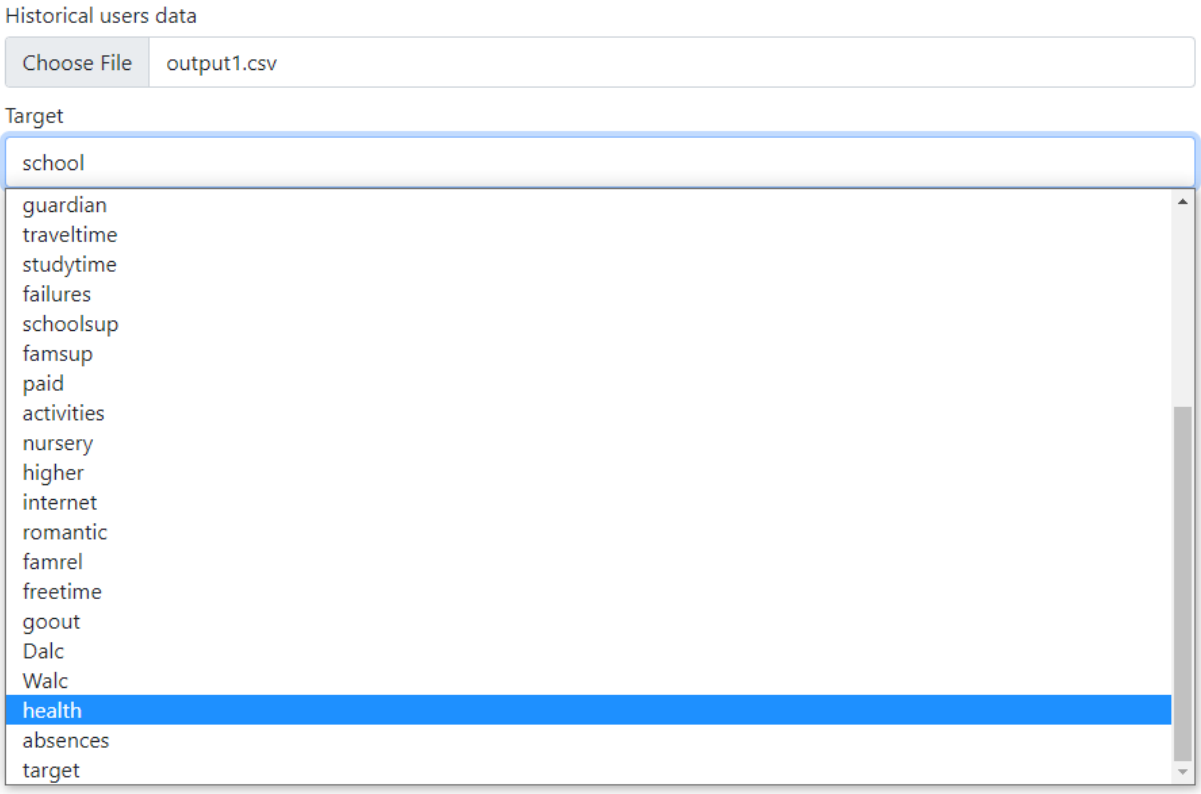


Рис. 4.10 Вибір класифікаційного поля в історичному наборі даних

Дана форма містить поля, які відповідають параметрам серверного API.

Заповнимо форму та спробуємо використати XGboost модель для попередньо обраних наборів даних, після тренування моделі, ми отримаємо

сторінку з результатами, що складається з метрик та переліку релевантних атрибутів та їх значенням (рис. 4.11 – 4.12).

Results

Metric	
Accuracy	79%
Precision	51%
Clasification error	22%
Recall	51%
F1-Score	51%

Column	Feature name	Importance
failures	-	18%
health	-	8%
school	MS	7%
Dalc	-	7%
higher	no	5%
absences	-	5%
famrel	-	4%
activities	yes	4%
guardian	father	4%
Walc	-	4%
goout	-	3%
traveltime	-	3%
age	-	3%
address	R	3%

Рис. 4.11 Сторінка результатів

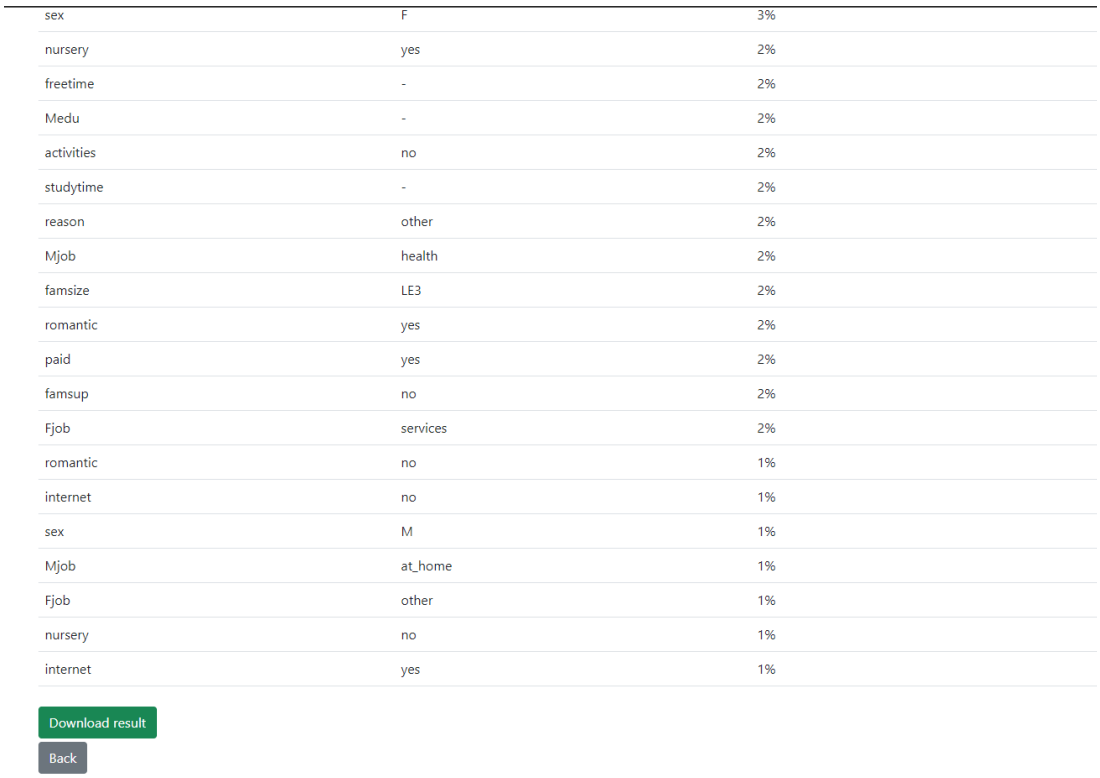


Рис. 4.12 Сторінка результатів

Для завантаження csv файлу з результатом прогнозного аналізу потрібно натиснути кнопку “Download” (рис. 4.13).

Для завантаження попередньо натренованої моделі, на сторінці зі списком потрібно натиснути кнопку “Load” після цього користувач зможе вибрати файл з новими даними та використати уже натреновану модель для прогнозного аналізу.

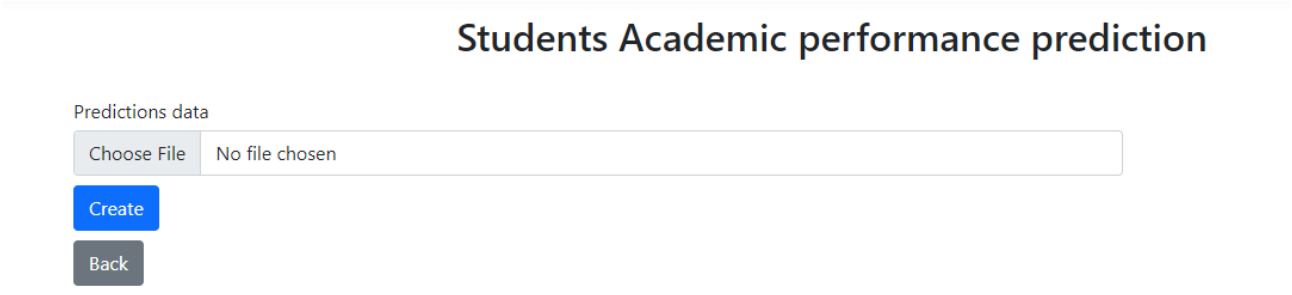


Рис. 4.13 Форма для використання попередньо натренованої моделі

Після заповнення та відправки форми користувач отримає csv файлу з результатом прогнозного аналізу.

В цілому в веб додатку можна виокремити наступні модулі:

1. PredictionApp.EF (DataAccesLayer) – відповідає за роботу з БД,реалізований на основі ORM (Object-relational mapping) – Microsoft.EntityFrameworkCore

- Repositories – класи, які реалізують інтерфейси (PredictionApp.Core.Repositories) та відповідають за роботу з базою даних, а саме створення, редагування, видалення та вилучення даних;

2. PredictionApp.Core – бізнес логіка:

- Repositories – інтерфейси, що описують поведінку роботи з базоюданих;
- DataAnalysis – класи, які інкапсулють в собі логіку роботи з Flask Python APi;
- Entities – класи, що виступають моделями даних для БД;

3. PredictionApp.Web – веб-частина системи, додаток ASP.NET MVC Core.

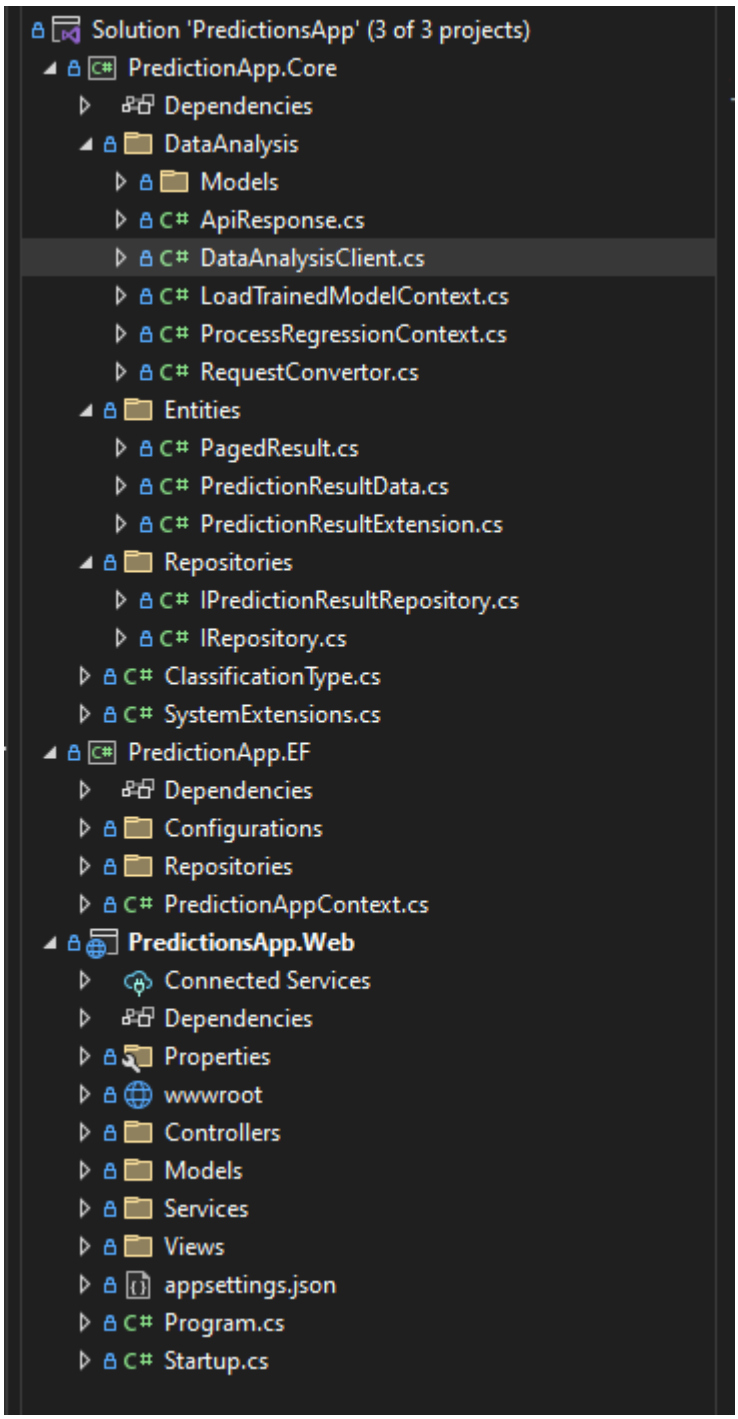


Рис. 4.14 Структура веб-додатку

Також потрібно взяти до уваги структуру таблиці, що зберігає дані додатку.

Таблиця 4.1 Структура таблиці «PredictionResults»

Назва поля	Тип даних	Опис поля
Id	Uniqueidentifier	Первиний ключ
Type	nvarchar(100)	Тип використаної моделі

Продовження таблиці 4.1

Data	nvarchar(max)	Дані про проведене дослідеження у форматі JSON
FolderPath	nvarchar(100)	Шлях до папки з результаатами дослідження та збереженої моделі
PublishDate	datetime2	Дата проведення дослідження

Отриманий веб-додаток додаток можна підсумувати наступною діаграмою компонентів (рис. 4.15).

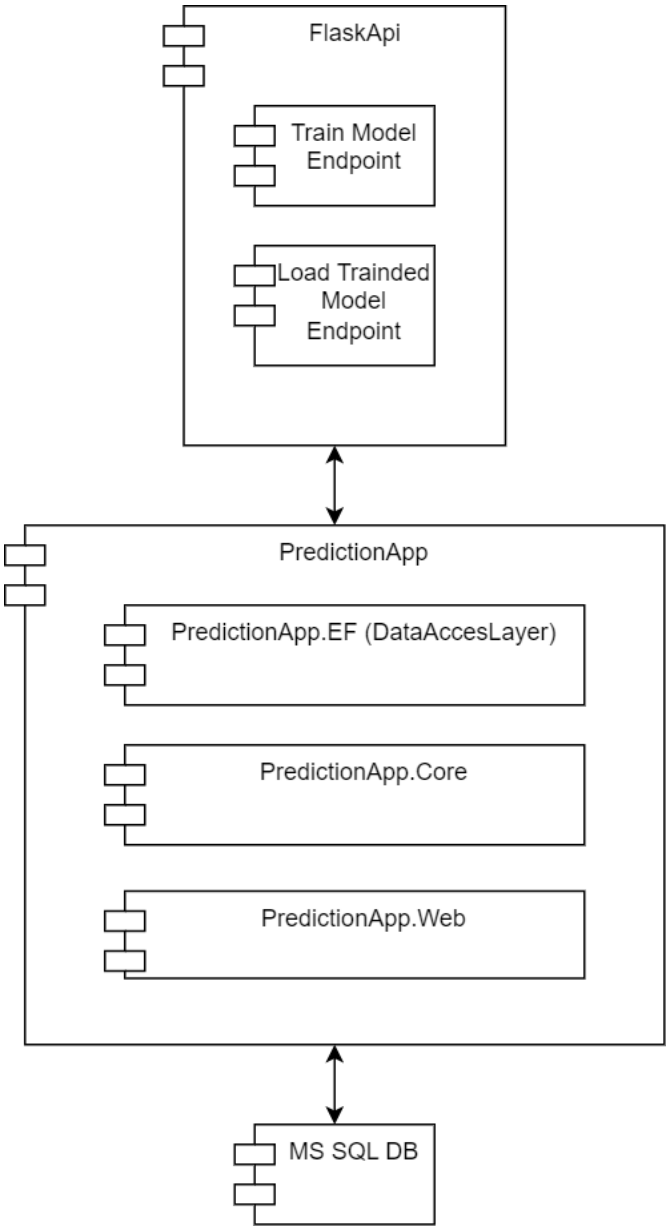


Рис. 4.15 Діаграма компонентів системи

Висновки до розділу 4

У цьому розділі були розглянуті розроблений API сервера та веб-інтерфейс. Серверна частина написана з використанням мови програмування Python та бібліотеки Flask. Він складається лише з двох кінцевих точок. використовуватися для навчання та використання їх на нових даних, а другий надає доступ до попередньо натренованих моделей.

Продемонстровано найважливіші моменти роботи з системою, описано структуру інтерфейсу. Була детально розглянута структура та додатку, спроектований та описаний алгоритм роботи системи.

Були визначені основні об’єкти в системі та зв’язки між ними. Проведено функціональне тестування веб-додатку.

ВИСНОВКИ

В даній кваліфікаційної роботи було визначено проблему прогнозного моделювання та аналітики успішності студентів з використанням алгоритмів машинного навчання. Дослідження і аналіз алгоритмів машинного навчання для прогнозного моделювання та аналітики успішності студентів є важливим кроком у покращенні освітнього процесу. Відштовхуючись від цього, було проведено дослідження та проаналізовано декілька алгоритмів машинного навчання.

В першому розділі було розглянуто основну проблематику кваліфікаційної роботи. Розглянуто та проаналізовано основні методи прогнозного аналізу та опрацьовано теорію пов’язану з алгоритмами машинного навчання.

В другому розділі, на основі проведеного дослідження та аналізу аналогів було здійснено чітку постановку завдання для проведення розробки власного додатку. Було обрано інструментарій для реалізації алгоритму та обрано інструментарій для розробки веб-інтерфейсу та інфраструктурної частини додатку.

В третьому розділі було проведено аналіз метрик та результатів натренованої моделі. З матриці помилок можна побачити, що ми отримали досить непогані результати після підбору параметрів для моделей. В основному моделі показали досить гарні результати. Також проведено приклад використання отриманих даних.

В четвертому розділі були розглянуті розроблений API сервера та веб-інтерфейс. Продемонстровано найважливіші моменти роботи з системою, описано структуру інтерфейсу.

З наукової точки зору кожна фаза життєвого циклу машинного навчання може бути досліджена далі. Науковці, які зараз досліджують нові алгоритми фільтрації, методи та процедури нормалізації, можуть виміряти їхню ефективність у контексті освіти.

У результаті виконання завдання поставленого, було отримано реалізацію алгоритму прогнозного аналізу з використанням моделей RandomForest та

XGBoost, який підходить для подальшої модернізації та практичному застосуванні. З освітньої точки зору, це дослідження може допомогти викладачам визначити студентів, які перебувають у групі ризику щодо поганої успішності, і може допомогти педагогам вжити своєчасних коригувальних заходів.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

[1] Зубрицький В. В. ОГЛЯД МЕТОДІВ ТА ТЕХНОЛОГІЙ ШТУЧНОГО ІНТЕЛЕКТУ В ЕЛЕКТРОННОМУ НАЧАННІ //Сучасні виклики і актуальні проблеми науки, освіти та виробництва: міжгалузеві диспути [зб. наук. пр.]: матеріали XXIV міжнародної науково-практичної інтернет-конференції (м. Київ, 28 січня 2022 р.). Київ, 2022. С. 43-45.

[2] Zubrytskyi Vasyl, Vakaliuk Tetiana. Overview of methods of intellectual data analysis. Тези доповідей II Міжнародної студентської наукової конференції (Т. 2), м. Одеса, 17 грудня, 2021 рік / ГО «Молодіжна наукова ліга».

[3] Al-Shabandar, R., Hussain, A., Laws, A., Keight, R., Lunn, J., & Radi, N. (2017). 'Machine learning approaches to predict learning outcomes in Massive open online courses', Proceedings of the International Joint Conference on Neural Networks

[4] Chilukuri, K. C. (2020). A novel framework for active learning in engineering education mapped to course outcomes. Procedia Computer Science, 172, 28–33.

[5] Dewan, M. A. A., Murshed, M., & Lin, F. (2019). Engagement detection in online learning: a review. Smart Learning Environments, 6(1).

[6] Dhawan, S. (2020). Online learning: A Panacea in the time of COVID-19 crisis. Journal of Educational Technology Systems, 49(1), 5–22

[7] Python Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://www.python.org/doc/.

[8] Вакалюк Т. А. Хмаро орієнтоване навчальне середовище: категорійно-понятійний апарат / Науковий вісник Ужгородського національногоуніверситету: Серія «Педагогіка. Соціальна робота» – 2015. – No35 –С. 38-41. Режим доступу: http://eprints.zu.edu.ua/19458/1/selection.pdf

[9] Ko, C. Y., & Leu, F.-Y. (2021). Examining successful attributes for undergraduate students by applying machine learning techniques. IEEE Transactions on Education, 64(1), 50–57.

[10] Krawczyk, B., Minku, L. L., Gama, J., Stefanowski, J., & Woźniak, M. (2017). Ensemble learning for data stream analysis: A survey. Information Fusion, 37, 132–156

[11] Liu, Z., Yang, C., Rüdian, S., Liu, S., Zhao, L., & Wang, T. (2019). Temporal emotion-aspect modeling for discovering what students are concerned about in online course forums. Interactive Learning Environments, 27(5–6), 598–627. https://www.tandfonline.com/doi/full/10.1080/10494820.2019.1610449

[12] Liu, Z., Zhang, N., Peng, X., Liu, S., Yang, Z., Peng, J., Su, Z., & Chen, J. (2022b). Exploring the relationship between social interaction, cognitive processing and learning achievements in a MOOC discussion forum. Journal of Educational Computing Research., 60(1), 132–169. https://journals.sagepub.com/doi/10.1177/07356331211027300

[13] Marczyk, G. R., Dematteo, D., & Festinger, D. (2005). Essentials of research design and methodology. John Wiley & Sons

[14] Moscoso-Zea, O., Paredes-Gualtor, J., & Lujan-Mora, S. (2018). A holistic view of data warehousing in education. IEEE Access, 6, 64659–64673

[15] Moscoso-Zea, O., & Lujan-Mora, S. (2017). Knowledge management in higher education institutions for the generation of organizational knowledge. In 2017 12th Iberian Conference on Information Systems and Technologies (CISTI).

[16] Perveen, A. (2016). Synchronous and asynchronous e-language learning: A case study of virtual university of Pakistan. Open Praxis, [online] 8(1). Available at: https://fles.eric.ed.gov/fulltext/EJ1093436.pdf

[17] Russell, R. (2018). Machine learning step-by-step guide to implement machine learning algorithms with Python. Editorial: Columbia, Sc.

[18] Shahiri, A. M., Husain, W., & Rashid, N. A. (2015). A review on predicting student's performance using data mining techniques. Procedia Computer Science, 72, 414–422.

[19] Shida, N., Osman, S., & Abdullah, A. H. (2019). Students' perceptions of the use of asynchronous discussion forums, quizzes, and uploaded resources. International Journal of Recent Technology and Engineering, 8(2S9), 704–708.

ДОДАТКИ

Додаток А

Посилання на записник для реалізації алгоритму машиного навчання: https://gitlab.com/master20221/ipzm-21-1/zubrytskyi-vasyl/Diploma/-/blob/main/Documentation/ML_Algorithm.ipynb

Посилання на код серверного Python API: https://gitlab.com/master20221/ipzm-21-1/zubrytskyi-vasyl/Diploma/-/tree/main/FlaskApi

Посилання на код веб-інтерфейсу реалізованого за допомогою ASP.NET Core MVC:https://gitlab.com/master20221/ipzm-21-1/zubrytskyi-vasyl/Diploma/-/tree/main/PredictionsApp