МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних
інформаційних систем, д-р техн. наук, проф.
_____Ю. П. Кондратенко
«____»_____2024 р.

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

СИСТЕМА АНАЛІЗУ АКАДЕМІЧНОЇ УСПІШНОСТІ СТУДЕНТІВ ЗА ДОПОМОГОЮ МАШИННОГО НАВЧАННЯ

Спеціальність 122 «Комп'ютерні науки»

122 – КРМ – 601.2610106

Виконав студент 6-го курсу, групи 601
_____ М. Ю. Галущак
«19» лютого 2024 р.

Керівник: канд. техн. наук, доцент
_____Є. В. Сіденко
«19» лютого 2024 р.

Миколаїв – 2024

Чорноморський національний університет ім. Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

Освітньо-кваліфікаційний рівень **магістр**
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри інтелектуальних інформаційних систем, д-р техн. наук, проф.
_____ Ю. П. Кондратенко
«____» _____20___ р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи
Галущаку Максиму Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи «Система аналізу академічної успішності студентів за допомогою машинного навчання».
Керівник роботи Сіденко Євген Вікторович, канд. техн. наук, доцент
Затв. наказом Ректора ЧНУ ім. Петра Могили від « » жовтня 2023 р. №
2. Строк подання студентом роботи 19 лютого 2024 р.
3. Вхідні (початкові) дані до роботи: експертні оцінки аналізу академічної успішності студентів за допомогою машинного навчання; пріоритетність критеріїв. Очікуваний результат: система аналізу академічної успішності студентів за допомогою машинного навчання.
4. Зміст пояснювальної записки (перелік питань, які потрібно розглянути):

– аналіз сучасного стану задачі прогнозування академічної успішності студентів за допомогою машинного навчання;
– огляд існуючих методів машинного навчання;
– експертне оцінювання технологій прогнозування академічної успішності студентів за допомогою машинного навчання;

– порівняльний аналіз результатів застосування обраних методів машинного для розв’язання поставленої задачі.

5. Перелік графічного матеріалу: презентація.

6. Завдання до спеціальної частини: Захист від іонізуючих випромінювань.

7. Консультанти:

Розділ	**Прізвище, ініціали та посада консультанта**	**Підпис**
Спеціальна частина з охорони праці	д-р біол. наук Л. І. Григор’єва	
Методична частина	канд. техн. наук, доцент Є. В. Сіденко	

Керівник роботи __канд. техн. наук, доцент. Сіденко Є. В._____
(наук. ступінь, вчене звання, прізвище та ініціали)

_____
(підпис)

Завдання прийнято до виконання ___Галущак М. Ю._____
(прізвище та ініціали)

_____
(підпис)

Дата видачі завдання «__31__» _____жовтня____ 2023 р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи магістра

Тема: «Система аналізу академічної успішності студентів за допомогою машинного навчання»

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Визначення керівника і теми КРМ. Подання заяви на затвердження теми КРМ	01.09.2023	20.10.2023	Виконано
2.	Отримання завдання на виконання КРМ	21.10.2023	10.11.2023	Виконано
3.	Складання календарного плану	11.11.2023	15.11.2023	Виконано
4.	Огляд літератури за темою дослідження. Аналіз	16.11.2023	27.11.2023	Виконано
5.	Проходження переддипломної практики, збір та аналіз матеріалів до КРМ	28.11 2023	18.12.2023	Виконано
6.	Аналіз предметної області та розробка технічного завдання	19.12.2023	22.12.2023	Виконано
7.	Проектування та програмна реалізація	23.12.2023	15.01.2024	Виконано
8.	Робота над розділами фахової частини КРМ	16.01.2024	24.12.2024	Виконано
9.	Розробка методичної частини КРМ та спеціальної частини з охорони праці	25.01.2024	01.02.2024	Виконано
10.	Обговорення отриманих результатів з керівником та попередній захист КРМ	02.02.2024	3.02.2024	Виконано
11.	Корегування роботи за результатами попереднього захисту	4.02.2024	6.02.2024	Виконано
12.	Остаточне оформлення пояснювальної записки та слайдів доповіді до зихисту	7.02.2024	9.02.2024	Виконано
13.	Подання рецензенту та рецензування КРМ	9.02.2024	12.02.2024	Виконано
14.	Подання КРМ, її електронної копії та інших документів (відгуку, рецензії) до захисту	15.02.2024	16.02.2024	Виконано
15.	Захист КРМ перед ЕК	23.02.2024	23.02.2024	Виконано

Розробив студент _Галущак М. Ю._____ _____

(прізвище та ініціали) (*підпис*)

Керівник роботи _канд. техн. наук, доцент. Сіденко Є. В._ _____

(наук. ступінь, вчене звання, прізвище та ініціали) *(підпис)*

«__» _____ 2024 р.

АНОТАЦІЯ

до кваліфікаційної роботи магістра
студента групи 601 ЧНУ ім. Петра Могили

Галущака Максима Юрійовича

на тему: **“СИСТЕМА АНАЛІЗУ АКАДЕМІЧНОЇ УСПІШНОСТІ СТУДЕНТІВ ЗА ДОПОМОГОЮ МАШИННОГО НАВЧАННЯ”**

Тема прогнозування успішності студентів за допомогою машинного навчання є надзвичайно актуальною в контексті сучасної вищої освіти. Персоналізація навчання та попередження відпадання студентів стають ключовими аспектами, які можуть покращити якість освіти та забезпечити більший успіх у навчанні.

Об’єктом є процес прогнозування академічної успішності студентів.

Предметом є алгоритми машинного навчання для прогнозування академічної успішності студентів.

Метою є розробка системи для прогнозування академічної успішності студентів за допомогою машинного навчання.

В результаті виконання роботи було досліджено декілька алгоритмів машинного навчання для прогнозування, проаналізовано вплив їх внутрішніх параметрів на роботу алгоритмів, визначені основні їх переваги та недоліки, а також розроблено програмне забезпечення, в якому реалізовані відповідні методи.

Дана робота складається з розділів. Кожен розділ відповідно присвячений: аналізу предметної області, математичним моделям і методам, використаним у магістерській роботі, моделюванню і проектуванню системи для прогнозування академічної успішності студентів, аналізу отриманих результатів, охороні праці, методичній частині магістерської роботи. Загальний обсяг роботи – 103 сторінки. Кваліфікаційна робота магістра містить 1 додаток, 37 рисунків і 45 посилання на літературних джерел.

Ключові слова: Машинне навчання, прогнозування, успішність студентів

ABSTRACT

to the master's qualification work
by the student of the group 601 of Petro Mohyla Black Sea National University

Halushchak Maksym

“SYSTEM OF ANALYSIS OF ACADEMIC SUCCESS OF STUDENTS WITH THE HELP OF MACHINE LEARNING”

The topic of predicting student success using machine learning is extremely relevant in the context of modern higher education. Personalization of learning and prevention of student dropout are becoming key aspects that can improve the quality of education and ensure greater academic success.

The object is the process of predicting students' academic performance.

The subject is machine learning algorithms for predicting students' academic performance.

The goal is to develop a system for predicting students' academic performance using machine learning.

As a result of the work, several machine learning algorithms for forecasting were investigated, the influence of their internal parameters on the operation of the algorithms was analyzed, their main advantages and disadvantages were determined, and software was developed in which the corresponding methods were implemented.

This work consists of sections. Each section is respectively dedicated to: analysis of the subject area, mathematical models and methods used in the master's work, modeling and design of the system for predicting the academic success of students, analysis of the obtained results, labor protection, methodical part of the master's work. The total volume of work is 103 pages. Master's thesis contains an 1 appendix, 37 figures, and 45 references to literary sources.

Keywords: Machine learning, forecasting, student success

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ВДТ – Візуальний дисплейний термінал

ПЕОМ – Персональні електронні обчислювальні машини

КПО – Коефіцієнт природної освітленості

ПК – Персональний комп’ютер

ЕОМ – Електронна обчислювальна машина

ССБП – Система стандартів безпеки праці

ML (англ. machine learning) - машинне навчання

SVM (англ. support vector machine) - машина опорного вектора

FC (англ. functional connectivity) - функціональна зв’язність

DL (англ. deep learning) - глибоке навчання

RBM (англ. restricted Boltzmann machine) - обмежена машина Больцмана

ВСТУП

Застосування машинного навчання для прогнозування успішності дозволяє враховувати індивідуальні особливості кожного студента. Аналізуючи різноманітні дані, такі як академічні показники, взаємодія з навчальним матеріалом та соціальні аспекти, можна визначити потенційні труднощі та вчасно реагувати на них. Це сприяє створенню індивідуалізованих навчальних планів та наданню рекомендацій, які підтримують успіх студентів.

Важливим аспектом є також попередження відпадання студентів. Системи машинного навчання можуть виявляти ризикованих студентів, які можуть стикнутися з труднощами або втратити мотивацію. Шляхом вчасної інтервенції та надання підтримки можна підвищити ймовірність того, що студенти завершать навчання та досягнуть успіху.

Оптимізація викладання та навчальних методик є ще однією вагомою перевагою застосування машинного навчання в освіті. Аналіз великих обсягів даних дозволяє визначити ефективні та неефективні підходи до навчання, що може призвести до подальшої оптимізації освітнього процесу.

Не менш важливою є ідея взаємодії зі студентами та залучення їх до навчання. Системи прогнозування успішності можуть стимулювати більш активну участь студентів, надаючи їм індивідуалізовані рекомендації та підтримку. Це створює позитивний цикл, де студенти більше зацікавлені у своєму навчанні та досягають кращих результатів.

У контексті сучасних викликів у сфері освіти, таких як пандемія та зміни в форматах навчання, прогнозування успішності студентів залишається ключовим інструментом для підтримки та покращення якості вищої освіти.

Об’єктом дослідження є процес прогнозування академічної успішності студентів.

Предметом дослідження є алгоритми машинного навчання для прогнозування академічної успішності студентів.

Мета дослідження – є розробка системи для прогнозування академічної успішності студентів за допомогою машинного навчання.

Досягнення поставленої мети обумовлює необхідність вирішення наступних завдань:

1. Дослідити теоретичні засади для прогнозування академічної успішності студентів за допомогою машинного навчання.
2. Обґрунтувати вибір інструментальних засобів розробки системи для прогнозування академічної успішності студентів.
3. Розробити та здійснити програмну реалізацію для прогнозування академічної успішності студентів.

Дослідження прогнозування успішності студентів за допомогою алгоритмів машинного навчання вимагає систематичного та об'єктивного підходу для отримання надійних результатів.

Основні принципи методології включають в себе постановку дослідницьких питань, вибір джерел та типів даних, визначення змінних та параметрів, вибір алгоритмів машинного навчання, оцінку точності та релевантності моделей, контроль біасів та етики, узагальнення та розширення результатів, а також комунікацію та публікації.

Важливо чітко сформулювати дослідницькі питання, спрямовані на вирішення конкретних завдань. Вибір джерел та типів даних повинен враховувати специфіку вищої освіти та етичні аспекти. Визначення ключових змінних та параметрів є необхідним етапом для побудови адекватних моделей.

Вибір алгоритмів машинного навчання повинен бути обгрунтованим, з урахуванням особливостей досліджуваної проблеми та характеристик даних. Оцінка точності та релевантності моделей є важливою для визначення їхньої ефективності в конкретному контексті.

Наукова новизна одержаних результатів. Вивчення прогнозування успішності студентів за допомогою алгоритмів машинного навчання відкриває широкий простір для наукових досліджень та інновацій. Основною новизною у

цьому контексті є розробка та вдосконалення методів, що спеціально враховують вимоги вищої освіти та унікальні аспекти академічного процесу.

Одним з ключових напрямків досліджень є комплексне врахування факторів, які впливають на успішність студентів. Моделі повинні включати не лише академічні показники, але й інші аспекти, такі як соціальна взаємодія, емоційний стан та участь у додаткових заходах. Це відкриває можливості для більш глибокого розуміння контексту успішності вищої освіти.

Застосування адаптивних моделей є іншою ключовою характеристикою досліджень. Алгоритми повинні бути здатні адаптуватися до змін умов, таких як перехід до дистанційного навчання чи зміни педагогічних підходів. Це важливо для забезпечення стабільної та точної прогностичної спроможності навчальних систем.

Дослідницькі зусилля також спрямовані на врахування контекстуальних аспектів, таких як різниці між дисциплінами, факультетами та програмами навчання. Розуміння того, які фактори важливі для конкретних галузей, може покращити точність моделей та їх релевантність в практичному застосуванні.

Важливим вектором досліджень є інтеграція високої інтерпретованості в моделі. Це дозволяє враховувати "чому" та "як" модель зробила конкретне прогнозування, що є ключовим для прийняття обґрунтованих рішень у сфері вищої освіти.

Подальші аспекти новизни включають врахування часових аспектів та динамічність успішності студента протягом тривалості навчання. Це особливо актуально в умовах змін у навчальному процесі та відповіді на нові виклики.

Не останню роль відіграє увага до етичних аспектів та конфіденційності даних студентів. Дослідники вдосконалюють методи, які гарантують етичне використання та захист особистої інформації.

Загальна новизна у дослідженнях полягає у вирішенні складних завдань, пов'язаних із прогнозуванням успішності студентів, з огляду на унікальні вимоги вищої освіти та широкий спектр факторів, що впливають на академічну продуктивність.

Практичне значення отриманих результатів полягає в тому, що вони можуть сприяти персоналізації навчання, що враховує індивідуальні потреби та здібності кожного студента. Крім того, вони можуть бути використані для попередження відпадання студентів, ідентифікації тих, хто стикається з труднощами.

Практичне застосування таких моделей включає оптимізацію розподілу ресурсів в університетах, щоб ефективно використовувати викладацький персонал, приміщення та інші ресурси. Важливо також зазначити, що отримані результати можуть підтримувати педагогічні інновації та сприяти розвитку нових методик викладання.

Застосування алгоритмів машинного навчання також може допомогти студентам готуватися до майбутніх викликів на ринку праці, прогнозуючи їхні успіхи та надаючи можливість отримати необхідні навички та знання. Повідомлення студентів про індивідуалізовані рекомендації та підтримку може сприяти підвищенню їхньої залученості та мотивації в навчанні.

Такий підхід може також визначати області ризику та допомагати вчасно реагувати на проблеми, що виникають у студентів. Загалом, отримані результати дослідження мають значущі наслідки для покращення ефективності та якості освітнього процесу в університетах.

Структура магістерської роботи. Відповідно до мети, завдань і предмета дослідження, магістерська робота містить основну, методичну та спеціальну частини. Основна частина магістерської роботи складається із вступу, трьох розділів, висновку, списку використаних джерел та 1 додатку. Загальний обсяг магістерської роботи – 103 сторінок, із них основного тексту основної частини – 66 сторінок, методичної частини – 17 сторінок, спеціальної – 20 сторінок. Кількість використаних джерел – 43.

1 АНАЛІЗ СУЧАСНОГО СТАНУ ЗАДАЧІ СИСТЕМ ПРОГНОЗУВАННЯ АКАДЕМІЧНОЇ УСПІШНОСТІ СТУДЕНТІВ ЗА ДОПОМОГОЮ МАШИННОГО НАВЧАННЯ

1.1 Огляд предметної сфери

Прогнозування успішності студентів в сучасному освітньому контексті висвітлює глибокий інтерес до використання аналітики та технологій для підвищення ефективності навчального процесу.

Аналітика в освіті є ключовим напрямком, де спеціалісти вивчають різноманітні аспекти даних, починаючи від історичних академічних показників студентів до їхньої активності в електронних навчальних платформах. Важливо враховувати, що успішність студента визначається не лише оцінками, але й його здатністю до активної участі в уроках, групових проектах та інших показниках.

Машинне навчання в освіті надає можливість розвивати прогностичні моделі, які здатні адаптуватися до унікальних особливостей кожного студента. Індивідуалізація стає ключем до успішності, де враховуються не лише загальні статистичні тенденції, але і особливості кожного окремого випадку.

Соціальна та поведінкова аналітика забезпечують вивчення взаємодії між студентами та їхньою соціальною активністю. Врахування цих аспектів дозволяє отримати більш повний образ успішності, оскільки взаємодія з колегами та викладачами може значно впливати на навчальний процес.

Впровадження технологій включає в себе створення інтегрованих систем управління освітою та електронних навчальних платформ. Це сприяє автоматизації процесів ведення статистики, а також надає можливість реального часу для аналізу та втручання в ситуації, які можуть впливати на успіх студентів.

Прогнозування успішності учнів є одним із невід’ємних завдань у секторі освіти. Існують різні фактори, які можуть вплинути на успішність студентів і які опосередковано впливають на акредитацію університету. Підтримувати високий рівень навчання в університетах може бути складно через низьку успішність

студентів. Широкий спектр досліджень відкритий для вдосконалення системи навчання на основі потреб студентів. Актуальне питання інтелектуального аналізу даних в освіті відкриває нові можливості дослідження. Методи інтелектуального аналізу даних, які застосовуються в освітній сфері, можна назвати освітнім інтелектуальним аналізом даних. Це процес автоматичного визначення корисної інформації з необроблених даних. Методи машинного навчання в освітньому секторі досліджуються для контролю значущих шаблонів, які покращують знання студентів, і навчальні заклади прийматимуть рішення на основі цих шаблонів.

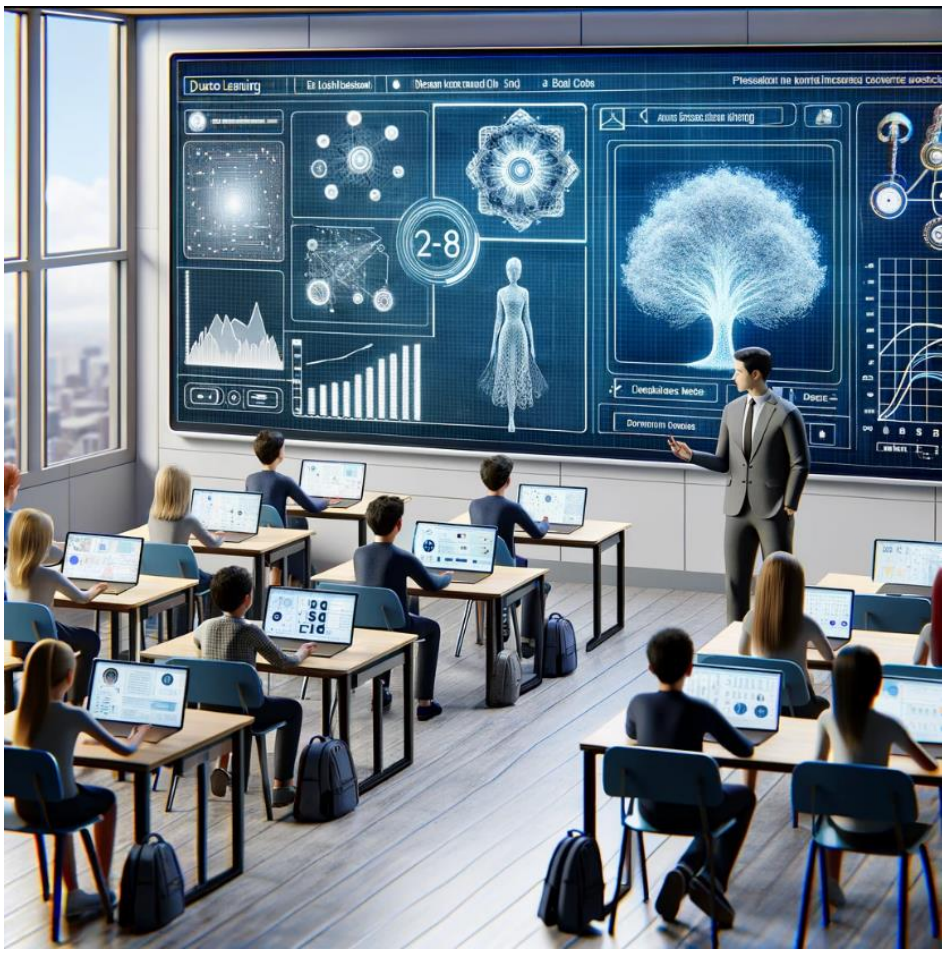

Рисунок 1.1 - Візуалізація освітнього процесу за допомогою методів машинного навчання

Сучасний навчальний заклад працює в дуже конкурентному та складному середовищі. Чи є оцінювання успішності студента, забезпечення високого рівня

викладання та методи навчання для прогнозування успішності студента та визначення цілей дослідження деякими складними питаннями, з якими стикається більшість сьогоднішніх університетів. Плани підвищення успішності студентів впроваджуються в університетах для подолання проблем студентів під час навчання. Прогнозування успішності студентів на початковому рівні та протягом навчального року допомагає організації розробити та оцінити заводи залучення, де обидва керівництва, наприклад викладач і студент, є бенефіціарами плану прогнозування успішності студентів. Електронне навчання швидко розвивається та просувається форма навчання, коли студенти записуються на онлайн-класи.

Платформи електронного навчання, такі як Massive Open Online Course (MOOC) користуються максимальними перевагами ADM у розгортанні та створенні автоматичної системи оцінювання.

Рисунок 1.2 – Порівняння найпопулярніших платформ електронного навчання

Ця платформа використовує інтелектуальні інструменти, які збирають цінну інформацію для користувачів, таку як частота доступу студентів до системи електронного навчання, точність відповіді студента на запитання, кількість годин,

витрачених на читання та перегляд відеоуроків. Машинне навчання — це підполе ШІ, де система машинного навчання вивчає дані, генерує шаблони та прогнозує результати. Методи машинного навчання можуть автоматично та швидко аналізувати великі та дуже складні дані з правдивими результатами. Алгоритм машинного навчання є корисним інструментом для раннього прогнозування низької успішності студента на основі отриманих даних журналу. Ця методика є більш досконалою, ніж традиційна в університетському містечку, де записи студентів, такі як: відвідуваність, тести, іспит, оцінки, використовуються для оцінювання та прогнозування успішності студента. Це дослідження показало, що алгоритм машинного навчання найчастіше використовувався для прогнозування успішності студентів. Найбільш використовувані алгоритми машинного навчання – це штучна нейронна мережа (ANN), наївний Байєс, дерево рішень і SVM.

З огляду на зростаючу залежність від алгоритмічних рішень у сфері освіти, важливо забезпечити, що такі системи не відтворюють існуючі упередження або не створюють нові бар'єри для студентів. Тому, розробка та імплементація систем прогнозування повинні супроводжуватися строгими протоколами забезпечення справедливості та прозорості.

Крім того, розширення можливостей прогнозування успішності студентів вимагає від освітніх установ впровадження комплексних підходів до підтримки студентського добробуту. Це включає розробку інтервенцій, спрямованих на підтримку студентів, які можуть зіткнутися з академічними труднощами, а також на забезпечення доступності ресурсів для психологічної та емоційної підтримки.

Освітній аналіз даних та машинне навчання пропонують величезний потенціал для покращення освітнього процесу, але їх використання має бути збалансовано з врахуванням потреб та інтересів усіх зацікавлених сторін. Це вимагає від освітніх закладів залучення студентів, викладачів та інших зацікавлених сторін до діалогу про те, як найкраще використовувати ці технології для досягнення спільної мети — підвищення якості освіти та успішності студентів.

1.2 Прогнозування данних за допомогою машинного навчання.

Перший метод машинного навчання використовує вхідні дані для прогнозування числового значення. Дані з попередніх випробувань використовуються для прогнозування чисельних результатів майбутнього випробування. Лінійна регресія застосовується у фінтехах, машинобудуванні та фінансах. Він також використовується для прогнозування погоди, прогнозування цін на нерухомість і зниження рівня відтоку клієнтів. Крім того, дослідження CIO Review демонструє, як модель лінійної регресії оцінює рівень відтоку клієнтів. Якщо у вас є роздрібний бізнес, що розвивається, і ви можете підвищити показники електронної комерції, впровадження ML може зробити для вас чудову роботу.

Серед плюсів цієї техніки машинного навчання можна відзначити здатність передбачати складні зв’язки та відносини між вхідними та вихідними даними. Відомості, отримані в результаті регресії, можна використовувати для моделювання та модифікації зв’язків між змінними. За допомогою техніки регресії дослідники даних можуть визначити важливість кожної змінної та її роль у прогнозуванні вихідних даних. Техніка має і свої мінуси. Основна з них полягає в тому, що модель чутлива до викидів.

Ще один підхід до машинного навчання, — це класифікація. Це один із методів машинного навчання, який класифікує вхідні дані. Кожна вхідна змінна отримує мітку, що позначає категорію. Іншими словами, метод класифікації використовується для зіставлення вхідних даних з однією з категоріальних вихідних міток. Цей тип техніки машинного навчання часто використовується для виявлення шахрайства та маркетингу, діагностики захворювань та в інших сферах.

Плюси алгоритмів класифікації включають можливість використовувати їх для прогнозування зв’язків між входом і виходом. Статті цієї моделі можна використати для визначення того, як кожна змінна впливає на прогноз вихідної мітки. Модель дозволяє дослідникам даних ставити біполярні запитання та використовувати ML для отримання точних відповідей. Що стосується слабких

сторін, модель класифікації іноді не може чітко відповісти, яка змінна відіграє найважливішу роль у процесі прогнозування. Крім того, дисбаланс класів може вплинути на результати впровадження класифікаційної переобладнання або недообладнання вихідних даних.

Кластеризація - це техніка машинного навчання, яка передбачає групування подібних фрагментів даних у кластери або групи на основі аналізу їхніх характеристик і ознак. Кластеризація допомагає інженерам машинного навчання виявляти закономірності та структури під час роботи з даними без міток. Цей метод машинного навчання використовується в біології, обробці зображень, маркетингу та інших сферах. Розробники електронної комерції застосовують цей підхід до сегментації клієнтів на основі їхніх інтересів і поведінки. Ми також можемо застосувати методи кластеризації, коли запускаємо вилучення функцій або стискаємо дані.

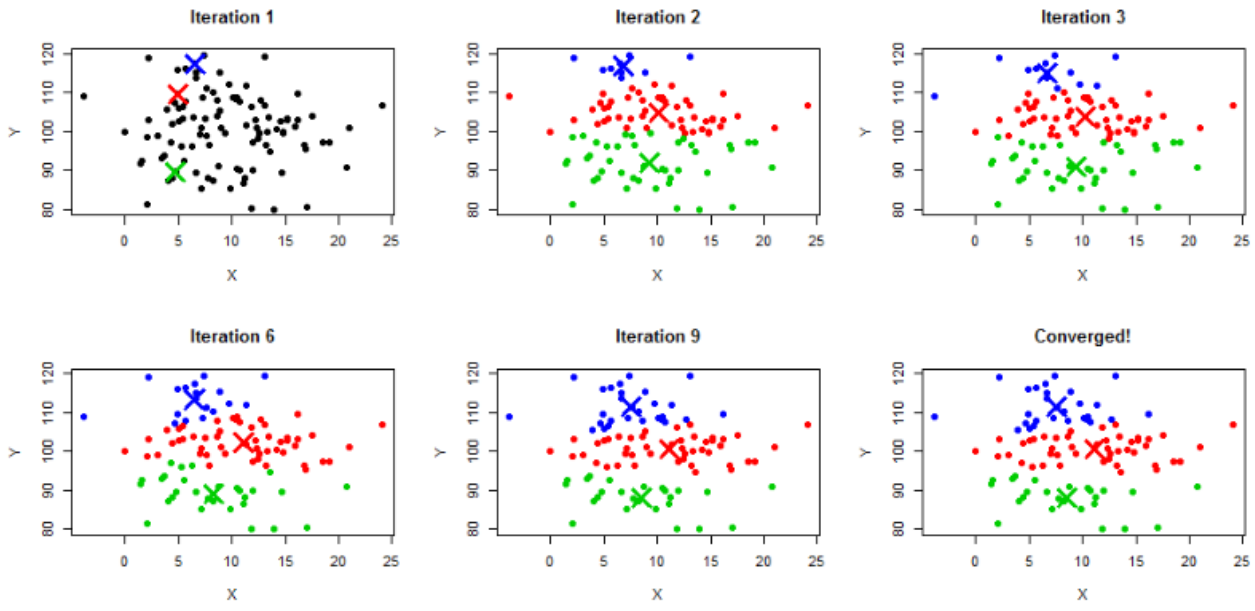

Рисунок 1.3 – Порівняння найпопулярніших платформ електронного навчання

Важливою перевагою кластеризації є те, що вона допомагає виявляти приховані шаблони або структури даних. Це, у свою чергу, веде інженерів програмного забезпечення до нових відкриттів і винаходів. Недоліки цього методу включають проблему вибору правильного алгоритму кластеризації. Крім того, фахівцям з ML може бути важко визначитися з кількістю кластерів для використання в кожному конкретному випадку. Ще одним недоліком цієї техніки машинного навчання є те, що вона вимагає багато обчислювальних ресурсів і на неї може вплинути ініціалізація центроїдів.

Дерево рішень — це один із методів машинного навчання, який вирішує дилему класифікації та передбачає використання певних умов або правил для процесу прийняття рішень. Відповідно до цього підходу вхідні дані поділяються на 2 або більше однорідних наборів даних на основі визначальних атрибутів. Подібно до інших типів методів машинного навчання, дерево рішень використовується у фінансових технологіях (схвалення позики, кредитний скоринг), маркетингу та охороні здоров’я (прогнозування діагнозу на основі даних).

Дерева рішень прості у використанні та інтерпретації, що робить цю модель популярною серед експертів ML. Вони можуть працювати як з числовими, так і з категоріальними даними. Що стосується її недоліків, то деревоподібна структура дерев рішень робить цю модель дуже чутливою навіть до незначних змін даних.

Нейронні мережі - це техніка машинного навчання передбачає глибоке навчання, яке нагадує спосіб навчання людського мозку. Нейронні мережі складаються з кількох вузлів, які з’єднані між собою. Взаємопов’язані вузли або нейрони складають шари та здатні передбачати, аналізувати дані та вчитися на них. Дослідження Google повідомляє, що нейронні мережі зазвичай застосовуються для розпізнавання мови, обробки природної мови (NLP), комп’ютерного зору тощо.

До плюсів нейронних мереж можна віднести їх здатність виявляти та ідентифікувати складні шаблони даних. Це робить цю модель машинного навчання високоточною. Нейронні мережі можуть працювати з різними типами даних,

такими як числові та категоріальні дані. Однак ця техніка ML схильна до переобладнання, якщо вхідні дані неправильно регуляризовані.

Як вказує його назва, техніка виявлення аномалій використовується для виявлення аномальної або незвичної точки даних у вхідному наборі даних. Що означає ненормальний або незвичайний? Це дані, які відрізняються від решти в деяких важливих аспектах. Ця техніка часто використовується у виявленні шахрайства, кібербезпеці (виявлення незвичайного трафіку), охороні здоров’я (виявленні незвичних шаблонів у даних пацієнтів) і фінансах.

Застосування виявлення аномалій приносить багато переваг бізнесу. Це допомагає виявляти шахрайські дії, запобігати кіберзлочинам і виявляти помилки до того, як вони вплинуть на всю систему. Слабка сторона цього підходу машинного навчання полягає в тому, що іноді важко визначити, що таке аномалія.

1.3 Останні дослідження та публікації.

Недавнє метааналітичне дослідження М. Річардсона, К. Абрахама та Р. Бонда проаналізувало понад 400 статей, опублікованих в англомовних психологічних журналах з 1997 по 2010 рік. Вони включали традиційні предиктори (середній бал, тести досягнень, тести інтелекту тощо) і 42 неінтелектуальні змінні.

Вчені розділили ці змінні на п’ять груп: риси особистості, мотиваційні фактори, стратегії саморегуляції навчання, підходи до навчання та ситуативні психосоціальні фактори. Результати показали, що серед особистісних характеристик, найбільш тісно пов’язаних з успіхом у навчанні, були сумлінність, когнітивні потреби та схильність відкладати виконання завдання на останній момент.

У групі мотиваційних факторів з успішністю найбільш тісно пов'язані самооцінка академічної здібності, наявність або відсутність мети отримати високу оцінку, самооцінка виконавської здібності. Що стосується стратегій саморегуляції, то регуляція зусиль і тривога перед тестами видаються найбільш актуальними. У групі підходу до навчання стратегічне навчання виявилося найкращим

прогностичним фактором. З восьми психосоціальних факторів цілеспрямованість, загальний стрес і академічний стрес мали найвищу, хоча й слабку, кореляцію з академічною успішністю.

Автори спробували побудувати рівняння багатовимірної регресії, щоб передбачити середній бал, і виявили, що це можливо на певному рівні. Однак, коли інтелектуальні характеристики контролювалися статистично, ефект був обмеженим. Подібний зв'язок між академічною успішністю та результатами зовнішнього оцінювання було підтверджено вітчизняними даними, але жодне дослідження ще не використовувало тести академічних здібностей або мотиваційні змінні.

У цій статті використовували алгоритми машинного навчання, щоб передбачити труднощі навчання учнів за допомогою LMS, тоді як багато досліджень використовували алгоритми машинного навчання для прогнозування успішності учнів. (Jayaprakash та ін., 2020) зосереджується на виявленні учнів групи ризику та використанні методів випадкового лісу, наївного Байєса та інших комплексних методів для розробки проміжних або ранніх втручань для покращення успішності учнів. Ми створили модель для класифікації атрибутів, які використовуються як механізми попередження . Вони дійшли висновку, що такі фактори, як стать, структура сім'ї, батьківський статус, освіта матері та батька, а також функціонування матері та батька є одними з факторів впливу, які можуть негативно вплинути на успішність учнів (Kime та ін., 2019), використовувалося інтерактивне машинне навчання. . Класифікувати обчислювальні навички учнів. Вони почали використовувати машинні методи для класифікації навичок і прогнозування того, які навички додадуть досвідчені вчителі, щоб покращити успішність учнів. (Bajpai та ін., 2019) порівняли кілька алгоритмів машинного навчання, щоб вибрати найбільш прийнятний і ефективний алгоритм для прогнозування академічної успішності дослідників.

Вони досягли точки, коли деякі алгоритми було легко реалізувати та зрозуміти, а інші були громіздкими та вимагали величезної кількості часу для

обчислень. (Hussain et al., 2018) використали кілька алгоритмів машинного навчання, щоб передбачити найефективніші сесії електронного навчання для студентів. Вони виявили, що алгоритми RF і DL підходять для прогнозування сеансів, які є корисними в електронному навчанні, і на основі своїх прогнозів вони дослідили вплив таких факторів, як участь сім'ї, навчальне середовище та стиль викладання, на продуктивність сеансу. сприяючі фактори. (Athani та ін., 2017) запропонували систему прогнозування, що використовує алгоритм SVM.

Результати показують, що алгоритм SVM можна використовувати для прогнозування успішності студентів і надання відділам навчальних закладів інформації про ситуацію студентів, щоб вони могли надавати відповідні додаткові навчальні завдання, щоб допомогти покращити успішність студентів. Я це зробив. (Sorour та ін., 2014) використовували алгоритми SVM та ANN для прогнозування балів студентів відповідно до їхніх коментарів. Результати показали, що точність прогнозування за допомогою SVM була вищою, ніж за допомогою ANN.Стаття з назвою "Mapping Student's Performance Using K-Mean Cluster Algorithm" (Картування успішності студентів за допомогою алгоритму кластеризації K-Means) була опублікована у "Agriculture and Agricultural Science Procedia" в 2015 році. Авторами дослідження є Harwati та інші. Вона зосереджується на застосуванні алгоритму кластеризації K-Means для аналізу та класифікації студентів на основі їхніх демографічних характеристик, академічних досягнень та інших відповідних змінних.

Ця стаття зосереджена на використанні алгоритму кластеризації K-means для картирования стану студентів з метою покращення їхньої академічної успішності в університетському контексті. Дослідження охоплює дані приблизно 300 студентів, використовуючи шість змінних: демографічні характеристики (стать, походження), середній бал, оцінки за певні курси та середню відвідуваність занять. За допомогою програмного забезпечення SPSS 16 було ідентифіковано три кластери студентів: високоосвічені студенти (45,74%), студенти середнього рівня (33,33%) та студенти з низькими показниками успішності (20,92%).

Criteria	Cluester		
	1	2	3
Zscore(Parent's Occupation)	.14542	1.2534	-.67893
Zscore(Gender)	-.35093	.25977	.13692
Zscore(Origin)	.14542	1.2534	-.67893
Zscore GPA	-.03815	-.02640	.03986
Zscore(OptimizatioGrade)	-1.262042	.61232	.63838
Zscore(PPPGrade)	-.82642	.61750	.31969
Zscore(Absence)	-.77936	.21293	.47072

Рисунок 1.5 – Результати кластерізації

Введення розглядає важливість розуміння умов студентів для планування програм підвищення успішності, а також контекст вищої освіти як сфери послуг зі студентами як основними споживачами. Аналізується академічна інформаційна система та середні показники успішності студентів, що вказують на потребу в покращенні.

Методологія дослідження включає попереднє вивчення для аналізу проблеми та збір вторинних даних з бази даних університету. Вибірка включає 20% від загальної кількості студентів з різними класифікаційними змінними та кластерами, зосередженими на академічному профілі та профілі активності студентів.

Результати та обговорення демонструють, що використання алгоритму "K-Means Cluster" дозволяє ідентифікувати різні категорії студентів за їх академічною успішністю та активністю, що може бути корисним для розробки цільових програм підвищення успішності.

У висновку підкреслюється, що дані з 306 студентів відділення промислової інженерії Ісламського університету Індонезії сформували три різні кластери, кожен з яких має свої характеристики. Найбільший кластер включає групу розумних та активних студентів (45,75%), за ними йдуть студенти з нижче середнього рівня здібностей (33,33%)

Наступна стаття, озаглавлена "Predicting Students' Final GPA Using Decision Trees: A Case Study", авторів Mashael A. Al-Barrak та Muna Al-Razgan, опублікована в 2016 році, присвячена застосуванню методів освітнього аналізу даних для прогнозування кінцевого середнього балу (GPA) студентів на основі їхніх оцінок у попередніх курсах. Автори зосереджуються на дослідженні даних студентів відділення інформаційних технологій Королівського університету Саудівської Аравії, використовуючи алгоритм класифікаційного дерева J48 для виявлення правил класифікації, які дозволяють прогнозувати кінцевий GPA студентів.

У вступі розглядається зростання обсягів освітніх даних і потреба в їх аналізі за допомогою методів освітнього аналізу даних (EDM), що дозволяє адаптувати новітні методики навчання та покращити освітній процес.

Основна мета дослідження - вивчення взаємозв'язку між оцінками студентів у обов'язкових курсах та їх кінцевим GPA, що дозволяє ідентифікувати найважливіші предмети в навчальному плані студентів, які мають найбільший вплив на їхню кінцеву успішність.

Автори зібрали та підготували дані транскриптів студентів, що випускалися з факультету комп'ютерних наук університету в 2012 році, обробили дані, використовуючи інструментарій WEKA, та застосували алгоритм J48 для побудови моделі класифікації. Аналіз результатів дозволив визначити ключові курси, які мають найбільший вплив на кінцевий GPA студентів.

Semester	Course Name
3	Java1 (the only specialized course)
4	Database Principles
5	Software Engineering 1
6	Information security
7	Computer Ethics
8	Project 2

Рисунок 1.6 – Результат аналізу ключових курсів

У висновках підкреслюється потенціал застосування аналізу даних у вищій освіті для підвищення успішності студентів та раннього виявлення факторів, що впливають на їхню кінцеву оцінку. Автори пропонують подальше розширення дослідження шляхом включення елективних та загальноосвітніх курсів для отримання більш точних результатів та застосування інших технік аналізу даних, таких як нейронні мережі та кластеризація.

Висновки до розділу 1

У висновку до даного розділу можна відзначити, що сучасні дослідження та публікації в галузі освітньої аналітики та машинного навчання відкривають нові горизонти для підвищення ефективності навчального процесу та прогнозування успішності студентів. Аналіз великих даних та застосування алгоритмів машинного навчання, таких як штучна нейронна мережа, наївний Байєс, дерево рішень та SVM, демонструють високу потенційну здатність до ідентифікації ранніх ознак труднощів у навчанні, а також до адаптації освітніх стратегій відповідно до індивідуальних потреб студентів. Виявлено, що окрім академічних показників, важливу роль у прогнозуванні успішності відіграють особистісні риси, мотиваційні чинники, стратегії саморегуляції навчання, підходи до навчання та контекстуальні психосоціальні фактори. Це підкреслює необхідність комплексного підходу до оцінки та підтримки студентів, який включає не лише інтелектуальний, але й емоційний, соціальний та психологічний розвиток.

2 ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТАЛЬНІ ЗАСОБИ РОЗРОБКИ СИСТЕМИ

2.1 Мова програмування Python

Python є мовою програмування загального призначення високого рівня, що знайшла широке застосування. Розроблена Гвідо ван Россумом у 1991 році та підтримувана Python Software Foundation, мова була спрямована на підвищення читабельності коду, дозволяючи програмістам виражати концепції в компактнішій формі порівняно з такими мовами, як Java, C++ та C. Розробка Python розпочалася в кінці 1980-х років як хобі Гвідо ван Россума під час роботи в Centrum Wiskunde & Informatica (CWI) у Нідерландах, маючи на меті усунення недоліків мови програмування ABC. Ван Россум використав синтаксис та деякі позитивні аспекти ABC, прагнучи створити мову, що вирішувала б проблеми попередниці. Назва Python була надихнута телешоу "Літаючий цирк Монті Пайтона", відображаючи бажання Россума мати унікальну та запам'ятовувану назву. Ван Россум обіймав посаду "Довічного диктатора" (BDFL) до своєї відставки у 2018 році, працюючи в таких компаніях, як Google та Dropbox.

Рисунок 2.1 – Логотип Python та PyCharm

Python набув широкої популярності завдяки своїй елегантності, простоті синтаксису та багатому набору можливостей, що сприяло його адаптації

провідними технологічними організаціями, включаючи Dropbox, Google, Quora, Mozilla, Hewlett-Packard, Qualcomm, IBM та Cisco. Версії Python 2.x і 3.x конкурують за увагу розробників, кожна з яких має своїх прихильників та специфічні використання.продуктивність розробників. Коли він був випущений, він мав більш ніж достатньо можливостей для надання класам успадкування, обробки винятків кількох основних типів даних і функцій.

На честь 30-річчя Python, на конференції pycon22 було представлено нову функцію pyscript, що дозволяє виконувати код Python безпосередньо у веб-браузері, аналогічно до JavaScript. Цей крок відкриває нові перспективи для використання Python у веб-розробці. Незважаючи на технічний прогрес, Python зберігає свою унікальність та привабливість для широкого кола користувачів, що демонструється високим інтересом до мови в порівнянні з популярними медійними персонами.

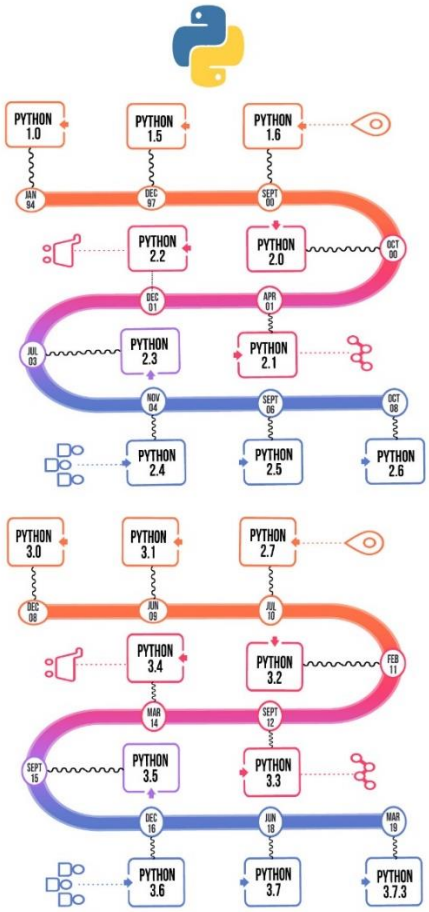

Рисунок 2.2 – Візуалізація версій Python

Завдяки своїй універсальності та доступності, Python став інструментом вибору для широкого спектру застосувань, включаючи веб-розробку, наукові обчислення, аналіз даних, штучний інтелект, автоматизацію та багато іншого. Його гнучкість та інтеграція з іншими технологіями роблять його ідеальним вибором для стартапів та великих корпорацій, які прагнуть швидко розвивати інноваційні рішення. Мова програмування Python продовжує розвиватися, підтримуючи зростаючу спільноту розробників та користувачів, які вносять свій вклад у її вдосконалення та розширення можливостей. Активний внесок спільноти, велика кількість бібліотек та фреймворків, а також відкритість і доступність навчальних ресурсів забезпечують Python стабільне місце на передньому краї технологічного прогресу. Отже, Python не лише зберігає свою актуальність, але й продовжує бути в авангарді інновацій у сфері програмування.

2.2 Фреймворк Scikit-learn

Scikit-learn є однією з найпопулярніших бібліотек для машинного навчання в середовищі Python і забезпечує широкий набір інструментів для класифікації, регресії, кластеризації, вимірювання якості моделей та інші завдання машинного навчання.

Рисунок 2.3 – Логотип scikit-learn

Scikit-learn спрощує процес створення та навчання моделей машинного навчання завдяки консистентному та легко зрозумілому API. Вона надає доступ до різноманітних алгоритмів, таких як метод опорних векторів, рішучі дерева, наївний баєсівський класифікатор, k-середніх, лінійна та логістична регресія та багато інших. Однією з сильних сторін Scikit-learn є його гнучкість та можливість використання в різноманітних сценаріях. Вона підтримує векторизацію та оптимізацію обчислень за допомогою бібліотеки NumPy, що робить її ефективною для роботи з великими обсягами даних.

Scikit-learn включає в себе утиліти для обробки даних, такі як підготовка даних, вибір ознак, а також інструменти для оцінки та налаштування параметрів моделей. Це сприяє створенню повноцінних пайплайнів для обробки та аналізу даних. За своєю природою вона є відкритим програмним забезпеченням та активно підтримується спільнотою. Це дозволяє користувачам легко долучати власні алгоритми та розширювати функціональність бібліотеки для вирішення конкретних завдань машинного навчання.

Scikit-learn також надає інструменти для оцінки результатів моделей за допомогою різних метрик якості, таких як точність, відзив, точність та F1-міра. Це важливо для об'єктивного порівняння та вибору найкращої моделі для конкретного завдання.

Бібліотека також включає в себе інструменти для роботи з текстовими даними, використовуючи методи векторизації та обробки тексту. Це робить її потужним інструментом для вирішення завдань аналізу тексту та природної мови. Крім того, Scikit-learn активно використовується в навчанні та дослідженнях, оскільки вона дозволяє швидко виконувати експерименти з різними моделями та алгоритмами, а також розгортати їх у виробничих середовищах.

Загалом, вона є важливим інструментом для вивчення, розуміння та застосування методів машинного навчання в різних областях, від науки про дані та дослідження до вирішення прикладних завдань в індустрії та бізнесі. Основна перевага Scikit-learn полягає в його гнучкості та застосовності в різних сценаріях.

Вона використовує векторизацію та оптимізацію обчислень за допомогою NumPy, що робить її ефективною для обробки великих обсягів даних. Бібліотека також включає в себе інструменти для оцінки результатів моделей, вирішення завдань обробки текстових даних та взаємодії з іншими бібліотеками в екосистемі Python.

Scikit-learn не тільки дозволяє вивчати та розуміти методи машинного навчання, але і є практичним інструментом для розв'язання прикладних завдань в галузі науки про дані, дослідження та виробництва. Завдяки своїй відкритій природі та підтримці спільноти, вона залишається активно розвиваючоюся та використовується в різних галузях.

```python
from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Завантаження даних для класифікації облич (приклад)
# X - зображення, y - клас (облич)

# Розділення на тренувальний та тестовий набори
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2

# Ініціалізація та навчання моделі SVM
clf = svm.SVC(kernel='linear')
clf.fit(X_train, y_train)

# Прогнозування на тестовому наборі
y_pred = clf.predict(X_test)

# Оцінка точності моделі
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

Рисунок 2.4 – Приклад застосування

Цей код демонструє використання Scikit-learn для створення моделі SVM для класифікації облич на основі зображень

2.3 Бібліотека Pandas

В наукових дослідженнях, що займаються аналізом та обробкою даних, часто використовується програмне забезпечення Pandas - бібліотека мови програмування Python, призначена для швидкого, потужного, гнучкого та зручного аналізу та маніпулювання даними. Вона надає високопродуктивні, легкодоступні структури даних та інструменти аналізу, здатні ефективно вирішувати складні завдання обробки даних та аналітики.

Рисунок 2.5 – Логотип Pandas

Pandas включає в себе такі ключові структури даних, як DataFrame та Series, що забезпечують можливість виконання операцій з різними типами даних - як числовими, так і категоріальними. DataFrame представляє собою двовимірну мутабельну табличну структуру з потенційно різнорідними стовпцями, тоді як Series - це одновимірний масив, індексований масив даних.

Основні функціональні можливості Pandas включають завантаження даних з різноманітних файлових форматів, таких як CSV, Excel та баз даних SQL; маніпулювання та зміну даних з можливістю злиття, групування, переіндексації та вибірки; статистичний аналіз; та здатність вирішувати як прості, так і складні запити та операції з даними.

Pandas інтегрується з багатьма іншими науковими та інженерними бібліотеками Python, такими як NumPy та SciPy, а також із бібліотеками візуалізації даних, як Matplotlib, дозволяючи проводити комплексний аналіз даних від їх обробки до візуалізації результатів. Pandas широко використовується в академічних та комерційних сферах, де потрібно швидке та ефективне вирішення завдань з обробки великих обсягів даних, забезпечуючи високу продуктивність та гнучкість для дослідницької роботи.

2.3 Алгоритм LinearRegression

Лінійна регресія — це алгоритм, який забезпечує лінійний зв’язок між незалежною змінною та залежною змінною для прогнозування результатів майбутніх подій. Це статистичний метод, який використовується в науці про дані та машинному навчанні для прогнозного аналізу.

Незалежна змінна також є прогностичною або пояснювальною змінною, яка залишається незмінною через зміну інших змінних. Однак залежна змінна змінюється разом із коливаннями незалежної змінної. Регресійна модель передбачає значення залежної змінної, яка є змінною відповіді або результату, що аналізується або вивчається.

Таким чином, лінійна регресія — це контрольований алгоритм навчання, який моделює математичний зв’язок між змінними та робить прогнози для неперервних або числових змінних, таких як продажі, зарплата, вік, ціна продукту тощо.

Цей метод аналізу є перевагою, коли в даних доступні принаймні дві змінні, як це спостерігається при прогнозуванні фондового ринку, управлінні портфелем, науковому аналізі тощо.

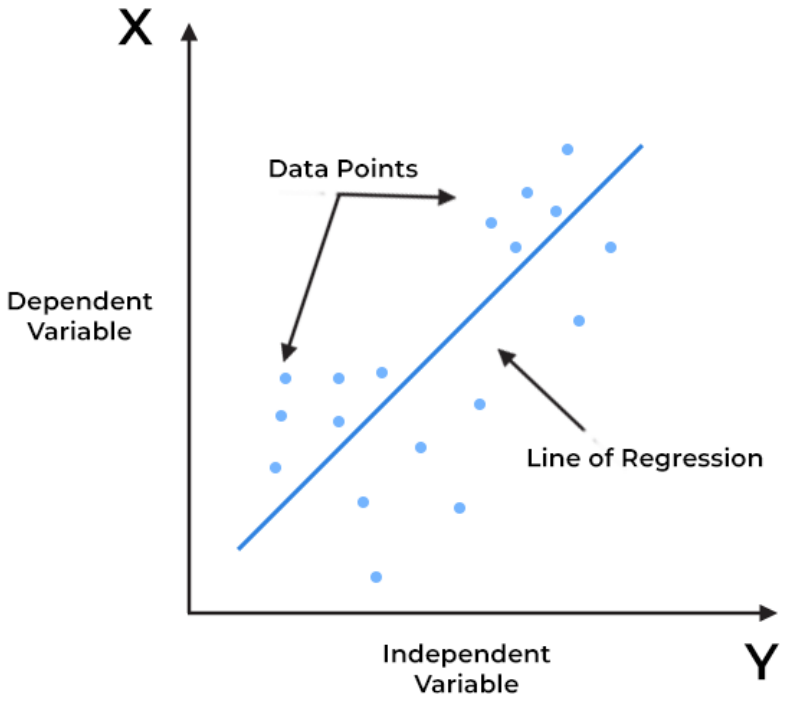

Рисунок 2.6 – Лінійна регресія

Тут будується лінія для заданих точок даних, які відповідним чином відповідають усім проблемам. Тому її називають «найкращою лінією». Мета алгоритму лінійної регресії полягає в тому, щоб знайти лінію, яка найкраще підходить, як показано на малюнку вище.

Лінійна регресія є популярним статистичним інструментом, який використовується в науці про дані, завдяки кільком перевагам, як-от:

Легка реалізація. Модель лінійної регресії обчислювально проста для реалізації, оскільки вона не вимагає великих інженерних витрат ні перед запуском моделі, ні під час її обслуговування.

Інтерпретованість. На відміну від інших моделей глибокого навчання (нейронні мережі), лінійна регресія є відносно простою. Як наслідок, цей алгоритм випереджає моделі чорної скриньки, які не можуть обґрунтувати, яка вхідна змінна викликає зміну вихідної змінної.

Масштабованість. Лінійна регресія не є важкою для обчислень і, отже, добре підходить у випадках, коли масштабування є важливим. Наприклад, модель може добре масштабуватися щодо збільшення обсягу даних (великі дані).

Оптимальний для онлайн налаштувань. Простота обчислення цих алгоритмів дозволяє використовувати їх в онлайн-налаштуваннях. Модель можна навчати та перенавчати з кожним новим прикладом, щоб генерувати прогнози в режимі реального часу, на відміну від нейронних мереж або опорних векторних машин, які є важкими з точки зору обчислень і вимагають багато обчислювальних ресурсів і значного часу очікування для повторного навчання на новому наборі даних. Усі ці фактори роблять такі інтенсивні обчислювальні моделі дорогими та непридатними для програм реального часу.

Наведені вище функції підкреслюють, чому лінійна регресія є популярною моделлю для вирішення реальних проблем машинного навчання.

2.4 Алгоритм Lasso

Регресія LASSO, також відома як регулярізація L1, є популярним методом, який використовується в статистичному моделюванні та машинному навчанні для оцінки зв’язків між змінними та прогнозування. LASSO означає оператор найменшого абсолютного скорочення та вибору.

Основна мета регресії LASSO — знайти баланс між простотою та точністю моделі. Це досягається шляхом додавання штрафу до традиційної моделі лінійної регресії, яка заохочує розріджені рішення, де деякі коефіцієнти змушені дорівнювати точно нулю. Ця функція робить LASSO особливо корисним для вибору функцій, оскільки він може автоматично ідентифікувати та відкидати нерелевантні або надлишкові змінні.

Регресія ласо є технікою регуляризації. Він використовується замість методів регресії для більш точного прогнозування. У цій моделі використовується усадка. Скорочення – це коли значення даних зменшуються до центральної точки як середнього значення. Процедура ласо заохочує прості, розріджені моделі (тобто

моделі з меншою кількістю параметрів). Цей окремий тип регресії добре підходить для моделей, що демонструють високий рівень мультиколінеарності, або коли потрібно автоматизувати певні частини вибору моделі, як-от вибір змінної/усунення параметрів.

Lasso Regression використовує техніку регуляризації L1. Він використовується, коли у нас є більше функцій, оскільки він автоматично виконує вибір функцій.

Модель лінійної регресії : регресія LASSO починається зі стандартної моделі лінійної регресії, яка передбачає лінійний зв’язок між незалежними змінними (ознаками) і залежною змінною (цільовою).

Регулярізація L1 : регресія LASSO вводить додатковий штрафний термін на основі абсолютних значень коефіцієнтів. Термін регулярізації L1 – це сума абсолютних значень коефіцієнтів.

Цей штрафний термін регуляризації L1 має вирішальне значення у формуванні кінцевої моделі регресії LASSO. Він не тільки сприяє зменшенню магнітуди коефіцієнтів, але й може звести деякі з них до нуля. Це означає, що модель LASSO може слугувати не лише для прогнозування, але й як метод відбору ознак. Таким чином, ознаки, коефіцієнти яких зведені до нуля, вважаються неважливими для прогнозування цільової змінної і, отже, виключаються з моделі. Ця здатність до "відбору ознак" робить LASSO особливо корисним у ситуаціях, де даних багато, але не всі вони однаково важливі для прогнозування. Отже, регресія LASSO не лише покращує точність моделі за рахунок уникнення перенавчання, але й робить модель більш інтерпретованою, спрощуючи її структуру через виключення неважливих ознак.

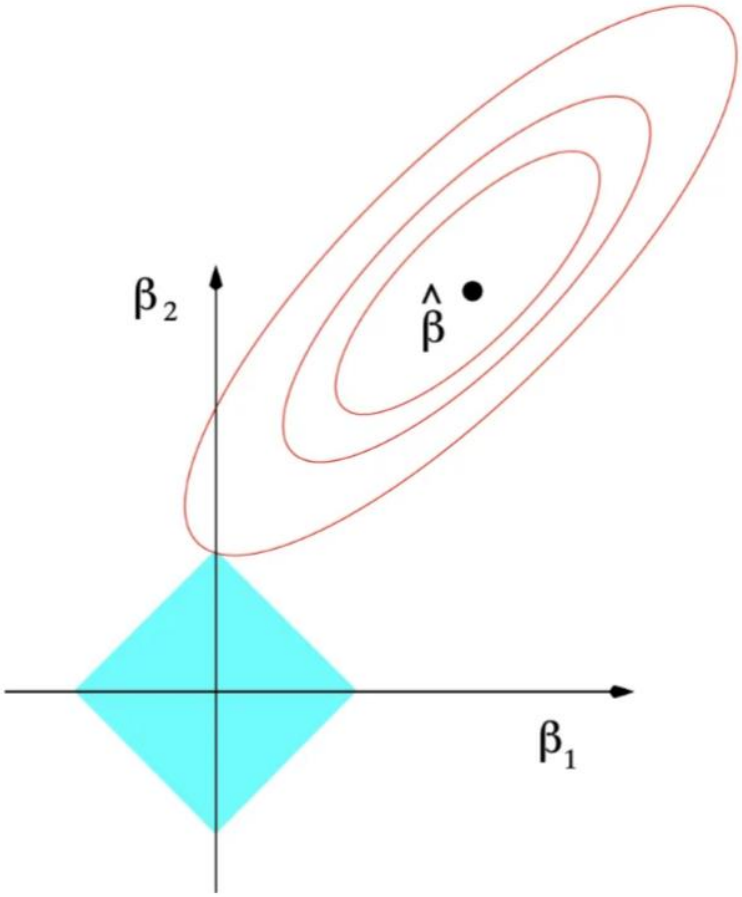

Рисунок 2.7 – Регулярізація L1

Цільова функція : мета регресії LASSO полягає в тому, щоб знайти значення коефіцієнтів, які мінімізують суму квадратів різниць між прогнозованими значеннями та фактичними значеннями, а також мінімізують термін регулярізації L1: код makefileCopy Minimize: RSS + L_1 Де: RSS це залишкова сума квадратів, яка вимірює похибку між прогнозованими значеннями та фактичними значеннями.

Коефіцієнти скорочення: шляхом додавання члена регуляризації L1 регресія LASSO може зменшити коефіцієнти до нуля. Коли λ є достатньо великим, деякі коефіцієнти зводяться точно до нуля. Ця властивість LASSO робить його корисним для вибору ознак, оскільки змінні з нульовими коефіцієнтами ефективно видаляються з моделі.

Параметр налаштування λ : вибір параметра регуляризації λ є вирішальним у регресії LASSO. Більше λ значення збільшує кількість регуляризації, що призводить до того, що більше коефіцієнтів зсуваються до нуля. І навпаки, менше λзначення зменшує ефект регуляризації, дозволяючи більшій кількості змінних мати ненульові коефіцієнти.

Підгонка моделі: щоб оцінити коефіцієнти регресії LASSO, використовується алгоритм оптимізації для мінімізації цільової функції. Зазвичай використовується спуск координат, який ітеративно оновлює кожен коефіцієнт, утримуючи інші фіксованими.

2.5 Алгоритм Ridge

Ridge-регресія, також відома як регулярізація Тихонова, — це техніка, яка використовується для аналізу даних множинної регресії, які страждають від мультиколінеарності. Коли має місце мультиколінеарність, оцінки методом найменших квадратів є незміщеними, але їх дисперсії великі, і вони можуть бути далекими від справжнього значення. Додаючи ступінь зміщення до оцінок регресії, гребенева регресія зменшує стандартні помилки.

Основна ідея хребтової регресії полягає в тому, щоб знайти нову лінію, яка також не відповідає даним навчання. Іншими словами, це вносить невелику кількість упередженості в те, як новий рядок відповідає даним. Цей компроміс між зміщенням і дисперсією є тим, що дозволяє хребтовій регресії досягати кращих довгострокових прогнозів.

Ридж-регресія вирішує проблему мультиколінеарності (незалежні змінні, які сильно корельовані) у моделях лінійної регресії. Мультиколінеарність може призвести до викривлених або завищених оцінок коефіцієнтів регресії, що може вплинути на інтерпретацію моделі. Регресія Ріджа вирішує цю проблему шляхом додавання штрафного члена до звичайного рівняння найменших квадратів (OLS).

Вибір правильного значення для параметра регуляризації (лямбда) є критичним у гребеневій регресії. Якщо лямбда занадто велика, модель може стати

занадто упередженою та не відповідати даним. І навпаки, якщо лямбда занадто мала, модель поводитиметься подібно до стандартної моделі лінійної регресії та переобладнати дані.

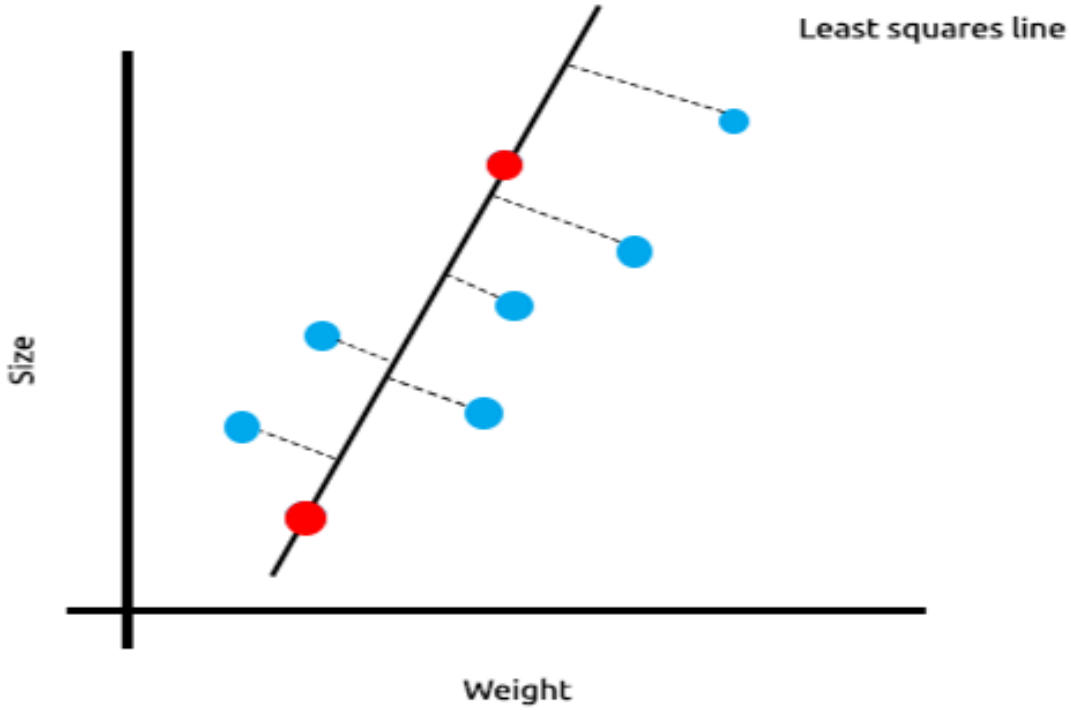

Рисунок 2.8 – Ridge-регресія

Одним із поширених методів вибору лямбда є перехресна перевірка. Перехресна перевірка передбачає поділ набору даних на декілька підмножин, підгонку моделі до деяких підмножин, а інші підмножини використовують як дані перевірки для обчислення показника ефективності моделі, наприклад середньоквадратичної помилки (MSE). Зазвичай вибирається значення лямбда , яке мінімізує помилку перехресної перевірки.

2.6 Алгоритм DecisionTree

Дерево рішень — це непараметричний контрольований алгоритм навчання для задач класифікації та регресії . Він має ієрархічну структуру дерева, що

складається з кореневого вузла, гілок, внутрішніх вузлів і листових вузлів. Дерева рішень використовуються для завдань класифікації та регресії, забезпечуючи прості для розуміння моделі..Ця алгоритмічна модель використовує оператори умовного керування та є непараметричним, контрольованим навчанням, корисним як для завдань класифікації, так і для регресії. Деревоподібна структура складається з кореневого вузла, гілок, внутрішніх вузлів і листових вузлів, які утворюють ієрархічну деревоподібну структуру.

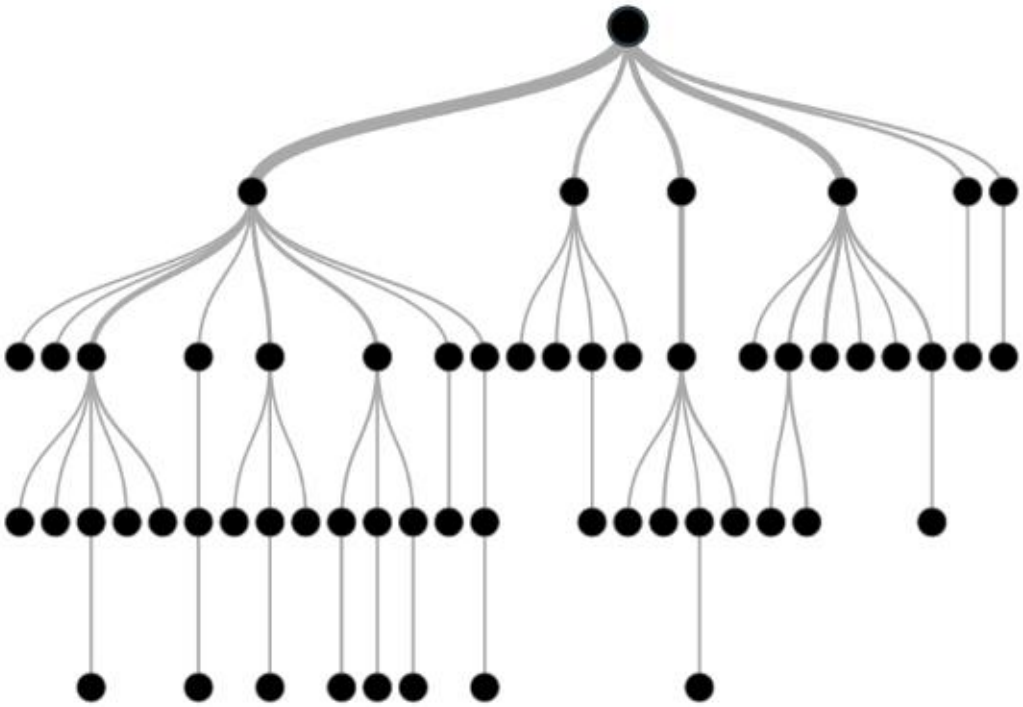

Рисунок 2.9 – Дерево рішень

Це інструмент, який має програми, що охоплюють кілька різних областей. Дерева рішень можна використовувати для класифікації, а також для задач регресії. Сама назва говорить про те, що він використовує блок-схему, подібну до деревоподібної структури, щоб показати прогнози, які є результатом серії розділень на основі функцій. Він починається з кореневого вузла і закінчується рішенням, прийнятим листям.

2.7 Алгоритм RandomForest

Алгоритми випадкового лісу мають три основні гіперпараметри, які необхідно встановити перед навчанням. До них належать розмір вузла, кількість дерев і кількість відібраних функцій. Звідти класифікатор випадкового лісу можна використовувати для вирішення проблем регресії або класифікації.

Алгоритм випадкового лісу складається з набору дерев рішень, і кожне дерево в ансамблі складається із вибірки даних, отриманої з навчального набору із заміною, яка називається початковою вибіркою. З цієї навчальної вибірки одна третина відкладена як тестові дані, відомі як вихідна (oob) вибірка, до якої ми повернемося пізніше. Інший випадок випадковості потім вводиться через пакетування функцій, додаючи більше різноманітності до набору даних і зменшуючи кореляцію між деревами рішень. Залежно від типу проблеми визначення прогнозу буде різним. Для завдання регресії окремі дерева рішень будуть усереднені, а для завдання класифікації більшість голосів, тобто найпоширеніша категоріальна змінна, дасть прогнозований клас. Нарешті, зразок oob використовується для перехресної перевірки, завершуючи це передбачення.

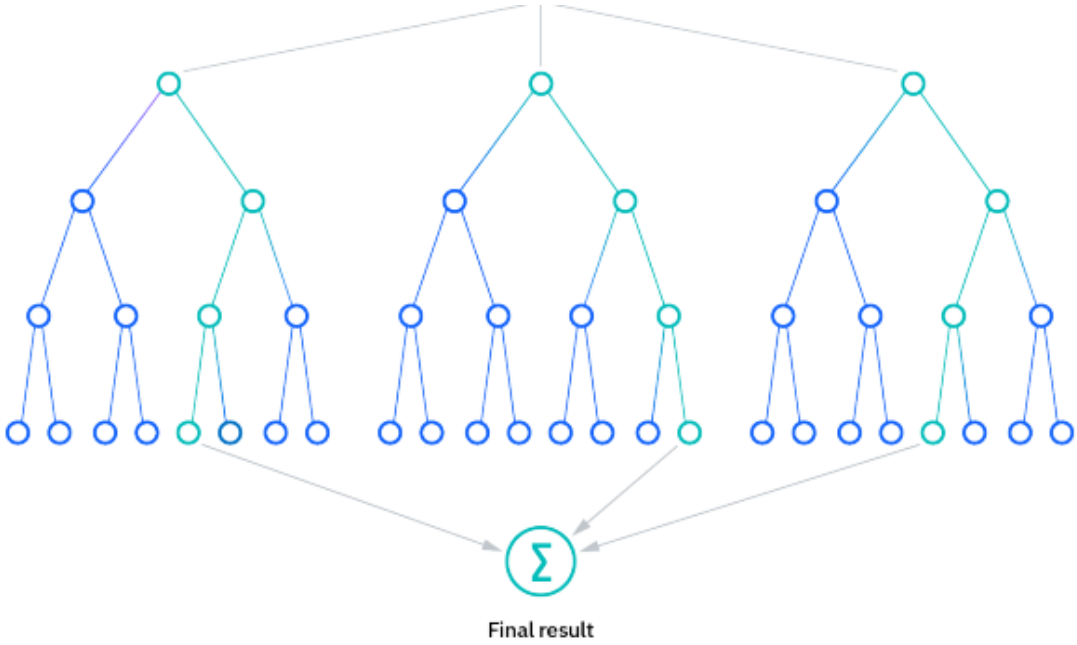

Рисунок 2.10 – Алгоритм випадкового лісу

Алгоритм випадкового лісу має ряд ключових переваг і труднощів, коли він використовується для задач класифікації або регресії.

Зменшення ризику переобладнання, дерева рішень мають ризик переобладнання, оскільки вони, як правило, щільно підходять для всіх зразків у навчальних даних. Однак, коли у випадковому лісі існує значна кількість дерев рішень, класифікатор не буде перебільшувати модель, оскільки усереднення некорельованих дерев зменшує загальну дисперсію та помилку прогнозу.

Оскільки випадковий ліс може обробляти завдання як регресії, так і класифікації з високим ступенем точності, це популярний метод серед дослідників даних. Розміщення функцій також робить класифікатор випадкового лісу ефективним інструментом для оцінки відсутніх значень, оскільки він підтримує точність, коли частина даних відсутня.

Випадковий ліс дозволяє легко оцінити важливість змінної або внесок у модель. Є кілька способів оцінити важливість функції. Важливість Джіні та середнє зменшення домішок (MDI) зазвичай використовуються для вимірювання того, наскільки зменшується точність моделі, коли дана змінна виключається. Однак важливість перестановки, також відома як точність зниження середнього значення (MDA), є ще одним показником важливості. MDA визначає середнє зниження точності шляхом випадкової перестановки значень ознак у зразках oob.

Основні недоліки є, оскільки випадкові алгоритми лісу можуть обробляти великі набори даних, вони можуть надавати точніші прогнози, але можуть повільно обробляти дані, оскільки вони обчислюють дані для кожного окремого дерева рішень.

Вимагає більше ресурсів, оскільки випадкові ліси обробляють більші набори даних, їм знадобиться більше ресурсів для зберігання цих даних.

Більш складний: передбачення окремого дерева рішень легше інтерпретувати порівняно з лісом із них.

Висновки до розділу 2

У даному розділі проведено детальний аналіз технологічного інструментарію, який знаходить застосування у сфері машинного навчання, з особливим акцентом на мову програмування Python та асоційовані з нею бібліотеки. Мова Python демонструє високу ефективність у реалізації алгоритмів машинного навчання, що підтверджується її широким використанням серед науковців та розробників. Бібліотека Scikit-learn, яка визнана однією з найбільш універсальних у цій області, забезпечує необхідні інструменти для ефективної роботи з даними та моделями.

Детально розглянуто алгоритми лінійної регресії, Lasso та Ridge, які забезпечують основу для розуміння принципів регуляризації та методів боротьби з явищем перенавчання. Описані особливості, переваги та потенціал застосування цих методів дозволяють глибше осягнути механізми регуляризації та їх значення для побудови точних прогностичних моделей. Аналіз моделей DecisionTree та RandomForest розкриває можливості застосування цих методів для роботи з складними нелінійними даними та моделювання різноманітних залежностей у досліджуваних наборах даних.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ВПРОВАДЖЕННЯ СИСТЕМИ

3.1 Огляд датасету

У 2023 році в Сполучених Штатах Америки був створений унікальний національний датасет, який мав на меті дослідити різноманітні аспекти навчального процесу студентів. Цей датасет був складений з відкритих даних, зібраних з різних джерел, таких як навчальні установи, державні агентства з освіти та організації, що здійснюють моніторинг освіти.

Одним із ключових джерел інформації були шкільні адміністрації, які забезпечували дані про студентів, такі як стать, расу/етнічну належність, рівень освіти батьків, участь в підготовчих курсах та результати студентів з математики, читання та письма. Ці дані були анонімізовані та об'єднані в один датасет для подальшого вивчення та аналізу.

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group D	some college	standard	completed	59	70	78
1	male	group D	associate's degree	standard	none	96	93	87
2	female	group D	some college	free/reduced	none	57	76	77
3	male	group B	some college	free/reduced	none	70	70	63
4	female	group D	associate's degree	standard	none	83	85	86
...	...	...	...	...	...	...	...	...
995	male	group C	some college	standard	none	77	77	71
996	male	group C	some college	standard	none	80	66	66
997	female	group A	high school	standard	completed	67	86	86
998	male	group E	high school	standard	none	80	72	62
999	male	group D	high school	standard	none	58	47	45

Рисунок 3.1 – Огляд датасету

3.2 Реалізація системи

Спочатку імпортуються необхідні бібліотеки, такі як NumPy, Pandas, Matplotlib, Seaborn, Scikit-learn і SciPy. Це необхідно для зручної роботи з даними та візуалізації результатів. Далі вимикається виведення попереджень для поліпшення читабельності виводу.

Після цього імпортуються різні моделі машинного навчання, такі як лінійна регресія, дерева рішень, опорні вектори, K-ближайших сусідів та інші. Ці моделі можуть бути використані для аналізу та прогнозування даних.

Також імпортується метрика R2 для оцінки точності моделей під час їхньої продуктивності. У завершення виводиться повідомлення "All dependencies are imported.", підтверджуючи успішний імпорт необхідних бібліотек. Такий підхід створює необхідне середовище для подальшого аналізу даних та побудови моделей машинного навчання.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

from IPython.display import display

warnings.filterwarnings('ignore')
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import scipy.stats as stats
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor, ExtraTreesRegressor
from sklearn.metrics import r2_score
print("All dependencies are imported.")
```

Рисунок 3.2 – Імпорт бібліотек

Спочатку відбувається зчитування даних з файлу 'exams.csv' за допомогою функції pd.read_csv(), і результат зберігається у змінній per. Потім виводиться повідомлення "Student performance dataset:", за яким слідує виведення самого датасету за допомогою display(per).

Далі виводиться повідомлення "Dataset information:", а після нього виводиться інформація про датасет за допомогою display(per.info()).

На завершення виводить повідомлення "Null counts:" та відображає кількість нульових значень для кожного стовпця в датасеті за допомогою display(per.isnull().sum()).

У виведенні інформації про датасет було внесено зміни відповідно до зауважень. Виведення per.info() більше не виводиться в неповний текст, а замість цього використовується display(per.info()) для кращої читабельності.

```
per=pd.read_csv('exams.csv')
print("Student performance dataset:")
display(per)

print("Dataset information:")
display(per.info())

print("Null counts:")
display(per.isnull().sum())
```

Рисунок 3.3 – Лістинг коду

```
All dependencies are imported.
Student performance dataset:
     gender race/ethnicity  ... reading score writing score
0    female        group D  ...            70            78
1      male        group D  ...            93            87
2    female        group D  ...            76            77
3      male        group B  ...            70            63
4    female        group D  ...            85            86
..      ...            ...  ...           ...           ...
995    male        group C  ...            77            71
996    male        group C  ...            66            66
997  female        group A  ...            86            86
998    male        group E  ...            72            62
999    male        group D  ...            47            45

[1000 rows x 8 columns]
```

Рисунок 3.4 – Виведення інформації про датасет

```
Dataset information:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   gender                       1000 non-null   object
 1   race/ethnicity               1000 non-null   object
 2   parental level of education  1000 non-null   object
 3   lunch                        1000 non-null   object
 4   test preparation course      1000 non-null   object
 5   math score                   1000 non-null   int64
 6   reading score                1000 non-null   int64
 7   writing score                1000 non-null   int64
dtypes: int64(3), object(5)
memory usage: 62.6+ KB
```

Рисунок 3.5 – Вивід інформації про датасет

```
Null counts:
gender                       0
race/ethnicity               0
parental level of education  0
lunch                        0
test preparation course      0
math score                   0
reading score                0
writing score                0
dtype: int64
```

Рисунок 3.6 – Вивід інформації про нульові значення

Спочатку створюється матриця графіків з розміром (3, 2) за допомогою plt.subplots(3, 2, figsize=(20, 20)).

Далі за допомогою циклу for та enumerate виконується прохід по кожній категоріальній змінній із списку cat_cols. Для кожної змінної будується стовбчатий графік за допомогою sns.countplot(x=col, data=per, ax=axes[r, c]), де r та c обчислюються для визначення рядка і стовпця в матриці графіків.

Додатково, для кожного графіка виводяться літерники на верхніх краях стовбців з використанням axes[r, c].bar_label(container, label_type="edge"), що дозволяє побачити точну кількість спостережень у кожній категорії.

Також встановлюється належна назва для вісі Y, назва графіку, та виконується налаштування для кращої читабельності графіків, такі як обертання позначень осі X на 90 градусів.

У кінці використовується plt.subplots_adjust(hspace=0.5) для налаштування відстані між графіками та fig.text() для виведення заголовка.

У разі, якщо кількість категоріальних змінних непарна, останній графік видаляється для збереження зручності виведення. Нарешті, виводиться побудована матриця графіків за допомогою plt.show().

```
cat_cols = ["gender", "race/ethnicity", "parental level of education", "lunch", "test preparation course"]
fig, axes = plt.subplots(3, 2, figsize=(20, 20))
for i, col in enumerate(cat_cols):
    r = i // 2
    c = i % 2
    sns.countplot(x=col, data=per, ax=axes[r, c])
    for container in axes[r, c].containers:
        axes[r, c].bar_label(container, label_type="edge")
    axes[r, c].set_ylabel(f"{col}")
    axes[r, c].set_xlabel("Frequency")
    axes[r, c].set_xticklabels(axes[r,c].get_xticklabels(),rotation="vertical")
    axes[r, c].set_title(f"{cat_cols[i]} Distribution")
if len(cat_cols) % 2 != 0:
    fig.delaxes(axes[-1, -1])
plt.subplots_adjust(hspace=0.5)
fig.text(0.5,0.9,"Few Categorical Columns Distribution (Bar Graph)", va="center", ha="center", fontsize=14)
plt.show()
```

Рисунок 3.7 – Лістинг коду

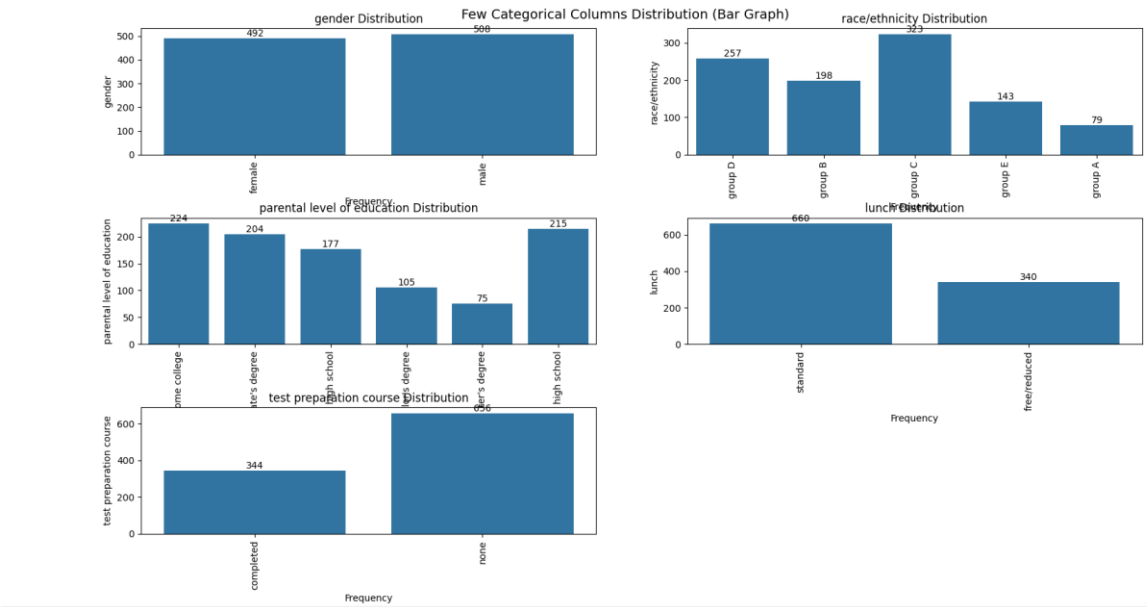

Рисунок 3.8 – Матриця графіків

Створюється матриця графіків з розміром (3, 2) за допомогою plt.subplots(3, 2, figsize=(10, 10)).

Далі за допомогою циклу for та enumerate виконується прохід по кожній категоріальній змінній із списку cat_cols. Для кожної змінної будується кругова діаграма за допомогою axes[r, c].pie(counts, labels=counts.index, autopct='%1.1f%%', startangle=90).

Також встановлюється заголовок графіка за допомого axes[r, c].set_title(f"{col} Distribution").

У разі, якщо кількість категоріальних змінних непарна, останній графік видаляється для збереження зручності виведення. На завершення використовується plt.subplots_adjust(hspace=0.5, wspace=0.5) для налаштування відстані між графіками та fig.text() для виведення заголовка.

Нарешті, виводиться побудована матриця графіків за допомогою plt.show().

```
cat_cols = ["gender", "race/ethnicity", "parental level of education", "lunch", "test preparation course"]
fig, axes = plt.subplots(3, 2, figsize=(10, 10))
for i, col in enumerate(cat_cols):
    r = i // 2
    c = i % 2
    counts = per[col].value_counts()
    axes[r, c].pie(counts, labels=counts.index, autopct='%1.1f%%', startangle=90)
    axes[r, c].set_title(f"{col} Distribution")
if len(cat_cols) % 2 != 0:
    fig.delaxes(axes[-1, -1])
plt.subplots_adjust(hspace=0.5, wspace=0.5)
fig.text(0.5,0.95,"Few Categorical Columns Distribution (Pie Chart)",va="center", ha="center", fontsize=14)
plt.show()
```

Рисунок 3.9 – Лістинг коду

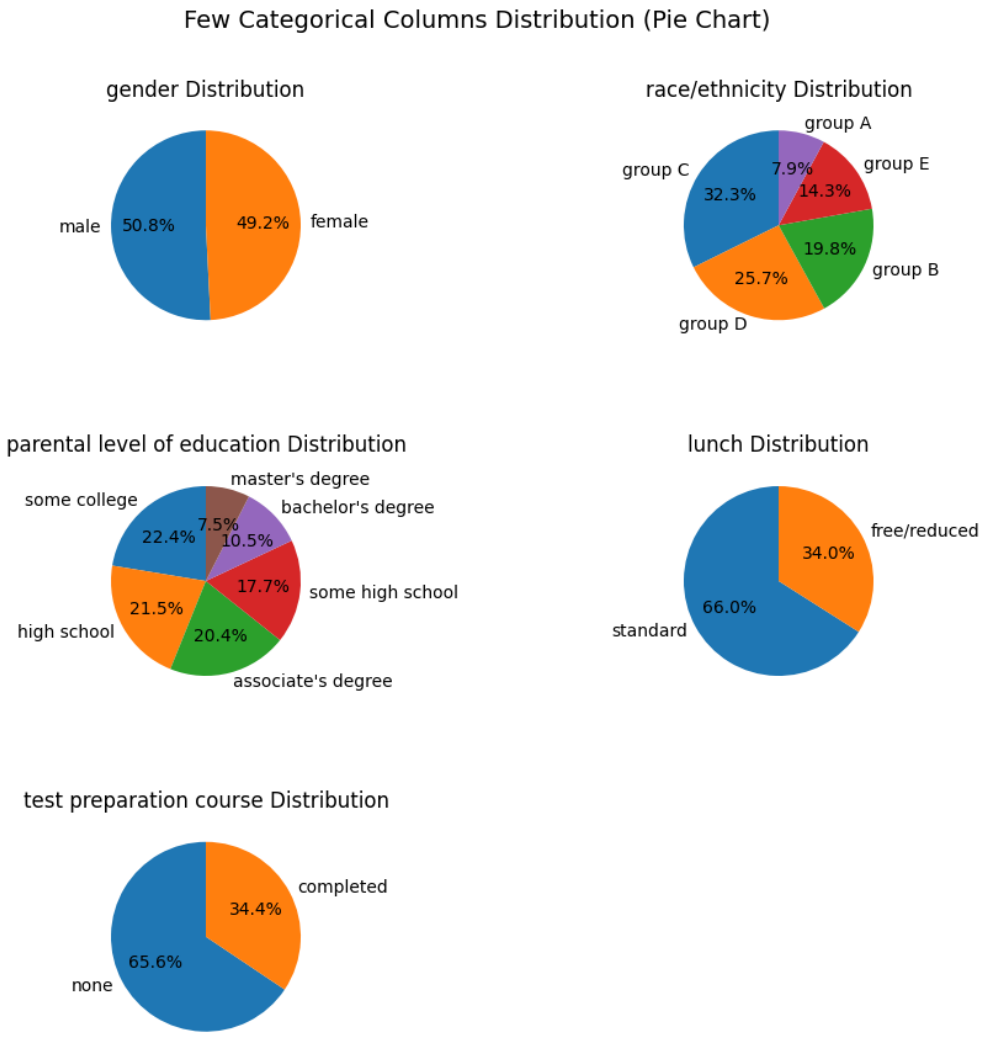

Рисунок 3.10 – Діаграми данних

Створюється матриця графіків з розміром (2, 2) за допомогою plt.subplots(2, 2, figsize=(10, 10)).

Далі за допомогою циклу for та enumerate виконується прохід по кожній числовій змінній із списку num_cols. Для кожної змінної будується гістограма за допомогою sns.histplot(x=col, data=per,ax=axes[r,c]).

Також встановлюється підпис для вісі Y та заголовок графіка за допомогою axes[i//2,i%2].set_ylabel("Density") та axes[i//2,i%2].set_title(f"{col} Density") відповідно.

У разі, якщо кількість числових змінних непарна, останній графік видаляється для збереження зручності виведення. На завершення використовується fig.text() для виведення заголовка та plt.show() для виведення побудованої матриці графіків.

```
num_cols=["math score","reading score","writing score"]
fig, axes = plt.subplots(2, 2, figsize=(10, 10))
for i, col in enumerate(num_cols):
    r,c=i//2,i%2
    sns.histplot(x=col, data=per,ax=axes[r,c])
    axes[i//2,i%2].set_ylabel("Density")
    axes[i//2,i%2].set_title(f"{col} Density")
if len(num_cols) % 2 != 0:
    fig.delaxes(axes[-1, -1])
fig.text(0.5,0.95,"Numerical Columns Distribution",va="center", ha="center", fontsize=14)
plt.show()
```

Рисунок 3.11 – Лістинг коду

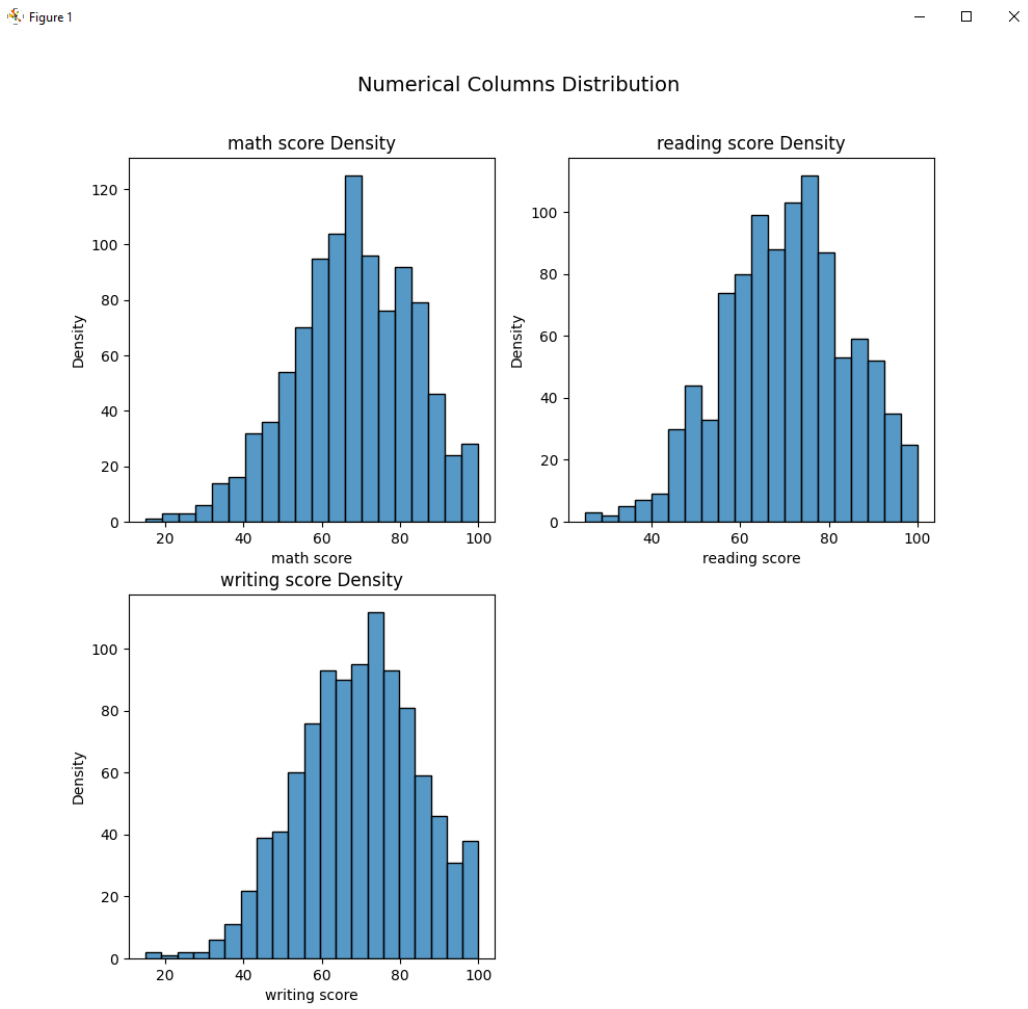

Рисунок 3.12 – Огляд датасету

У цьому коді виводиться повідомлення "Checking the confidence interval of the course scores before creating the new columns for the target variable:". Далі визначаються змінні course_cols, які містять назви предметів. Також визначається насіння для генерації випадкових чисел np.random.seed(10) та розмір вибірки sample_size 1000.

```
print("Checking the confidence interval of the course scores before creating the new columns for the target variable:")
course_cols = ["math score", "reading score", "writing score"]
np.random.seed(10)
sample_size = 1000
```

Рисунок 3.2.12 – Перевірка довірчого інтервалу балів курсу перед створенням нових стовпців для цільової змінної

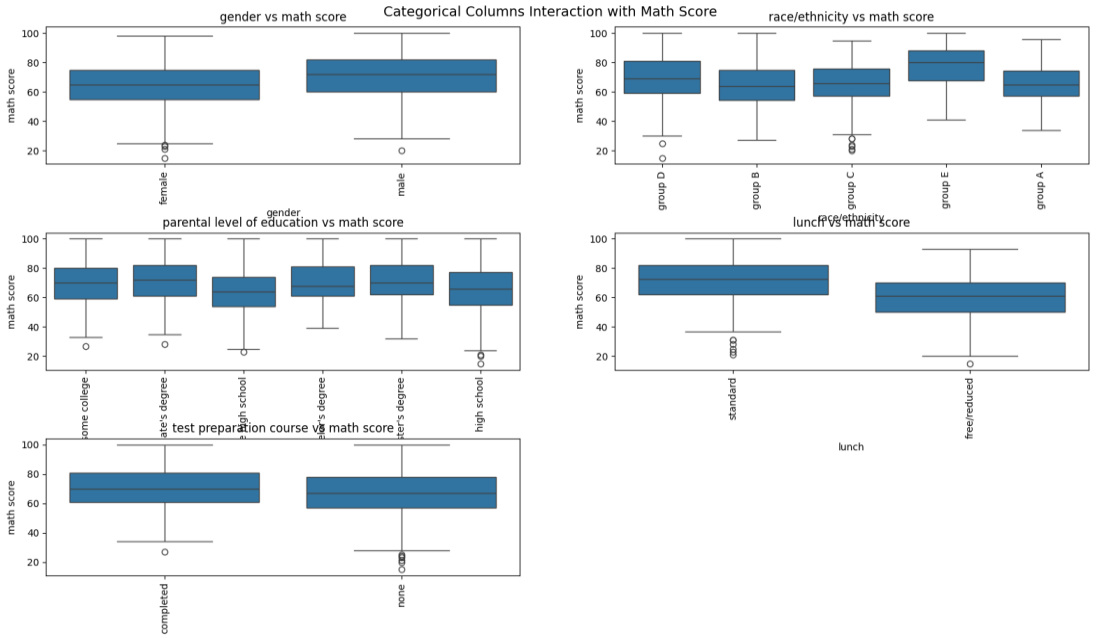

Рисунок 3.14 – Вивід графіків

Далі проведено дослідження взаємодії між числовими та категоріальними змінними, використовуючи графіки розкиду (scatter plots) для числових змінних "reading score" та "writing score" в порівнянні з "math score". Графіки визначають можливі залежності та кореляції між цими оцінками.

Додатково, для підготовки даних до подальшого використання у моделях машинного навчання, застосовано Label Encoding до категоріальних змінних. Використовуючи бібліотеку Scikit-learn, кожна категоріальна змінна кодується у числовий формат, що полегшує їх використання у моделях, які вимагають числові дані.

У виведенні вказано, які стовпці були вибрані для графіків розкиду та які саме числові змінні застосувались для цього аналізу. Також виведено початкові рядки датасету після використання Label Encoding для категоріальних змінних. Окремо наведено інформацію про проведене Label Encoding для кожної категоріальної змінної, зокрема, класи, до яких вони були призначені.

```
num_cols = ["reading score", "writing score"]
fig, axes = plt.subplots(1, 2, figsize=(10, 5))
for i, col in enumerate(num_cols):
    sns.scatterplot(x=col, y="math score", data=per, ax=axes[i])
    axes[i].set_xlabel(f"{col}")
    axes[i].set_ylabel("math score")
    axes[i].set_xticklabels(axes[i].get_xticklabels(), rotation="vertical")
    axes[i].set_title(f"{col} vs math score")
plt.subplots_adjust(hspace=0.5)
fig.suptitle("Numerical Columns Interaction with Math Score", va="center", ha="center", fontsize=14)
plt.show()

label_encoders={}
for col in cat_cols:
    le=LabelEncoder()
    per[col]=le.fit_transform(per[col])
    label_encoders[col]=le
print("Label Encoding the category columns:")
display(per.head())
print("\nEncoding details:")
for column, encoder in label_encoders.items():
    print(f"{column} Label Encoder Classes: {encoder.classes_}")
```

Рисунок 3.15 – Лістинг коду

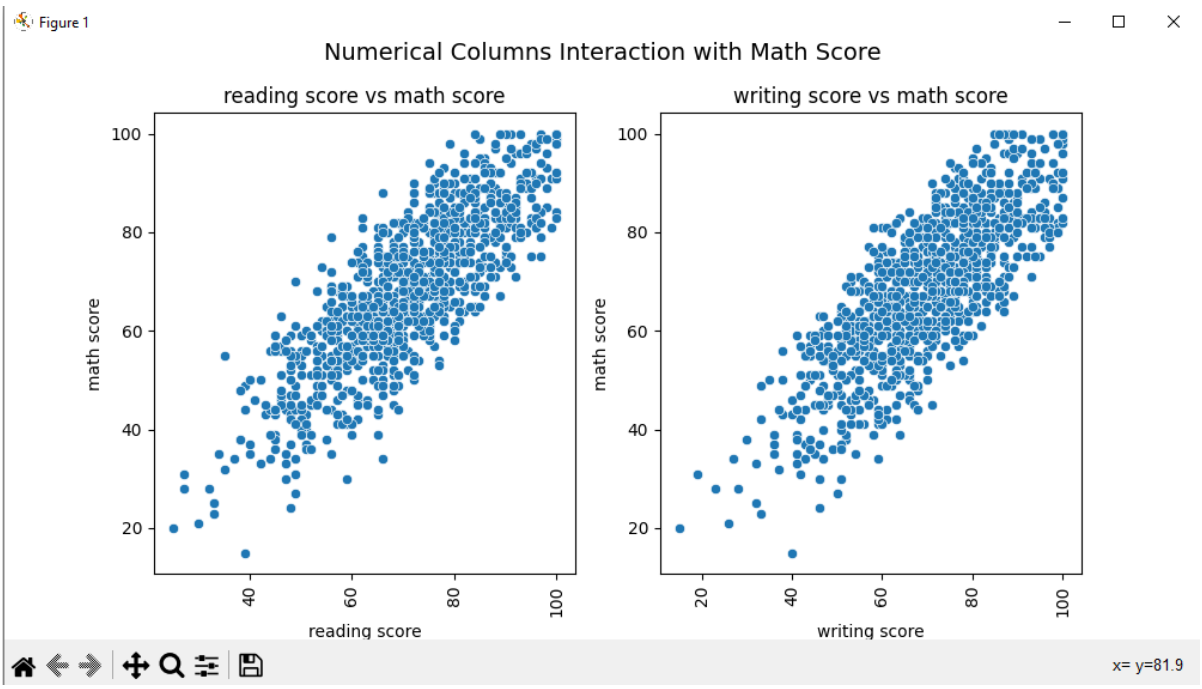

Рисунок 3.16 – scatter plots для числових змінних "reading score" та "writing score" в порівнянні з "math score"

Далі виконується аналіз кореляції між атрибутами датасету за допомогою теплової карти. Дані графічно відображаються за допомогою sns.heatmap, що дозволяє визначити ступінь взаємозв'язку між різними змінними. Після цього дані розділяються на вхідні та вихідні змінні, де "math score" стає вихідною змінною, а інші змінні використовуються для побудови моделі. Далі дані поділяються на навчальний та тестовий набори за допомогою train_test_split для подальшого навчання та валідації моделі. Завершується фрагмент виведенням інформації про розміри навчальних та тестових наборів вхідних та вихідних змінних.

```
plt.figure(figsize=(15,10))
plt.title('Correlation between the attributes:')
sns.heatmap(per.corr(),annot=True)
plt.show()
```

Рисунок 3.17 – Лістинг коду

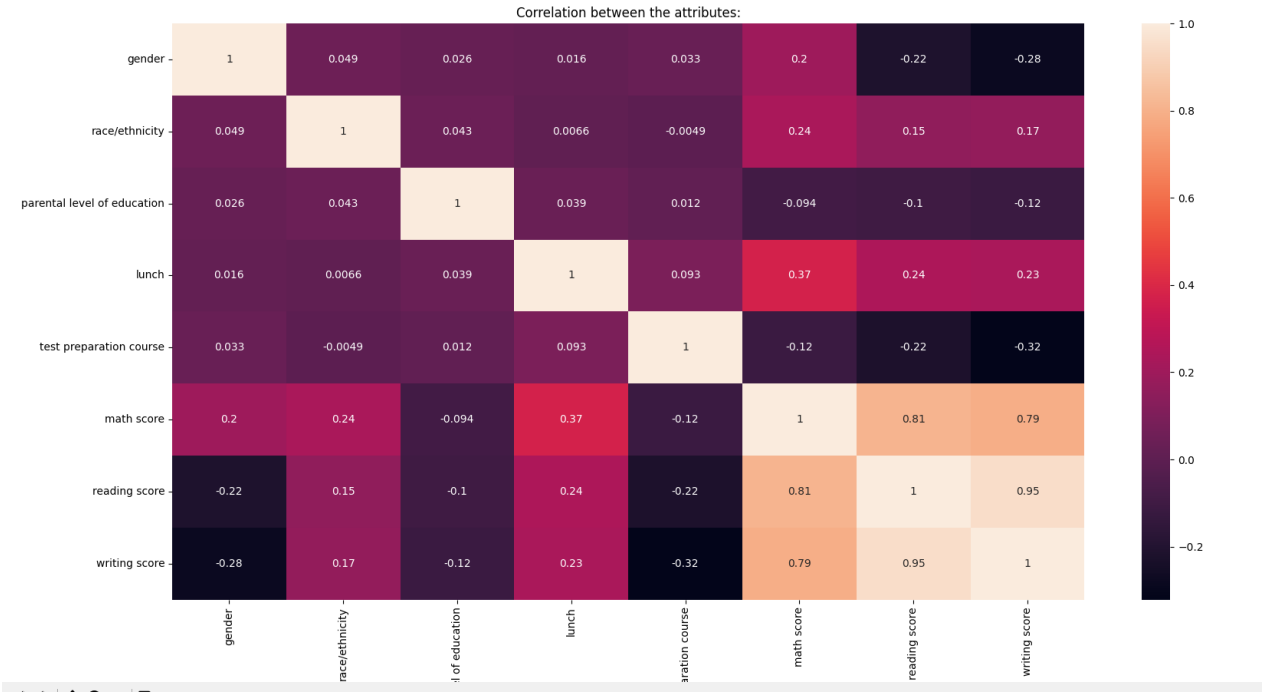

Рисунок 3.18 – Матриця кореляції атрибутів

```
Encoding details:
gender Label Encoder Classes: ['female' 'male']
race/ethnicity Label Encoder Classes: ['group A' 'group B' 'group C' 'group D' 'group E']
parental level of education Label Encoder Classes: ["associate's degree" "bachelor's degree" 'high school' "master's degree"
 'some college' 'some high school']
lunch Label Encoder Classes: ['free/reduced' 'standard']
test preparation course Label Encoder Classes: ['completed' 'none']
Input variables:
     gender  race/ethnicity  ...  reading score  writing score
0         0               3  ...             70             78
1         1               3  ...             93             87
2         0               3  ...             76             77
3         1               1  ...             70             63
4         0               3  ...             85             86
..      ...             ...  ...            ...            ...
995       1               2  ...             77             71
996       1               2  ...             66             66
997       0               0  ...             86             86
998       1               4  ...             72             62
999       1               3  ...             47             45

[1000 rows x 7 columns]
```

Рисунок 3.19 – Деталі енкодінгу

```
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.25,random_state=42)
print("Shapes after train-test split:")
print(f"Training input: {x_train.shape}")
print(f"Testing input: {x_test.shape}")
print(f"Testing output: {y_train.shape}")
print(f"Testing output: {y_test.shape}")
```

Рисунок 3.20 - Лістинг коду

```
Shapes after train-test split:
Training input: (750, 7)
Testing input: (250, 7)
Testing output: (750,)
Testing output: (250,)
```

Рисунок 3.21 – Огляд датасету

Далі виконується порівняльний аналіз різних моделей машинного навчання для прогнозування "math score". Спочатку створюється список моделей, таких як Linear Regression, Lasso, Ridge, KNeighborsRegressor, SVR, DecisionTreeRegressor, RandomForestRegressor та ExtraTreesRegressor. Потім, використовуючи цикл for, кожна модель навчається на навчальному наборі, здійснює прогноз на тестовому

наборі та обчислює коефіцієнт детермінації R2 Score. Для кожної моделі виводиться інформація про її використання та досягнутий R2 Score. На завершення циклу визначається модель, яка має найвищий R2 Score. У виведенні вказується модель з найкращою точністю в прогнозуванні "math score".

```
LinearRegression()
R2 Score: 0.8740713514754446


Model used:
Lasso()
R2 Score: 0.8063651078389094


Model used:
Ridge()
R2 Score: 0.8739411770651546


Model used:
KNeighborsRegressor()
R2 Score: 0.6415794710588982


Model used:
SVR()
R2 Score: 0.5951172875032489


Model used:
DecisionTreeRegressor()
R2 Score: 0.7223838281387234
```

Рисунок 3.22 – Порівняння моделей

```
Model used:
RandomForestRegressor()
R2 Score: 0.8517657684504504


Model used:
ExtraTreesRegressor()
R2 Score: 0.8509944916656274
```

Рисунок 3.23 – Порівняння моделей

```
Best R2 Score Recorded: 0.8740713514754446.
Best Model Performance: LinearRegression().
```

Рисунок 3.24 – Вивід найкращої моделі

Далі вводяться деталі для прогнозування "math score" на основі певних характеристик. Спочатку вводяться значення для різних характеристик, таких як стать, етнічна група, рівень освіти батьків, обід, курс підготовки до тесту, рівні оцінок з читання та письма. Значення заносяться в словник cols_to_encode. Далі використовуються попередньо збережені екземпляри LabelEncoder для кодування категоріальних змінних. Закодовані значення виводяться, а також створюється новий рядок даних для подальшого прогнозування. Модель, яка показала найкращу точність, застосовується для прогнозу "math score" для нових даних, і отримані результати виводяться разом зі створеним рядком даних з передбаченою оцінкою.

```
print("Enter the details for the prediction:")
encoded_values = {}
gender = "male"
race = "group C"
parental_level_of_education = "some college"
lunch = "standard"
test_preparation_course = "none"
reading_score = 77
writing_score = 71
print([gender,race,parental_level_of_education,lunch,test_preparation_course,reading_score,writing_score])
cols_to_encode = {
    "gender": gender,
    "race/ethnicity": race,
    "parental level of education": parental_level_of_education,
    "lunch": lunch,
    "test preparation course": test_preparation_course}

for col, val in cols_to_encode.items():
    le = label_encoders[col]
    encoded_values[col] = le.transform([val])

print("Encoded values:")
display(encoded_values)
print()
new_data = pd.DataFrame(encoded_values)
new_data['reading score'] = float(reading_score)
new_data['writing score'] = float(writing_score)
print("New row for predictions:")
display(new_data)
print()
```

Рисунок 3.25 – Лістинг коду

```
Enter the details for the prediction:
['male', 'group C', 'some college', 'standard', 'none', 77, 71]
Encoded values:
{'gender': array([1]), 'race/ethnicity': array([2]), 'parental level of education': array([4]), 'lunch': array([1]), 'test preparation course': array([1])}

New row for predictions:
   gender  race/ethnicity  ...  reading score  writing score
0       1               2  ...           77.0           71.0

[1 rows x 7 columns]

Predicted Math Score: 80.14

Row with predicted math score:
   gender  race/ethnicity  ...  writing score  math score
0       1               2  ...           71.0       80.14

[1 rows x 8 columns]
```

Рисунок 3.26 – Передбачення за допомогою найкращої моделі

Висновки до розділу 3

В рамках розробки системи аналізу успішності студентів було застосовано комплексний підхід до вибору та оцінки моделей машинного навчання. Використання мови програмування Python разом з бібліотеками Scikit-learn

забезпечило необхідні інструментальні засоби для реалізації та порівняння різних алгоритмів, включаючи лінійну регресію, дерева рішень, ансамблеві методи та інші.

Ключовим аспектом роботи системи став процес вибору найбільш ефективної моделі для прогнозування успішності студентів. Для цього було проведено порівняльний аналіз моделей на основі метрики R2 Score, що дозволяє оцінити точність прогнозувань моделі відносно варіативності вихідних даних. Використання цієї метрики забезпечило об'єктивну оцінку продуктивності кожної моделі.

На основі проведеного аналізу була вибрана найкраща модель - LinearRegression, що відзначилася найвищим значенням R2 Score, як найбільш підходяща для прогнозування успішності студентів. Вибір оптимальної моделі забезпечив основу для подальшого розроблення системи, дозволяючи впроваджувати ефективні стратегії оцінювання та вдосконалення навчального процесу на основі даних про успішність студентів.

ВИСНОВКИ

На основі аналізу сучасного стану використання машинного навчання у прогнозуванні академічної успішності студентів, розроблено комплексний підхід, що враховує не лише академічні показники, але й особистісні характеристики, мотиваційні аспекти, стратегії саморегуляції та психосоціальний контекст. Виявлено, що інтеграція різноманітних даних та використання передових алгоритмів машинного навчання забезпечують високу точність прогнозування, що може слугувати основою для розробки індивідуалізованих навчальних планів та своєчасної підтримки студентів. Особлива увага приділяється необхідності постійного оновлення моделей на основі зворотного зв'язку та адаптації до змінних освітніх умов, що підкреслює потребу в гнучких та відкритих системах електронного навчання, здатних ефективно реагувати на потреби студентів та освітнього процесу в цілому.

У розробленій системі для аналізу успішності студентів використано Python та Scikit-learn, обрано найкращу модель - LinearRegression, за її високий R2 Score. Система інтегрує різні алгоритми машинного навчання, здійснюючи порівняльний аналіз моделей для точного прогнозування. Важливим аспектом є вибір ефективної моделі, що забезпечує основу для подальшого вдосконалення навчального процесу, враховуючи дані про успішність.

Розроблена система, яка використовує алгоритми машинного навчання для прогнозування академічної успішності студентів, має великий потенціал для інтеграції в хмарні сервіси. Це дозволить забезпечити ширший доступ до системи для освітніх установ, спростить процес оновлення та масштабування системи, а також поліпшить зберігання та обробку даних. Інтеграція в хмарні сервіси також забезпечить високий рівень безпеки даних та зручність використання системи в реальному часі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Чередніченко О. Моделі формування рекомендацій у інтелектуальних системах електронної комерції / О. Чередніченко, О. Янголенко, О. Іващенко, О. Матвєєв // Системи обробки інформації. 2020. – Вип. 1 (160). – С. 32-39

2. Schedl M, Zamani H, Chen CW, Deldjoo Y, Elahi M. Current challenges and visions in music recommender systems research. Int J Multi Inform Ret. (2018) 7:95–116. doi: 10.1007/s13735-018-0154-2

11. Han J, Kamber M, Pei J. Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, 2006. – 816 с.

3. Zaki M.J., Meira Jr W. Data Mining and Analysis: Fundamental Concepts and Algorithms. Cambridge University Press, 2014. – 534 с.

4. Kelleher JD, Mac Namee B, D'Arcy A. Data Mining: A Tutorial-Based Primer. CRC Press, 2015. – 392 с.

5. Mohammed J. Zaki, Wagner Meira Jr. Data Mining and Analysis: Fundamental Concepts and Algorithms. Cambridge University Press, 2014. – 534 с.

6. John D. Kelleher, Brian Mac Namee, Aoife D'Arcy. Data Mining: A Tutorial-Based Primer. CRC Press, 2015. – 392 с.

7. Rakesh Agrawal, Ramakrishnan Srikant. "Fast algorithms for mining association rules" Proceedings of the 20th International Conference on Very Large Data Bases, 1994, pp. 487-499.

8. Kristina Chodorow. Scaling MongoDB: Sharding, Cluster Setup, and Administration – USA.: 1st Edition, Kindle Edition, 2011. – 79 с.

9. MongoDB документація. [Електронний ресурс] – Режим доступу до ресурсу: https://www.mongodb.com/

10. Chanchal Singh. MongoDB Cookbook: Harness the power of MongoDB to build efficient and scalable data-driven applications. Packt Publishing, 2013. – 324 с.

11. Manuel Kiessling. The Node Beginner Book: A comprehensive Node.js tutorial. Leanpub, 2011. – 68 с.

12. Kevin Scott. REST API Design: Best Practices in API Design with REST – UK.: Packt Publishing, Limited, 2018. – 256 с.

13. Mongoose документація. [Електронний ресурс] – Режим доступу: – https://mongoosejs.com/docs/index.html.

14. Robin Nixon. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. O'Reilly Media, Inc., 2014. – 832 с.

15. Eric A. Meyer, Estelle Weyl. CSS: The Definitive Guide: Visual Presentation for the Web. 4th Edition. O'Reilly Media, Inc., 2018. – 984 с.

16. Сучасний підручник з JavaScript. [Електронний ресурс] – Режим доступу: – https://uk.javascript.info.

17. J. Knell. Shopify Essentials: A Guide to Setting up Your Online Store. Packt Publishing, Limited, 2019. – 346 с.

18. Talha Majid. "Liquid and Shopify: The Ultimate Guide to Building Dynamic E-Commerce Websites" - Independently published, 2019. – 240 с.

19. Ivan Djordjevic. Shopify Theme Customization with Liquid: Principles, Top Techniques, and Projects to Leverage One of the Fastest-Growing ECommerce – UK.: Packt Publishing, Limited, 2021. – 338 с.

20. J. K. Shani, G. Gunawardana, and A. Karatzoglou. Recommender Systems An Introduction. Cambridge University Press, 2011. – 280 с.

21. B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-Based Collaborative Filtering Recommendation Algorithms. Proceedings of the 10th international conference on World Wide Web, 2001. – 285-295 с.

22. X. Ning and G. Karypis. SLIM: Sparse Linear Methods for Top-N Recommender Systems. ACM Transactions on Knowledge Discovery from Data, 2011. – 1-30 с.

23. Y. Koren. Matrix Factorization Techniques for Recommender Systems. Computer, 2010. – 30-37 с.

24. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems, 2004. – 5-53 с.

25. B. Liu. Integrating classification and association rule mining. In Proceedings of KDD, 1998. – 80-86 с.

26. ДСН 12.0.003-74. Небезпечні і шкідливі виробничі фактори.

27. ДСН 12.1.001-83. Ультразвук. Загальні вимоги безпеки.

28. ДСН 12.1.003-83. Шум. Загальні вимоги безпеки.

29. ДСН 12.1.004-91. Пожежна безпека.

30. ДСН 12.1.005-88. Загальні санітарно-гігієнічні вимоги до повітря робочої зони.

31. ДСН 12.1.006-76. Електромагнітні поля різночастот. Загальні вимоги безпеки.

32. Гандзюк М.П. Основи охорони праці: Підручник/ М.П. Гандзюк, Є.П. Желібо, М. О. Халімовський. - Львів: Новий світ-2000, 2003. – 408 с

33. Гогіташвілі Г.Г. Основи охорони праці: Навчальний посібник/ Г.Г. Гогіташвілі, В.М. Лапін. - Львів: Новий світ-2000, 2006. – 232 с

34. Гогіташвілі Г.Г. Управління охороною праці та ризиками за міжнародними стандартами:

35. Грищук М.В. Основи охорони праці: Підручник/ М. В. Грищук. - К.: Кондор, 2005. – 240 с.

36. Желібо Є. П., Баранова Н. І., Коваленко В.В. Охорона праці в органах державної податкової служби. Навч. посібник для ВНЗ. Ірпінь. – 2002.

37. Катренко Л.А., Кіт Ю.В., Пістун І. П. Охорона праці. Курс лекцій. Практикум: Навч. посіб. – Суми: Університетська книга, 2009. – 540 с.

38. Керб Л.П. Основи охорони праці : Навчальний посібник/ Л. П. Керб. - Вид. 2-ге, без змін. - К.: КНЕУ, 2006. – 216 с19. Закон України “Про охорону праці” / Законодавство України про охорону праці. - К. Нова редакція 2002 р.

39. Закон України “Про охорону навколишнього природного середовища” – К.: Україна. – 1991. - 59 с. (з усіма редакціями до 2017 року)

40. НПАОП 0.00-1.28-10 Правила охорони праці під час експлуатації електронно-обчислювальних машин/ Зареєстровано в Міністерстві юстиції України 19 квітня 2010 р. за N 293/17588

41. Правила улаштування электроустановок. ПУЕ.– Харків.: Форт – 2011 – 728 с.

42. Гігієнічна класифікація праці за показниками шкідливості та небезпечності факторів виробничого середовища, важкості та напруженості трудового процесу. Гігієнічні нормативи ГН 3.3.5-8-6.6.1 2002 р. Видання офіційне Київ, 2001 рік – 46 с.

43. ДБН В.2.5-67:2013. Опалення, вентиляція та кондиціювання . -К.: Мінрегіон України, 2013.-147 с

44. ДБН.В.2.5 – 28-2006 . Природне і штучне освітлення. – К.: Мінбуд України, - 2008 – 74 с.

ДОДАТОК А

Лістинг коду

```
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

from IPython.display import display

warnings.filterwarnings('ignore')
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
import scipy.stats as stats
from sklearn.linear_model import LinearRegression, Lasso, Ridge
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor, ExtraTreesRegressor
from sklearn.metrics import r2_score, mean_absolute_error,
mean_squared_error
print("All dependencies are imported.")

per=pd.read_csv('exams.csv')
print("Student performance dataset:")
display(per)

print("Dataset information:")
```

```
display(per.info())

print("Null counts:")
display(per.isnull().sum())

cat_cols = ["gender", "race/ethnicity", "parental level of education", "lunch", "test preparation course"]
fig, axes = plt.subplots(3, 2, figsize=(20, 20))
for i, col in enumerate(cat_cols):
    r = i // 2
    c = i % 2
    sns.countplot(x=col, data=per, ax=axes[r, c])
    for container in axes[r, c].containers:
        axes[r, c].bar_label(container, label_type="edge")
    axes[r, c].set_ylabel(f"{col}")
    axes[r, c].set_xlabel("Frequency")
    axes[r, c].set_xticklabels(axes[r,c].get_xticklabels(),rotation="vertical")
    axes[r, c].set_title(f"{cat_cols[i]} Distribution")
if len(cat_cols) % 2 != 0:
    fig.delaxes(axes[-1, -1])
plt.subplots_adjust(hspace=0.5)
fig.text(0.5,0.9,"Few Categorical Columns Distribution (Bar Graph)", va="center", ha="center", fontsize=14)
plt.show()

cat_cols = ["gender", "race/ethnicity", "parental level of education", "lunch", "test preparation course"]
fig, axes = plt.subplots(3, 2, figsize=(10, 10))
for i, col in enumerate(cat_cols):
```

```
        r = i // 2
        c = i % 2
        counts = per[col].value_counts()
        axes[r, c].pie(counts, labels=counts.index, autopct='%1.1f%%', startangle=90)
        axes[r, c].set_title(f"{col} Distribution")
    if len(cat_cols) % 2 != 0:
        fig.delaxes(axes[-1, -1])
    plt.subplots_adjust(hspace=0.5, wspace=0.5)
    fig.text(0.5,0.95,"Few Categorical Columns Distribution (Pie
Chart)",va="center", ha="center", fontsize=14)
    plt.show()

    num_cols=["math score","reading score","writing score"]
    fig, axes = plt.subplots(2, 2, figsize=(10, 10))
    for i, col in enumerate(num_cols):
        r,c=i//2,i%2
        sns.histplot(x=col, data=per,ax=axes[r,c])
        axes[i//2,i%2].set_ylabel("Density")
        axes[i//2,i%2].set_title(f"{col} Density")
    if len(num_cols) % 2 != 0:
        fig.delaxes(axes[-1, -1])
    fig.text(0.5,0.95,"Numerical Columns Distribution",va="center", ha="center",
fontsize=14)
    plt.show()

    print("Checking the confidence interval of the course scores before creating the
new columns for the target variable:")
    course_cols = ["math score", "reading score", "writing score"]
    np.random.seed(10)
```

```
    sample_size = 1000

    for col in course_cols:
        print(f"\n{col}:")
        sample = np.random.choice(a=per[col], size=sample_size)  # Corrected typo in np.random.choice
        sample_mean = sample.mean()
        z_critical = stats.norm.ppf(q=0.95)
        print("Z critical value:")
        print(z_critical)
        pop_stdev = per[col].std()
        print("Population standard deviation:")
        print(pop_stdev)
        margin_of_error = z_critical * (pop_stdev / np.sqrt(sample_size))  # Corrected typo in np.sqrt
        confidence_interval = (sample_mean - margin_of_error, sample_mean + margin_of_error)
        print("Confidence Interval:")
        print(confidence_interval)
        print(f"True Mean of {col}:")
        print(per[col].mean())

    cat_cols = ["gender", "race/ethnicity", "parental level of education", "lunch", "test preparation course"]
    fig, axes = plt.subplots(3, 2, figsize=(20, 20))
    for i, col in enumerate(cat_cols):
        r = i // 2
        c = i % 2
        sns.boxplot(x=col,y="math score", data=per, ax=axes[r, c])
```

```
            for container in axes[r, c].containers:
                axes[r, c].set_xlabel(f"{col}")
                axes[r, c].set_ylabel("math score")
                axes[r, c].set_xticklabels(axes[r,c].get_xticklabels(),rotation="vertical")
                axes[r, c].set_title(f"{cat_cols[i]} vs math score")
        if len(cat_cols) % 2 != 0:
            fig.delaxes(axes[-1, -1])
        plt.subplots_adjust(hspace=0.5)
        fig.text(0.5,0.9,"Categorical Columns Interaction with Math Score",
va="center", ha="center", fontsize=14)
        plt.show()

        num_cols = ["reading score", "writing score"]
        fig, axes = plt.subplots(1, 2, figsize=(10, 5))
        for i, col in enumerate(num_cols):
            sns.scatterplot(x=col, y="math score", data=per, ax=axes[i])
            axes[i].set_xlabel(f"{col}")
            axes[i].set_ylabel("math score")
            axes[i].set_xticklabels(axes[i].get_xticklabels(), rotation="vertical")
            axes[i].set_title(f"{col} vs math score")
        plt.subplots_adjust(hspace=0.5)
        fig.suptitle("Numerical Columns Interaction with Math Score", va="center",
ha="center", fontsize=14)
        plt.show()

        label_encoders={}
        for col in cat_cols:
            le=LabelEncoder()
            per[col]=le.fit_transform(per[col])
```

```
        label_encoders[col]=le
    print("Label Encoding the category columns:")
    display(per.head())
    print("\nEncoding details:")
    for column, encoder in label_encoders.items():
        print(f"{column} Label Encoder Classes: {encoder.classes_}")

    plt.figure(figsize=(15,10))
    plt.title('Correlation between the attributes:')
    sns.heatmap(per.corr(),annot=True)
    plt.show()

    y=per["math score"]
    x=per.drop("math score", axis=1)
    print("Input variables:")
    display(x)
    print()
    print("Output:")
    display(y)

    x_train, x_test, y_train,
y_test=train_test_split(x,y,test_size=0.25,random_state=42)
    print("Shapes after train-test split:")
    print(f"Training input: {x_train.shape}")
    print(f"Testing input: {x_test.shape}")
    print(f"Testing output: {y_train.shape}")
    print(f"Testing output: {y_test.shape}")
```

```
models=[LinearRegression(), Lasso(), Ridge(), KNeighborsRegressor(), SVR(),
DecisionTreeRegressor(),
        RandomForestRegressor(), ExtraTreesRegressor()]
r2_scores=[]
for model in models:
    print("Model used:")
    display(model)
    model.fit(x_train, y_train)
    y_pred=model.predict(x_test)
    r2=r2_score(y_test, y_pred)
    print(f"R2 Score: {r2}")
    r2_scores.append(r2)
    print("\n")
max_r2=max(r2_scores)
print(f"Best R2 Score Recorded: {max_r2}.")
max_idx=r2_scores.index(max_r2)
best_model=models[max_idx]
print(f"Best Model Performance: {best_model}.")

print("Enter the details for the prediction:")
encoded_values = {}
gender = "male"
race = "group C"
parental_level_of_education = "some college"
lunch = "standard"
test_preparation_course = "none"
reading_score = 77
writing_score = 71
```

```
print([gender,race,parental_level_of_education,lunch,test_preparation_course,reading_score,writing_score])
cols_to_encode = {
   "gender": gender,
   "race/ethnicity": race,
   "parental level of education": parental_level_of_education,
   "lunch": lunch,
   "test preparation course": test_preparation_course}

for col, val in cols_to_encode.items():
   le = label_encoders[col]
   encoded_values[col] = le.transform([val])

print("Encoded values:")
display(encoded_values)
print()
new_data = pd.DataFrame(encoded_values)
new_data['reading score'] = float(reading_score)
new_data['writing score'] = float(writing_score)
print("New row for predictions:")
display(new_data)
print()
pred_math_score = best_model.predict(new_data)
pred_math_score = round(pred_math_score[0],2)
print(f"Predicted Math Score: {pred_math_score}")
new_data["math score"]= pred_math_score
print("\nRow with predicted math score:")
display(new_data)
```