МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ СЕМЕНА КУЗНЕЦЯ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ

Рівень вищої освіти	Перший (бакалаврський)
Спеціальність	Інженерія програмного забезпечення
Освітня програма	Інженерія програмного забезпечення
Група	6.04.121.010.18.1

ДИПЛОМНИЙ ПРОЄКТ

на тему: «Розроблення модуля “Аналіз успішності студентів” на основі методів Data mining»

Виконав: студент Дмитро ЖОВТОБРЮХ

Керівник: к.т.н., проф. Олександр ЩЕРБАКОВ

Рецензент: к.т.н., доцент кафедри інформатики
та комп’ютерної техніки
Наталя БРИНЗА

Харків – 2022 рік

РЕФЕРАТ

Пояснювальна записка: 90 с., 49 рис., 39 джерел, 4 табл., 1 додаток.

Об'єкт дослідження: модель для аналізу успішності даних студента.

Мета роботи: розроблення модуля "Аналіз успішності студентів" на основі методів Data mining.

Методи дослідження. Для досягнення поставленої мети були використано наукові досягнення в галузях інтелектуального аналізу даних, машинного навчання і нейронних мереж.

Значення результатів дослідження полягає у аналізі та виявленні закономірностей у даних про поточну академічну успішність студента, а також в реалізації моделей даних для передбачення успішності студента у наступному семестрі.

Практичною цінністю результатів дослідження є створення модулів програмного продукту, які дозволяють робити висновки щодо навчального процесу на основі даних про поточну успішність.

Область застосування. Розроблений модуль та моделі можуть застосовуватися для вирішення завдань прогнозування успішності студента, зокрема, для виявлення чинників, що впливають на навчальний процес та академічну стратегію студента з метою покращення якості освіти, яка надається за певною освітньо-професійною програмою.

Значення роботи та висновки. Отримані моделі можуть бути застосовані при аналізі навчального процесу та дозволяють передбачати успішність студента і визначати чи має він схильність до високих чи низьких академічних досягнень.

Прогнози щодо розвитку досліджень. Розроблені моделі показали середню точність, але на такому рівні, який зазначався в інших дослідженнях за описаною проблематикою. Оскільки потенційним покращенням може бути збільшення вимірюваних характеристик навчального процесу, пропонується додати в навчальний процес відстеження відвідуваності занять, середньосеместрової оцінки та демографічних показників.

Список ключових слів: EDUCATIONAL DATA MINING, ОСВІТНЯ АНАЛІТИКА, ПЕРЕДБАЧЕННЯ УСПІШНОСТІ СТУДЕНТІВ, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, PYTHON.

ABSTRACT

Explanatory note: 90 p., 49 img., 39 references, 5 tab., 1 annex.

Object of study: a students' academic performance prediction model.

The goal of the project: the development of the software module for students' academic performance analysis.

Methods of research: state-of-the-art data science, machine learning methods and neural networks.

The value of the results is to analyze and identify patterns in the data on the current academic performance of the student, as well as in the implementation of data models to predict student performance in the next semester.

The practical value of the study is the development of software modules that allows you to obtain knowledge from the learning process based on data on current performance.

Scope. The developed module is applicable to solve the problem of students' performance prediction, in particular, to identify factors that affect the educational process and academic strategy of the student in order to improve the quality of education provided by a particular educational and professional program.

The value of the study and conclusions. The developed models can be used for the analysis of the educational process and allow to predict the student's success and determine whether he has a tendency to high or low academic achievement.

Forecasts for research development. The developed models showed average accuracy, however still at the sufficient level noted in other studies on the described issues. As recording of new features of the learning process might be a potential improvement, it is proposed to add to the learning process tracking of attendance, mid-term score and demographic indicators.

Key words list: EDUCATIONAL DATA MINING, LEARNING ANALYTICS, STUDENT PERFORMANCE PREDICTION, DATA SCIENCE, PYTHON.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ANN – Artificial Neural Network
CRISP–DM – Cross-industry standard process for data mining
DM – Data Mining
DS – Data Science
DT – Decision Tree
EDM – Educational Data Mining
kNN – K-Nearest Neighbours
LA – Learning Analytics
LR – Logistic Regression
ML – Machine Learning
NB – Naive Bayes
RF – Random Forest
SVM – Support Vector Machine
ІС – Інформаційна система

ВСТУП

Успішність студента – комплексна оцінка не тільки академічних досягнень студента, але й ефективності налагодження навчального процесу в закладі вищої освіти (ЗВО), тому є важливим показником, що визначає якість освіти, яка надається за певною освітньо-професійною програмою (ОПП). Аналіз та передбачення результатів навчання студента все більше цікавлять дослідників та науковців, адже якість освіти безпосередньо визначає спроможність кураторів навчального процесу відповідати потребам здобувачів освіти. Внаслідок цього, ЗВО, які прагнуть підтримувати якість освіти на високому рівні та долучатися до розвитку суспільства, повинні безперервно моніторити відповідність навчальних планів вимогам студентів та індустрії.

Основним завданням аналізу та передбачення успішності студента є віднайдення найбільш впливових чинників освітнього процесу на успішність студента. Великою перевагою даної ідеї є наявність інформаційних систем для збору інформації про навчальний процес у більшості освітніх інституцій та ЗВО, які є основним джерелом даних для подальшого аналізу. Суть аналізу полягає у віднайденні прихованих закономірностей в даних (Data Mining), які свідчать про вплив освітніх чинників на успішність студента та/або його кар'єрних успіхів у майбутньому. Знайдені закономірності є предметом для обговорень щодо актуалізації ОПП та покращення навчального процесу.

Метою даної бакалаврської роботи є розробка програмного модуля для аналізу успішності студентів з використанням засобів Data Mining.

Для досягнення поставленої в роботі мети було поставлено і розв'язано наступні завдання:

1. Аналіз стану досліджень за темою для визначення переваг та недоліків застосування існуючих методик аналізу успішності студентів.

2. Збір датасету (набору даних) для подальшого аналізу та використання попередньо визначених методик.

3. Застосування сучасних програмно-апаратних засобів та створення програмного модуля, що імплементує обрані методики.

Об'єкт дослідження – програмний модуль для аналізу успішності студентів.

Ідея дослідження полягає в розробці модуля аналізу успішності студентів на мові програмування Python, який імплементує різні методи передбачення та пошуку паттернів на основі зібраного датасету.

Методи дослідження. Для досягнення поставленої мети були використано наукові досягнення в галузях інтелектуального аналізу даних, машинного навчання і нейронних мереж та такі методи, як кластеризація, классифікація, регрессія, дерева рішень та ін.

Очікувані результати:

1. Визначення підходу до аналізу успішності студента на основі поточних академічних показників і балів ЗНО.

2. Обрання ефективних моделей для вирішення завдання аналізу успішності студента, оцінка їх точності і підведення висновків щодо подальших покращень або впроваджень моделей.

Наукова новизна отриманих результатів полягає в дослідженні закономірностей у параметрах поточної академічної діяльності студента, таких як оцінки за предметами та результати зовнішнього незалежного оцінювання, та їх можливий вплив на успішність у наступному семестрі.

Практичне значення отриманих результатів полягає в створенні програмного модуля, що імплементує методи аналізу успішності студентів.

Очікується, що результат даного дослідження дозволить ЗВО робити висновки з поточної академічної активності студентів, матиме позитивний вплив на організацію навчального процесу та посприяє підвищенню якості освіти.

Робота складається зі вступу, трьох розділів і висновків. Містить 90 сторінок друкованого тексту, в тому числі 65 сторінки тексту основної частини з 49 рисунками, списку з 39 використаних джерел з найменуваннями на сторінках, додатку на 25 сторінках.

1. ОГЛЯД СТАНУ ДОСЛІДЖЕНЬ ЗА ТЕМОЮ «АНАЛІЗ УСПІШНОСТІ СТУДЕНТІВ»

1.1. Проблема успішності студента та освітні технології

В сучасному світі освіта є стратегічно важливим чинником для довгострокового розвитку суспільства та підвищення рівня життя громадян, і саме тому актуальна та якісна освіта є характерною рисою усіх розвинутих країн. З розвитком технологій в багатьох закладах вищої освіти (ЗВО) в Україні та по всьому світу стали створюватися інформаційні системи (ІС). Також загальновідомою тенденцією є залучення інформаційних технологій шляхом створення і використання інтерактивних завдань, які мають зробити навчальний процес більш зрозумілим, цікавим, а тому і продуктивним. Крім того, в освіті все активніше використовуються інструменти автоматизації, які роблять адміністративні процеси, такі як створення навчальних планів, формування розкладів занять та виставлення оцінок більш прозорими і чесними. Незважаючи на те, що процеси інформатизації та використання ІС в освіті мають наочний позитивний вплив на процеси в ЗВО, деякі джерела вважають, що освітні ІС не повинні бути обмежені лише адміністративними функціями [1] і мають шукати прикладне застосування зібраним даним [2].

Новою характерною рисою останніх років є залучення інноваційних технологій аналізу даних до освітніх ІС. Світові технологічні гіганти наголошують, що галузі штучного інтелекту (AI), машинного навчання (ML) та обробки великих даних (BigData) створять революцію на ринку освітніх послуг (EdTech) [3]. Використання інноваційних технологій аналізу даних стає потужним інструментом як для покращення якості освіти, так і для створення більш доступних навчальних стратегій, які будуть підлаштовані під вимоги та особливості кожного студента.

Разом із поточними світовими тенденціями, інформатизація навчального процесу надає велику кількість цифрових даних з успішності студентів, що навчаються(лися) в певному університеті за певний проміжок часу [4]. Оскільки результат оцінювання успішності студента грунтується на тому, як добре студент досягає своєї навчальної мети, як він мотивований та наскільки ефективним був метод викладання дисципліни, ці дані можна інтерпретувати як ступінь корректності налагодження навчального процесу [5]. Аналіз та використання цих даних стейкголдерами навчального процесу може бути потенційно важливим для оцінювання якості вищої освіти, яку надає ЗВО, та

прийняття рішень щодо актуалізації змісту ОПП [6].

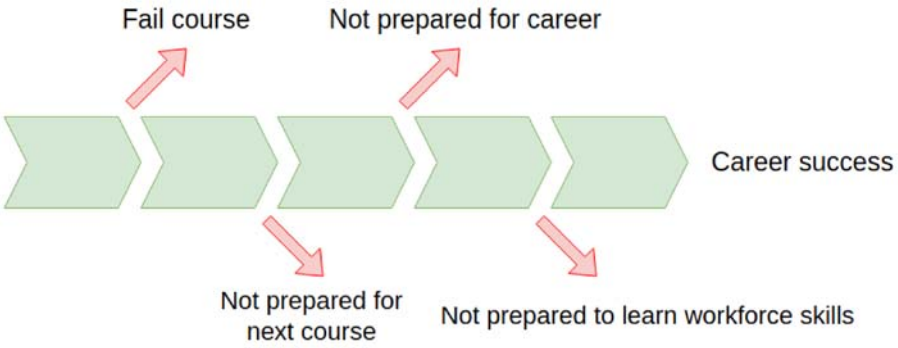


Рис. 1.1. Академічна стратегія студента

Як проіллюстровано на рис. 1.1 [6], академічна стратегія студента має негативні варіанти розвитку, які не дозволяють студентові досягнути поставлених кар'єрних цілей. В цьому разі заклади, які прагнуть розробити найкращі освітні стратегії для студентів, намагаються мінімізувати виникнення таких негативних сценаріїв шляхом розроблення програмних рішень для аналізу академічної поведінки студента, а точніше для аналізу і передбачення його успішності. Саме цими питаннями займаються галузі Learning Analytics та Educational Data Mining [7].

Саме поняття Data Mining (DM) розглядається як сукупність засобів, методів та інструментів для віднайдення попередньо невідомих закономірностей в наборах даних та встановлення залежностей між даними. DM звернув на себе увагу науковців, представників бізнесу та урядів держав за зручну і ефективну можливість робити висновки по даним. На відміну від описової статистики, аналіз даних дозволяє використовувати такі методи, як кластеризація, нейронні мережі та дерева рішень, і надає можливість підтверджувати оцінку досліджень, проведених різними способами. У DM значні і навіть незначні результати, отримані за допомогою описової статистики, можна обробити іншими способами. Завдяки цьому іноді можна досягти інтерпретації, не отриманої за рахунок застосування іншого методу. Іншими словами, дані отримані з баз даних, «видобуваються», щоб знайти різні відносини та визначення. Інтелектуальний аналіз даних забезпечує дослідника гнучкістю у виборі методів, які будуть використовуватися в дослідженні статистичних даних та інтерпретації результатів.

Educational Data Mining (EDM) – це галузь досліджень сфокусована на розвитку методів для аналізу унікальних та широкомасштабних масивів да-

них, що отримуються з освітніх ІС, та використанні цих методів для кращого розуміння поведінки учнів та їх академічних параметрів [8]. Застосування DM до освітніх даних надає унікальну можливість спеціалістам з різних галузей дослідити роботу студентів в університеті та прояснити, які чинники впливають на їх успішність, за допомогою використання сучасних технологій і методів [9–12].

Learning analytics (LA) – це галузь досліджень, яка займається вимірюванням, збором, аналізом та поданням даних про учнів, їх академічні контексти для цілей розуміння та оптимізації навчання і навчальних систем [13].

Для кращого розуміння предметної галузі та взаємовідношень між галузями досліджень, наводиться діаграма, показана на рис. 1.2. Можна відмітити тісний взаємозв'язок між фундаментальними доменами досліджень (Statistics, Computer Science та Education), які посприяли розвитку галузі EDM.

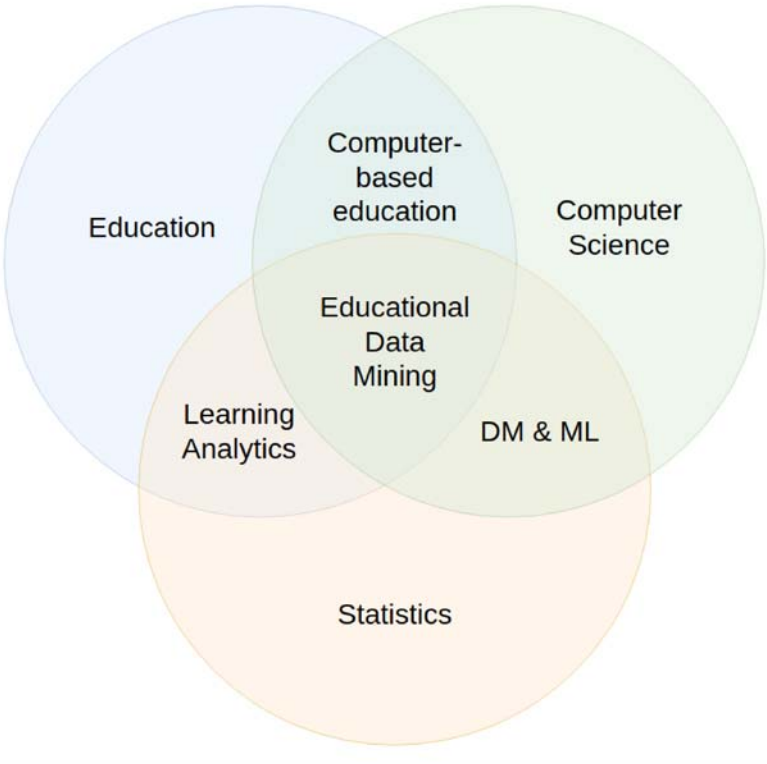


Рис. 1.2. EDM та суміжні галузі

Дане поєднання галузей дозволяє об'єднати кращі методи і практики для побудови моделей аналізу успішності студентів та дослідження їх академічного прогресу. Проблема передбачення успішності студентів є поширеною проблемою для закладів освіти, які роблять акцент на підвищенні якості освіти і ефективності навчального процесу. На рис. 1.3 [10] наведена схема однієї

з методологій, що застосовуються в EDM. Важливим аспектом є те, що дослідження такого типу є довогостроковим, ресурсомістким та ітеративним процесом, на кожному етапі якого є прагнення покращення попередньо отриманих результатів або впровадження змін до навчальної стратегії закладу.

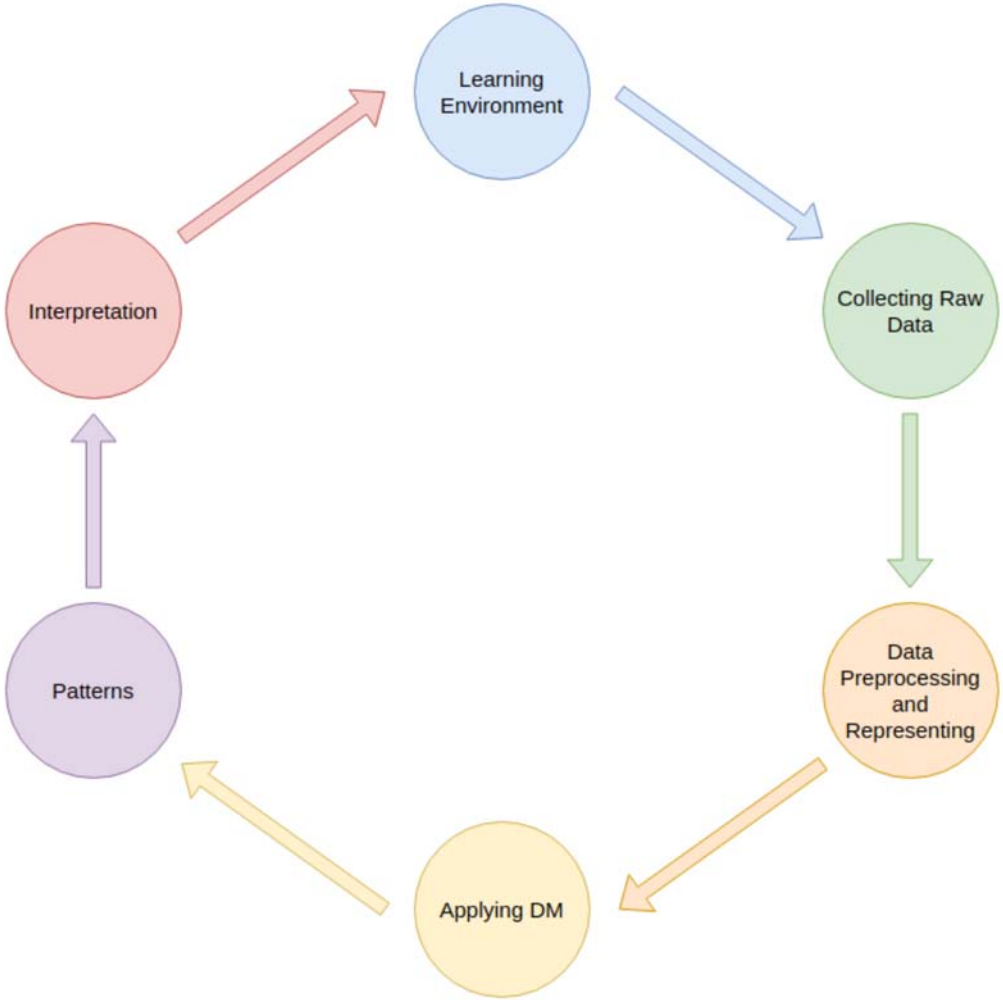


Рис. 1.3. Цикл EDM

Використання вищенаведеного циклу в рамках дослідження має систематизувати і узгодити план дій щодо збору даних, обрання методів, оцінки побудованих моделей і аналізу висновків.

Отже, визначена галузь досліджень є дійсно актуальною і дуже швидко розвивається з застосуванням останніх технологій, а результати досліджень мають практичне значення як для бізнес-сектору, так і для суспільства.

1.2. Огляд стану досліджень

Для огляду стану досліджень було проаналізовано матеріали вітчизняних та іноземних видань, статей з наукометричних баз ScienceDirect, IEEE Xplore та матеріалів конференцій.

Проблема оцінки якості навчання, та використання інтелектуального аналізу даних в цій сфері не є новою, про що свідчить великий проміжок часу між найдавнішими виданнями статей в цій галузі і останніх статей, що вийшли цього року [14–15].

При огляді даної предметної області та статей, особливої уваги потребують статті з поміткою review articles – статті, які публікуються зі спеціальним тегом та які підводять підсумки за найбільш використовуваними алгоритмами, методами та даними, таким чином демонструючи актуальні напрямки та досягнення досліджень даної галузі (рис. 1.4) [16].

No.	Database name	No of papers
1	IEEEXplore	10
2	ScienceDirect	60
3	Springer Link	50
4	EbcoHost	26

Рис. 1.4. Приклад таблиці з кількістю публікацій за темою

Зважаючі на дані (рис. 1.4) можемо зробити висновки, що тема є досить популярною для публікацій, а тому і актуальною.

В. Р. Вергун [17] розглядає проблему класифікації для прогнозування успішності студентів, аналізуючи дані отримані під час анкетування та резюме. Визначена проблема оцінюється як перспективна можливість віднайдення факторів, які можуть покращити зміст майбутніх навчальних програм. Застосовуються методи Random Forest (RF), Logistic Regression (LR), Multilayer Perceptron (MLP) та Support Vector Machines (SVM) .

H. AlQaheri та M. Panda [15] розглядають створення фреймворку для моніторингу академічних процесів. Викладено аналіз застосування Visual Miners для набору даних xAPI з великою кількістю вхідних зміних, які включають також дані про батьків студента та інших академічних метрик виміряних під час занять(наприклад, кількість виступів студента на занятті).

T. Thaher та ін. [10] розглядають передбачення успішності студента базуючись на даних про оцінки за дисципліни, присутність на заняттях, кіль-

кості перездач екзаменів. Автори підтверджують важливість використання класифікації та кластеризації в EDM та описують використання основних алгоритмів, таких як Decision Tree (DT), k-Nearest Neighbours (kNN), Naive Bayes (NB), Linear Discriminant Analysis, SVM та Artificial Neural Network (ANN).

M. Ashraf, M. Zaman та M. Ahmed [12] розглядають створення системи передбачення успішності студентів з використанням ансамблю методів класифікації (JN48(C4.5), NB, RF, kNN) та застосуванням процедури фільтрації. Відмічається, що для створення систем передбачення успішності студентів загальною практикою є використання методів регресії, класифікації та кластеризації. Дослідження виконувалося за методологією CRISP-DM.

M. Zafari та ін. [18] досліжують розробку системи передбачення успішності школярів, основаної на методах машинного навчання. В статті підкреслюються перспективи системи як для передбачення успішності, так і для виявлення студентів з недостатнім рівнем академічних досягнень, а також розглядаються потенціали використання методів RF, SVM, MLP, ANN та NB. Дослідження проводиться на даних про академчіну успішність, академічних показниках виміряних під час занять, а також про склад сім'ї і зайнятість. Визначено, що значними показниками для пердбачення є поточна успішність, час навчання, відвідуваність занять та ін.

K. Zengin та ін. [19] провели вибіркове дослідження застосувань DM в освітніх науках, для поглиблення знань про дані, які вони попередньо досліджували. Розглядається використання описової статистики, DT та кластеризації. Використовуються готові продукти Microsoft для проведення досліджень DM.

K. Govindasamy та T. Velmurugan [20] провели порівняльне дослідження з використання методів класифікації та кластеризації. В дослідженні проаналізовано використання методів C4.5, NB, kNN, k-Means та EM algorithm, та визначено, що дерево рішень C4.5 покзало найкращий результат точності 62.5%.

H. Nawang та ін. [16] провели дослідження-огляд існуючих методів передбачення успішності студентів, шляхом аналізу статей з 2016 по 2020 рр. Підкреслена актуальність досліджень в даній галузі та позначено використання методів регресії, класифікації та кластеризації. Визначено, що найчастіше датасети формують з демографічних характеристик, академічних характеристик, характеристик про попереднє навчання, характеристик соціаль-

них мереж та характеристик з електронних систем освіти. Зазначається, що впливовими характеристиками є не тільки середній бал студента, а й оцінки за екзамени, опитувальники, тести тощо. Другими за значимістю є характеристики про попередню освіту студента. Зазначається, що найбільш поширеними алгоритмами для передбачення є RF, DT, NB, SVM, ANN, LR, and kNN, а в більшості випадків RF показує найкращі результати з передбачення студентів в групі ризику або з незадовільними академічними досягненнями.

R. G. Santosa, Y. Lukito та A. R. Chrismanto [21] дослідили проблеми використання класифікації для передбачення успішності студента при прийомі до навчального закладу. В ході дослідження було застосовано k-means алгоритм на даних про попередню освіту студента. В статті зазначено, що через асиметрію даних були незначні відхилення в результаті кластеризації. Зазначено, що розмір набору даних значною мірою на результати роботи моделі не вплинув.

E. Osmanbegovic та M. Suljic [22] дослідили методи C4.5, MLP and NB для передбачення оцінок студентів. Моделі тренувались на даних про поточну успішність студента, наявність стипендії, попередні академічні досягнення тощо. Найкращу точність показав NB, в той час коли MLP виявився найбільш витратним з точки зору часу алгоритмом.

Alboaneen, D. та ін. [23] дослідили процес розробки веб-застосунку для передбачення успішності студента. В статті було проаналізовано роботу методів SVM, RF, KNN, ANN, LR на датасеті з даними про демогррафічні та академічні характеристики студента. Було розглянуто роботу на двох датасетах, в першому містилася тільки середньосеместрова успішність і демографічні показники, в другому – академічні і демографічні показники. Накращі результати були здобуті за допомогою LR на другому датасеті. Оцінено вплив характеристик на передбачення і визначено, що найбільш впливовою характеристикою є середньосеместрова успішність (вплив 0.6-0.7 в залежності від датасету)

F. J. Kaunang та R. Rotikan [9] розглянули використання Decision Trees в системі передбачення успішності студента. Дані було розмічено на три групи, датасет було створено на базі академічних, демографічних записів та результатів опитувань. Визначено, що великий вплив на передбачення мають освіта батьків студента та щотижневий час занять, а тестова точність моделі склала 63.5%.

Zhang Y та ін. [24] провели порівняльний аналіз проблем в галузі EDM

та їх рішень. Визначено, що регресія в основному використовується для передбачення успішності студента за певним курсом, семестром чи періодом навчання. Кластеризація використовується для розподілення студентів на визначену дослідником кількість груп, а класифікація – для отримання якоїсь дискретної величини (наприклад, оцінка, успіх чи невдача тощо). Розглянуто основні методи DM, такі як DT, RF, ANN, SVM та LR. Визначено основні сфери прикладного використання результатів досліджень: рекомендаційні системи та системи попередження про можливі незадовільні результати екзаменів.

В рамках огляду досліджень було визначено основні напрямки і методи досліджень в галузі EDM, обрано найбільш використовувані методи для подальшої оцінки можливості їх використання для власного дослідження.

1.3. Теоретичні передумови та можливі напрями вирішення

В підрозділі 1.2 було оглянуто широкий спектр досліджень в галузі EDM. Виходячи з розглянутих матеріалів існує два основні напрямки: передбачення чисельної характеристики (регресія), а також розподілення даних студентів на попередньо невідомі класи (кластеризація) та на попередньо відомі класи (класифікація).

Завдання, які потрібно буде вирішити за ходом дослідження і програмної реалізації моделей:

- Feature selection – обирання вхідних змінних (ознак), які будуть вхідними даними для моделі;
- Model selection – вибір моделей для тренування;
- Data preparation – оцінка, перевірка та вибір даних;
- Порівняльний аналіз – вибір моделі, що показала себе найкраще.

Таким чином, на даному етапі було розглянуто предметну галузь освітніх технологій та визначено проблематику аналізу успішності студентів. В ході аналізу джерел було отримано інформацію про актуальний стан досліджень, поширені методи розв'язку проблеми та методологію проведення досліджень. Слід зауважити, що дослідження та імплементація результатів за наведеною темою є критично важливими для створення або модернізації освітніх систем постпандемійного та післявоєнного часу з метою покращення якості освіти та навчального процесу.

2. ОБҐРУНТУВАННЯ ВИКОРИСТАННЯ ОСНОВНИХ ТЕОРЕТИЧНИХ ЗАКОНОМІРНОСТЕЙ ТА СПІВВІДНОШЕНЬ

В даному розділі буде розглянуто, які найкращі практики пропонують використовувати для такого дослідження, базуючись на результатах отриманих в розділі 1.

2.1. Методи машинного навчання та DM

Існує існує два основних підходи в аналізі даних, ML та DM: навчання з наглядом (supervised learning) і навчання без нагляду (unsupervised learning). Основна відмінність полягає в тому, що перший потребує розмічений датасет, щоб спрогнозувати результати, а інший – ні [25].

1. Навчання з наглядом — це метод машинного навчання, характерною рисою якого є використання розмічених наборів даних. Розмічений набір даних містить, як вхідні дані для моделі, так і очікуваний результат. При навчанні модель може порівняти передбачені нею дані з очікуваними, робити висновки і «навчатися» з часом, таким чином покращуючи точність роботи з кожною ітерацією.

З огляду на постановку завдання, навчання з наглядом поділяється на два типи: регресія (передбачення чисельної характеристики) та класифікацію (визначення до якого класу належить екземпляр даних).

2. Навчання без нагляду — це метод машинного навчання, характерною рисою якого є використання нерозмічених наборів даних. Навчання без нагляду використовується для вирішення завдань кластеризації, структурного аналізу даних та віднайдення патернів в датасетах.

Ключовим аспектом, на який необхідно звернути увагу при застосуванні зазначених підходів, є дані. Під час навчання з наглядом алгоритм «навчається» на тренувальному датасетів шляхом ітераційного передбачення даних і самокоригування для віднайдення правильної відповіді та покращення власної точності.

Незважаючи на те, що моделі навчання з наглядом, зазвичай, демонструють більшу точність, ніж моделі навчання без нагляду, вони вимагають попереднього втручання аналітика для належного розмічання даних та інтерпретації отриманих результатів. З іншого боку, моделі навчання без нагляду працюють самостійно, з метою виявлення структур в датасеті і знайдення закономірностей. Тим не меньш, вони все ще потребують втручання аналітика

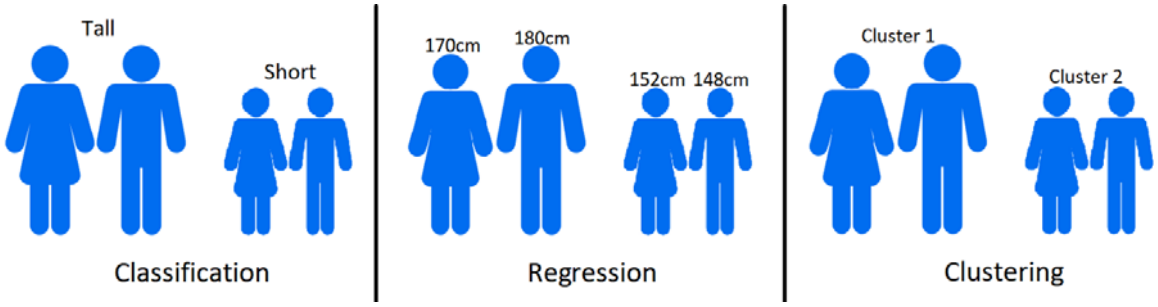


Рис. 2.1. Порівняння регресії, класифікації і кластеризації

для тлумачення вихідних змінних і визначення результатів роботи моделі.

Основні відмінності обох підходів:

- Цілі: навчання з наглядом повинно передбачати результати для нових даних. Попередньо відомо, яких результатів очікувати, оскільки датасет є розміченим і є підтвердження з предметної галузі, що модель повинна передбачати саме такі дані. З іншого боку у алгоритму навчання без нагляду мета полягає в тому, щоб отримати знання з великих обсягів нових даних. Таким чином, метод визначає, які є закономірності або цікаві висновки щодо даного датасету.

- Область застосування: моделі навчання з наглядом широко використовуються для передбачення поведінки ринків, фільтрації спаму, виявлення шахрайства, прогнозування цін та погоди. З іншого боку, навчання без нагляду підходить для виявлення аномалій, розділення ринку і візуалізації.

- Складність: навчання під наглядом — це простий метод машинного навчання для якого використовуються різні програмні інструменти або методи з обчислювальних пакетів, зокрема розроблених для мови Python. У навчанні без нагляду потрібні потужні засоби для роботи з великими обсягами нерозмічених датасетів. Моделі навчання без нагляду є обчислювально складними, оскільки для отримання очікуваних результатів необхідно зібрати великі набори даних.

- Недоліки: тренування моделей під наглядом є витратним з точки зору часу процесом, а мітки для вхідних і вихідних змінних вимагають експертизи. Крім того, методи навчання без нагляду можуть дати вкрай неточні результати, якщо не залучити аналітика для перевірки вихідних змінних.

Оскільки за допомогою аналізу джерел було визначено три основні методи досліджень в EDM: регресія, класифікація, кластеризація (рис. 2.1), далі буде розглянуто саме їх.

2.1.1. Регресія

Регресією (regression) називать метод DM, що використовується для передбачення чисельного значення (continuous) в датасеті, наприклад, для передбачення цін товарів та послуг, та є цінним інструментом для аналізу трендів та фінансових прогнозів. Оскільки регресія є методом машинного навчання з наглядом (supervised learning), тому для її тренування потрібен датасет з розміченими даними (labeled data). Цей метод дозволяє дослідити залежність між вхідними та цільовими змінними та чисельно визначити вплив вхідних змінних на цільову [26].

Регресія забезпечує хороший спосіб прогнозування змінних, але існують певні обмеження та припущення, наприклад незалежність змінних, властиві нормальні розподіли змінних. Якщо ми розглядаємо дві змінні, X1 і X2, і їх спільний розподіл є двовимірним розподілом, у цьому випадку ці дві змінні можуть бути незалежними, але мати кореляцію, тому граничні розподіли X1 і X2 необхідно вивести та використовувати. Перш ніж застосовувати регресійний аналіз, дані необхідно ретельно вивчити та виконати певні попередні тести, щоб переконатися, що регресія застосовна. У таких випадках доступні непараметричні тести (наприклад, random forest regression).

Суть регресії полягає у наближенні прямою або кривою точок з датасету таким чином, щоб відстань між точками даних та кривою була якнайменьшою.

Найбільш поширеними видами регресії є:
- Linear Regression;
- Polynomial Regression;
- Logistic Regression;
- Multi-Layer Perceptron;
- Random Forest Regression

Linear Regression є видом регресії, яка встановлює лінійне відношення між цільовою та незалежними змінними за допомогою прямої. Формула (2.1) відображає математичний сенс лінійної регресії.

$$Y = w_0 + w_1 * X \tag{2.1}$$

де w_0 – вільний член,
w_1 – кутовий коефіцієнт прямої,
X – незалежна (вхідна) змінна,

Y – залежна (цільова) змінна.

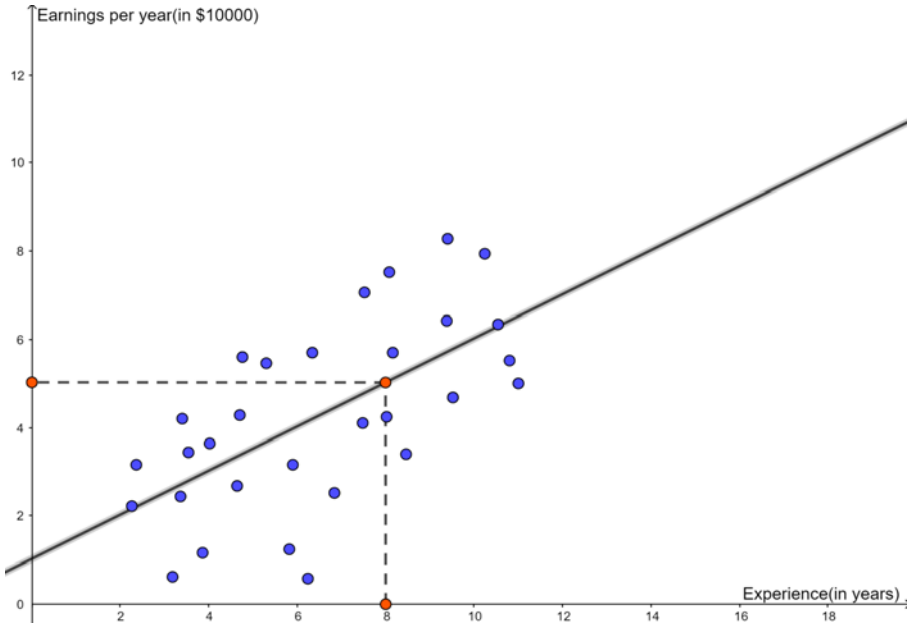


Рис. 2.2. Приклад лінійної регресії

У лінійній регресії найкраща лінія регресії, як показано на рис. 2.2 [35], досягається за допомогою методу найменших квадратів (МНК, LSE) шляхом мінімізації загальної суми квадратів відхилень від кожної точки даних до лінії регресії. В такому випадку позитивні та негативні відхилення не враховуються, оскільки всі відхилення зводяться у квадрат.

Polynomial Regression використовується тоді, коли незалежні змінні мають порядок > 1 і математичний сенс приймає такий вигляд, як показано на формулі (2.2). Оскільки наведена формула є поліномом, то найкращою лінією регресії буде крива підлаштована до всіх точок даних, як показано на рис. 2.3 [37].

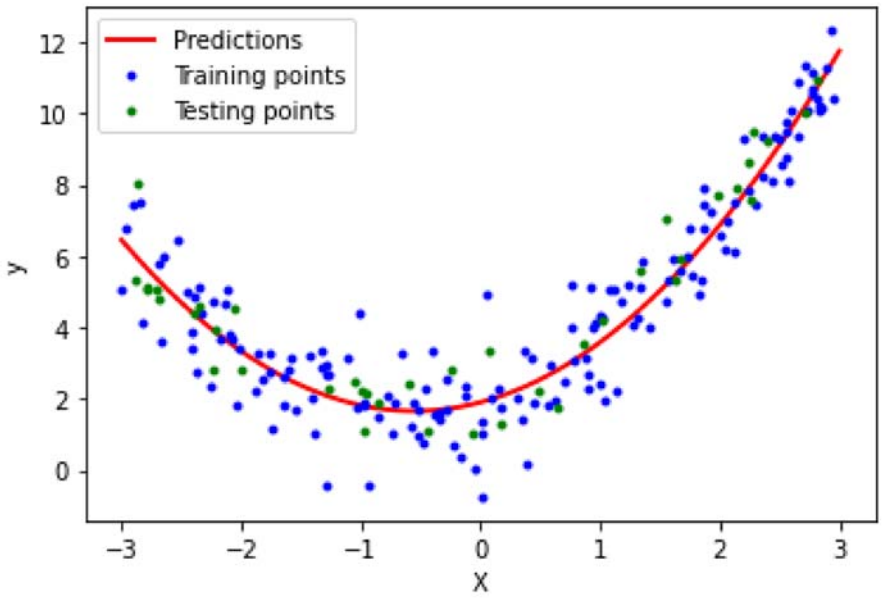


Рис. 2.3. Приклад поліноміальної регресії

$$Y = w_0 + w_1 * X + w_2 * X^2 \tag{2.2}$$

При використанні такого методу необхідно уникати проблеми overfitting, за допомогою методу регулярізації, який не дасть змоги коефіцієнтам приймати занадто великих значень.

Logistic Regression (LR) використовується лише в тому випадку, коли залежна змінна є бінарною, тобто приймає значення 0 або 1 [18,26]. В цьому методі цільова змінна (Y) коливається від 0 до 1, це в основному використовується для задач на основі класифікації (бінарної або мультикласової) і відображає вірогідність приналежності вхідних даних до певного класу.

Математичний сенс логістичної регресії показано на формулі (2.3) та рис. 2.4. Слід зауважити, що логістична регресія є простою моделлю нейрону штучної нейронної мережі (ANN).

$$Y = \sigma(w^T X) = \frac{1}{1 + e^{-w^T X}} \tag{2.3}$$

де w – вектор параметрів,
X – незалежна (вхідна) змінна,
Y – залежна (цільова) змінна.

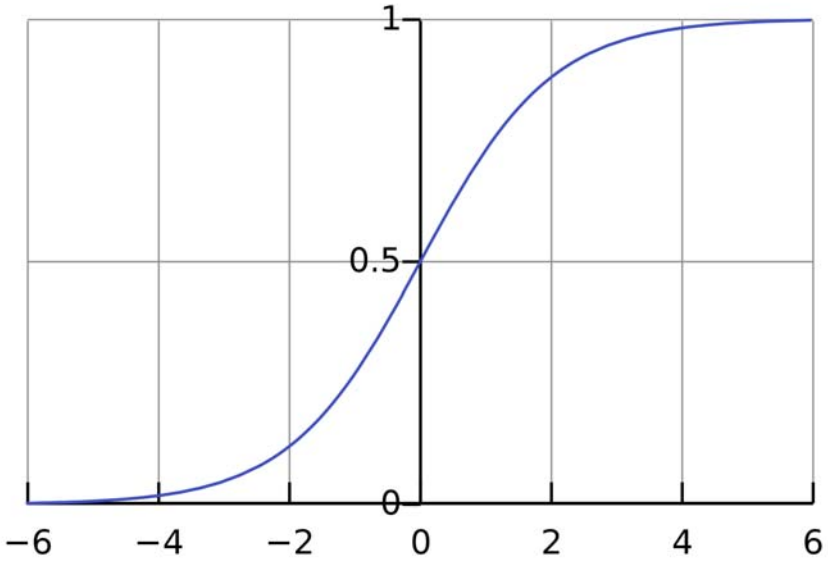


Рис. 2.4. Logistic Function

Multi-Layer Perceptron (MLP). Багатошаровими персептронами називають нейронні мережі прямого поширення (feed-forward neural network) (рис. 2.5) з однаковими функціями активації в кожному шарі, за виключенням вихідного, а також кількістю нейронів в кожному прихованом шарі. Багатошаровий персептрон складається з набору вхідних нейронів, який утворює вхідний шар; декількох прихованих шарів та одного вихідного шару [18].

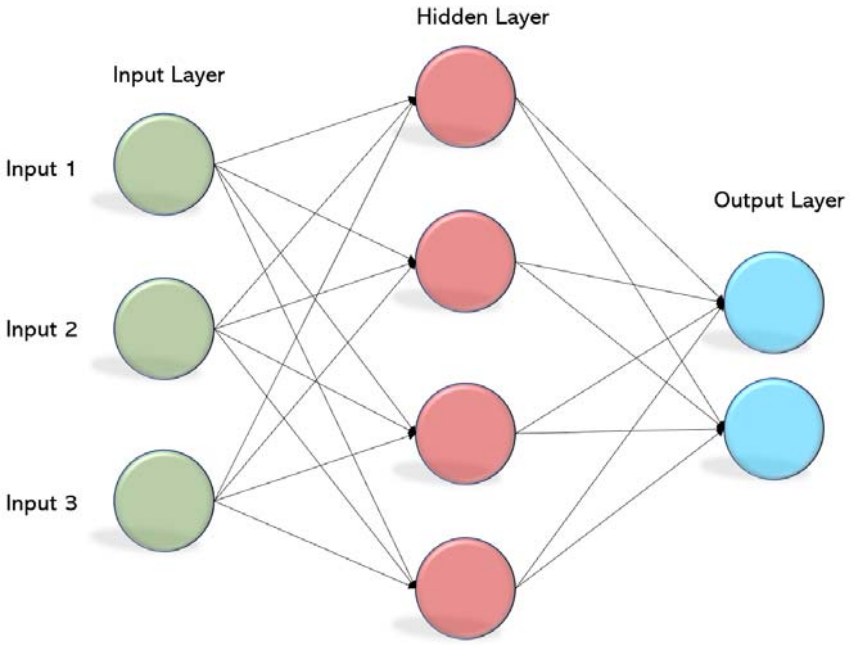


Рис. 2.5. Multilayer Perceptron

Random Forest Regression —- це алгоритм навчання з наглядом, який ви-

користовує метод ансамблевого навчання для регресії. Метод ансамблевого навчання – це техніка, яка поєднує прогнози з кількох алгоритмів машинного навчання, щоб зробити прогноз більш точним, ніж одна модель [36]. Кожен вузол намагається розділити вибірку на підвибірки, які найбільше відрізняються.

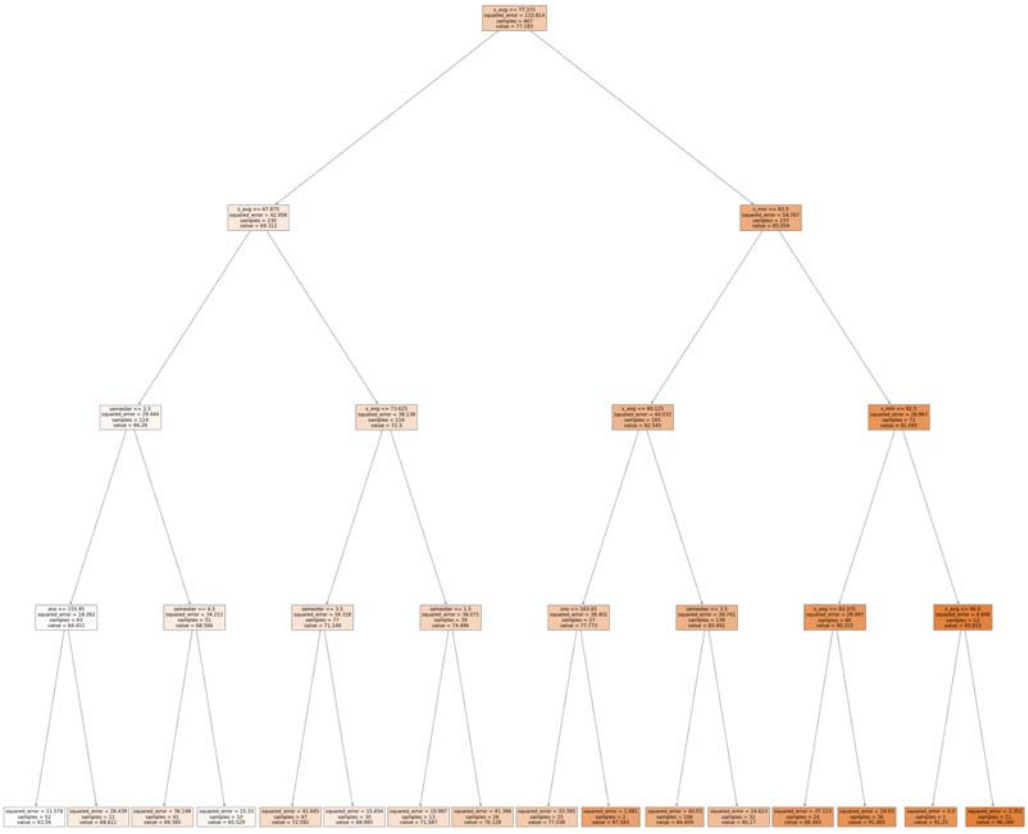


Рис. 2.6. Random Forest Regression

На рис. 2.6 показано структуру випадкового лісу. Випадковий ліс працює шляхом побудови кількох дерев рішень під час навчання та виведення середнього значення класів як передбачення всіх дерев. Далі наведено покрокову роботу алгоритму Random Forest:

1. Обрати випадково k точок даних із навчального набору.
2. Побудувати дерево рішень, пов'язане з цими k точками даних.
3. Обрати кількість N дерев для побудови і повторити кроки 1 і 2.
4. Для нової точки даних кожне з ваших N-дерев має передбачити значення y для відповідної точки даних і призначити новій точці даних середнє значення для всіх передбачених значень y.

Модель Random Forest Regression є потужною та точною. Зазвичай вона відмінно працює з багатьма проблемами, включаючи функції з нелінійними зв'язками.

До недоліків, однак, можна віднести наступне: відсутність інтерпретації, схильність до overfitting, необхідність в обранні правильних гіперпарамерів, зокрема кількості дерев, що включаються в модель.

SVM Regression (SVR). Метою методу опорних векторів є знайти гіперплощину в n-вимірному просторі, яка чітко розділяє точки даних. Точки даних по обидва боки від гіперплощини, які є найближчими до гіперплощини, називаються опорними векторами. Вони впливають на положення та орієнтацію гіперплощини і, таким чином, допомагають побудувати SVM.

Регресія методом опорних векторів — це алгоритм навчання під наглядом, який використовується для прогнозування дискретних значень. Регресія опорними векторами використовує той же принцип, що й SVM. Основна ідея SVR полягає в тому, щоб знайти найбільш підходящу лінію. У SVR найкраще підходить гіперплощина, яка має максимальну кількість точок.

На відміну від інших моделей регресії, які намагаються мінімізувати похибку між реальним і прогнозованим значенням, SVR намагається підібрати найкращу лінію в межах порогового значення. Порогове значення — це відстань між гіперплощиною та граничною лінією.

Отже, регресія застосовується для задач прогнозування чисельних значень та налічує багато методів доступних для використання. Зважачючи на результати попереднього аналізу публікацій в цьому дослідженні буде продемонстровано застосування Random Forest Regression, Multilayer Perceptron, SVM Regression та Linear Regression.

2.1.2. Класифікація

Класифікація полягає у визначенні категорії або мітки класу нового екземпляру даних (label), таким чином класифікація є методом навчання з наглядом. Спочатку набір даних поділяється на піднабори для тренування моделі і для тестування. Датасет для класифікації даних містить вхідні дані та пов'язані з ними мітки класів. Використовуючи датасет для тренування, алгоритм тренує модель або класифікатор. Отримана модель може бути деревом рішень, математичною формулою або нейронною мережею. У класифікації, коли моделі надаються немарковані дані, вона повинна знайти клас, до якого вона належить. Власне для оцінювання точності моделі в такому випадку і

використовується тестовий датасет.

Для ефективного процесу моделювання класифікація потребує навчального набору даних з багатьма екземплярами вхідних і вихідних даних, на яких можна тренувати модель.

Модель використовує навчальний набір даних і обчислює, як найкраще конвернувати набори вхідних даних у певні мітки класів. Таким чином, навчальний набір даних повинен бути достатньо репрезентативним для проблеми (збалансованим) та мати кількість прикладів мітки кожного класу відповідно до співвідношень, які спостерігаються в предметній галузі.

За кількістю міток класів класифікацію поділяють на:

- Бінарна класифікація – класифікація з кількістю міток = 2.

- Мультикласова класифікація – класифікація з кількістю міток > 2.

Далі були розглянуті поширені алгоритми класифікації.

Decision Tree (DT). Дерева рішень – це непараметричний метод навчання з наглядом, який використовується для класифікації та регресії. Мета побудови дерева полягає в створенні моделі для прогнозування значення цільової змінної за рахунок створення дерева, кожним вузлом якого є критерій який найсильніше розділяє дані. Оскільки кожен нижній лист дерева приймає якесь дискретне значення, побудоване таким шляхом дерево можна розглядати як кусково-постійне наближення. Методом класифікації деревом рішень є виконується в два етапи, які полягають у 1) будівництві дерева, і 2) обрізка. Обрізка — це техніка стиснення даних у машинному навчанні та алгоритмах пошуку, яка зменшує розмір дерев рішень шляхом видалення некритичних і зайвих для класифікації екземплярів розділів дерева. Обрізка зменшує складність остаточного класифікатора, а отже, покращує точність прогнозування за рахунок зменшення overfitting.

Аналіз дерева рішень особливо використовується для визначення стратегій, на основі яких робляться висновки щодо струкутури даних і впливу вхідних змінних на цільову змінну. Дерево рішень – це інструмент підтримки прийняття рішень, який використовує дерева для створення моделей, заснованих на критеріях розбиття інформації. Одержання інформації визначає чистоту даних, це означає, що якщо вузол дуже нечистий, то він складний. Ентропія вказує на перевагу ознаки для розрізнення цінності класу.

Внутрішні вузли дерева представляють критерії, зовнішні вузли або листки представляють мітки класів, а гілки з вузлів представляють результати проходження тестів або умов. Дерево рішень (DT) є одним із відомих алгори-

тми, що використовуються для прогнозного моделювання на освітних даних [16]. Наприклад, використано дані про поточну успішність у навчанні та попередню освіту, як особливості прогнозування успішності учнів у середніх школах [21]. Дерево рішень також дає хорошу точність щодо виявлення студентів, які потребують своєчасної допомоги для завершення їх навчання[38], за рахунок використання даних з систем електронного навчання, соціальних мереж та академічної успішності.

Деякі переваги дерев рішень:

- Простота інтерпретації за рахунок візуалізації дерев.

- Вимагає невеликої підготовки даних. Інші методи часто вимагають нормалізації даних, створення фіктивних змінних і видалення пустих значень, хоча цей модуль не підтримує пропущені значення.

- Обчислювальна складність використання дерева (тобто прогнозування даних) є логарифмичною кількістю точок даних, які використовуються для навчання дерева.

- Здатність обробляти як числові, так і категоріальні змінні.

- Здатність вирішувати проблеми з кількома вихідними змінними.

- Використовується модель white-box. Якщо дана ситуація спостерігається в моделі, умови легко пояснюються булевою логікою. Навпаки, у моделі black-box (наприклад, у штучній нейронній мережі) результати можуть бути складнішими для інтерпретації.

- Можливість перевірити модель за допомогою статистичних тестів, яка дозволяє оцінити точність моделі.

- Висока стійкість до outlier'ів.

До недоліків дерев рішень можна віднести:

- Створення надскладних дерев схильно до проблеми overfitting. Задля уникнення такої проблеми можна застосовувати гіперпараметри, зокрема обмеження глибини дерева.

- Нестабільність дерев рішень виникає через невеликі варіації в даних, які можуть призвести до створення зовсім іншого дерева. Ця проблема вирішується використанням дерев рішень в ансамблі.

- Прогнози дерев рішень не є ні гладкими, ні безперервними, а кусковопостійними наближеннями, тому вони погано вміють екстраполювати.

- При створенні дерева є необхідність в збалансуванні набору даних перед створенням моделі.

Naive Bayes – це класифікаційний метод, який приймає за основу прин-

цип умовної незалежності класу за теоремою Байєса. Це означає, що наявність однієї ознаки не впливає на наявність іншої у ймовірності даного результату, і кожен предиктор має однаковий вплив на цей результат.

Простий байєсівський алгоритм, або наївний байєсівський алгоритм, є методом інтелектуального аналізу, заснованим на принципах теорії ймовірності. У наївній байєсівській структурі немає інших дозволених відношень, і всі атрибути вважаються незалежними один від одного. Він працює на основі теореми Байеса (2.4) з припущеннями про незалежність між предикторами. Наївний алгоритм Байєса припускає, що вплив, який функції роблять на даний клас, не залежить від значення інших ознак.

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \tag{2.4}$$

де A та B – це події, причому $P(B) \neq 0$;

$P(A)$ – це апріорна вірогідність події A;

$P(A|B)$ – це апостеріорна вірогідність, тобто вірогідність коли експериментальні дані вже враховано.

Random Forest (RF) – це один із поширених методів машинного навчання, що полягає у використанні ансамблю дерев рішень [18]. Застосовується для задач класифікації, регресії і кластеризації. Дерево рішень будується з використанням навчальної вибірки та поняття ентропії. На кожному вузлі обирається один атрибут з даних, який найефективніше ділить навчальну множину на підмножини, що найсильніше розрізняються.

Support Vector Machine (SVM) – це один із методів машинного навчання з наглядом для регресії та класифікації, хоча краще він підходить для класифікації [18]. Метою даного методу є знайти гіперплощину в N-мірному просторі (рис. 2.7), яка найчіткіше розділяє дані, де розмірність залежить від кількості вхідних характеристик. Розміщення гіперплощини визначається підмножиною точок даних, відомих як опорні вектори.

Artificial Neural Network (ANN) в основному використовуються при розробленні алгоритмів глибокого навчання, за рахунок створення взаємозв'язків подібно людського мозку через шари нейорнів(вузлів). Кожен нейрон складається з входів, ваг,та функції активації, яка генерує вихідне значення і передає дані наступному шару в мережі. Процес навчання нейронних мереж полягає в мінімізації функції помилки (loss function) коригуючи ваги через процес градієнтного спуску. Коли мінімізація функції помилки досягає гло-

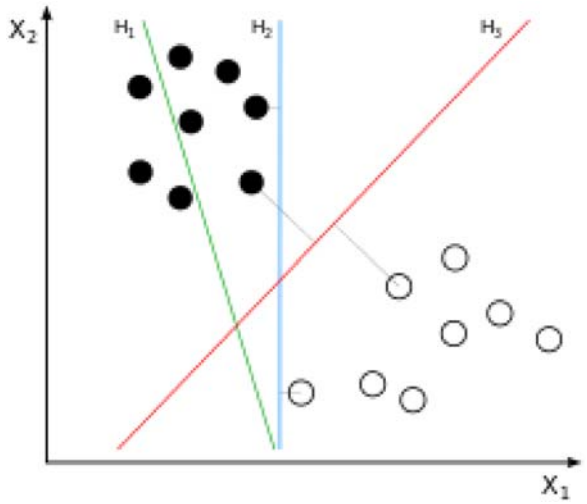


Рис. 2.7. Support Vector Machine

бального мінімуму, можна казати про досягнення найкращої точності моделі [25].

2.1.3. Кластеризація

K-Means – метод упорядкування кластерів в однорідні групи [21]. Суть полягає в тому що, коли ми маємо масив спостережень (об'єктів), кожен з яких має певні значення по ряду ознак, відповідно до цих значень об'єкт розташовується у багатовимірному просторі.

Алгоритм полягає в наступному:

1. Дослідник визначає кількість кластерів, що необхідно утворити

2. Випадковим чином обирається k спостережень, які на цьому кроці вважаються центрами кластерів

3. Кожне спостереження «приписується» до одного з n кластерів — того, відстань до якого найкоротша

4. Розраховується новий центр кожного кластера як елемент, ознаки якого розраховуються як середнє арифметичне ознак об'єктів, що входять у цей кластер

5. Відбувається така кількість ітерацій (повторюються кроки 3-4), поки кластерні центри стануть стійкими (тобто при кожній ітерації в кожному кластері опинятимуться одні й ті самі об'єкти), дисперсія всередині кластера буде мінімізована, а між кластерами — максимізована

Ключовим моментом є обрання кількості кластерів з огляду на оптимальну оцінку розбиття даних і, що є критично важливим, змістом предметної області, оскільки розбиття має давати не тільки оптимальну оцінку, а й знання відповідно до яких можна робити висновки щодо проведеної класте-

ризації.

Meanshift підпадає під категорію непараметричного алгоритму кластеризації на відміну від навчання без нагляду, яке призначає точки даних кластерам ітераційно шляхом зміщення точок у напрямку до центру плотності (центр плотності — це найвища щільність точок даних у регіоні). Таким чином, він також відомий як алгоритм пошуку режиму.

Враховуючи набір точок даних, алгоритм ітераційно призначає кожну точку даних до найближчого центроїда кластера, а напрямок до найближчого центроїда кластера визначається тим, де знаходиться більшість точок поблизу. Таким чином, кожна ітерація кожна точка даних буде переміщатися ближче до того місця, де знаходиться більшість точок, що є або приведе до центру кластера. Коли алгоритм зупиняється, кожна точка приписується кластеру.

На відміну від популярного алгоритму кластерів K-Means, середній зсув не вимагає вказувати кількість кластерів заздалегідь. Кількість кластерів визначається алгоритмом щодо даних.

2.2. Методи оцінки точності моделей

Після застосування вищезазначених методів необхідно буде провести порівняльний аналіз за допомогоюо спеціальних метрик: mean-squared-error (MSE), R2 score та F1 score [17].

Mean-squared-error, або середньоквадратична похибка – метрика, яка вимірює середнє значення квадратів похибок, як показано на формулі 2.5, і оскільки вона заснована на Евклідовій відстані, найкращий алгоритм матиме таку похибку близькою до 0. Метрика використовується для оцінки алгоритмів регресії.

$$MSE = \frac{1}{n}\sum_{n=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.5}$$

де Y_i – очікуване значення,

$\hat{Y}_i$ – передбачене моделлю значення.

R2 метрика є мірою відповідності для моделей лінійної регресії. Ця статистика вказує на відсоток дисперсії залежної змінної, яку незалежні змінні разом пояснюють. R-квадрат вимірює міцність зв'язку між вашою моделлю та залежною змінною за зручною шкалою від 0 до 100%, де більша оцінка відповідає кращому методу.

Davies–Bouldin Index. Оцінка визначається як середня міра подібності кожного кластера з його найбільш схожим кластером, де подібність – це відношення відстаней всередині кластера до відстаней між кластерами. Таким чином, кластери, які віддалені один від одного і менш розсіяні, призведуть до кращого результату. Мінімальний бал дорівнює нулю, а нижчі значення вказують на кращу кластеризацію.

F1-score поєднує точність (precision) і повноту (recall) класифікатора в одну метрику, беручи їх середнє гармонійне значення. В основному він використовується для порівняння ефективності двох класифікаторів. Якщо класифікатор A має вищу повноту, а класифікатор B має вищу точність, то у цьому випадку для обох класифікаторів можна використовувати метрику F1, щоб визначити, який із них дає кращі результати.

3. ОПИС МЕТОДИКИ ДОСЛІДЖЕНЬ

3.1. Обґрунтування вибору технологій та засобів розробки

Хоча при аналізі джерел матеріали статей демонстрували застосування різних інструменти для проведення досліджень, такі як WEKA, MATLAB, Microsoft SQL Server тощо, в цьому проєкті було обрано використання мови Python 3, разом з пакетами pandas(для роботи з даними), sklearn (для цілей кластеризації, класифікації, побудови дерев рішень тощо). Такий вибір зумовлений завданням завершити дослідження, шляхом обертання моделі в програмну оболонку.

3.1.1. Ubuntu

Ubuntu є безкоштовною операційною системою з відкритим кодом на базі ядра Linux, яка вільно розповсюджується як для професійних користувачів так і для аматорів. Великою перевагою є наявність великої спільноти користувачів та різних форумів, що освітлюють різні поширені та вузькоспеціалізовані аспекти системи та особливості її налаштування. Основні ідеї на яких базується спільнота Ubuntu викладені у Маніфесті, зокрема доступність програмних засобів, гнукість програмного забезпечення, кастомізація операційної системи та підтримка більшості мов програмування [30].

Головні ідеї ОС:

- Ubuntu є безкоштовною операційною системою, тому корпоративна версія має такий же широкий функціонал, як і базова версія і побидві версії поставляються на рівних умовах;

- Ubuntu є підтримцє різні локалізації, щоб популярізувати культуру вільного ПЗ та надавати доступні засоби для розробки;

- Дистрибутиви Ubuntu поділяються на стабільні та регулярні версії: стабільна верчія підримується 5 років, інші – 9 місяців.

3.1.2. Python

Python — інтерпретована мова програмування високого рівня. Строга динамічна типізація дозволяє швидко і просто створювати як професійні програмні продукти, так і мінімально працюючі продукти.

Основними перевагами є:

- потужний функціонал стандартної поставки, який включає в себе велику кількість модулів, що реалізують більшість базових і рутинних завдань;

- інтерпретатор добре підходить для експериментів та вирішення простих задач;

- зручний для розв'язання математичних проблем (має засоби роботи з комплексними числами, може оперувати з цілими числами довільної величини, у діалоговому режимі може використовуватися як потужний калькулятор);

- велике ком'юніті та велика кількість модулів розширення.

Основними модулями для проведення даного дослідження є:

- Numpy – модуль для швидкої та ефективної обробки великих масивів, швидкість досягається за рахунок застосування векторизації, тобто масив обробляється не елемент за елементом, а більш раціональним шляхом;

- Scipy – розширення модуля numpy за рахунок додавання математичних методів оптимізації, дерев тощо;

- Pandas – потужний модуль для обробки датафреймів (таблиць з наборами даних);

- Matplotlib – візуалізація статичних, анімованих та інтерактивних 2- та 3-вимірних графіків;

- Sklearn (scikit-learn) – інструменти для інтелектуального аналізу даних.

3.1.3. Jupyter Notebook

Jupyter Notebook (відомий раніше як IPython) – це клієнт-серверний застосунок для створення, редагування і запуску документів, пов'язанних з обчисленнями та аналізом даних[34]. Формат Notebook дозволяє створювати документи які одночасно містять дані, які може прочитати людина(графіки, таблиці тощо), так і кодом для виконання аналізу даних. Цей формат дозволяє легко ділитися кодом, обмінюватися та демонструвати результати досліджень.

Інтерактивний браузерний інтерфейс застосунку дозволяє легко запускати обчислення, керувати змінними та відображати результати. Застосунок може бути розгорнуто як в локальному варіанті (на власному комп'ютері), так і на віддаленому сервері (з доступом через Інтернет).

Кожен Notebook має своє ядро (kernel), яке є "обчислювальним центром"даного документа. Поки воно активне обчислення виконуються, а по їх завершенню розробник має доступ до всіх змінніх середовища в оперативній пам'яті, або запустити іншу частину кода на виконання. Код в Jupyter пишеться в ячейках, кожну з який можна в будь-який момент виконати в довільному порядку, що робить Jupyter дуже гнучким інструментом, порів-

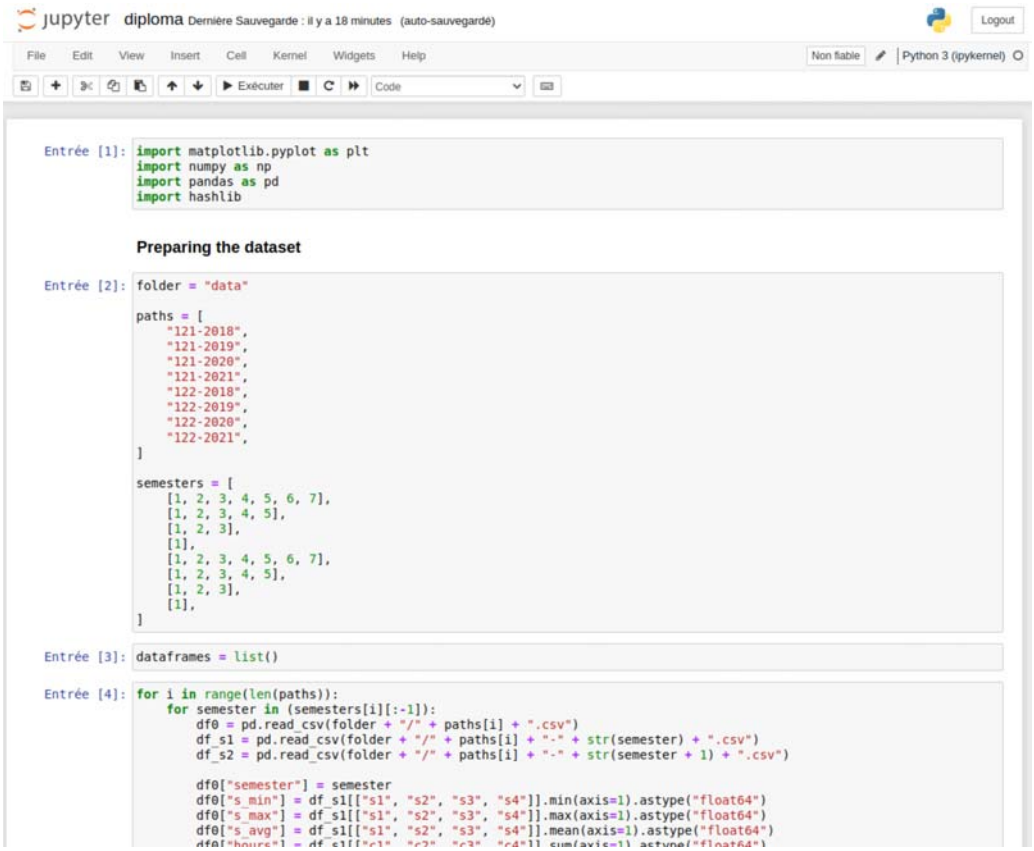


Рис. 3.1. Інтерфейс Jupyter Notebook

няно з іншими інтегрованими системами розробки. Jupyter підтримує більше 40 мов програмування і інтеграцію з різними модулями та пакетами, тому активно використовується розробниками і дослідниками по всьому світу.

3.1.4. Docker

Docker – це програмне забезпечення з відкритим кодом, найпопулярніша платформа для управління контейнерами [31]. Docker потрібний для більш ефективного використання системи і ресурсів, швидкого розгортання готових програмних продуктів, а також для їх масштабування і перенесення в інші середовища з гарантованим збереженням стабільної роботи. Коли розробляється додаток, потрібно надати код разом з усіма його складовими, такими як бібліотеки, сервер, бази даних тощо, тому можна опинитися в ситуації, коли додаток працює на вашому комп'ютері, але відмовляється вмикатися та працювати на пристрої іншого користувача.

Ця проблема вирішується через створення незалежності програмного забезпечення від системи. Docker розділяє ядро операційної системи на контейнери (Docker container), що працюють, як окремі процеси. Він вирішує безліч завдань, пов'язаних зі створенням контейнерів, розміщенням в них

додатків, управлінням процесами, а також тестуванням ПЗ і його окремих компонентів.

Docker допомагає:

- мінімізувати використання ресурсів;
- приховувати фонові процеси;
- легко масштабовувати проєкт;
- пришвидшити тестування продуктів;
- прискорити розгортання продуктів;
- прискорити розроблення продуктів.

Налаштування контейнеру Docker здійснюється шляхом створення Docker image – образу, на основі якого буде налаштовано пакети, код та змінні середовища контейнеру, що запускається.

Docker image — це шаблон, який містить набір інструкцій для створення Docker контейнера. Цей шаблон надає зручний спосіб розгортання програм і попередньо налаштованих серверних середовищ, які можливо використовувати для власного приватного використання або відкрити для інших користувачів Docker.

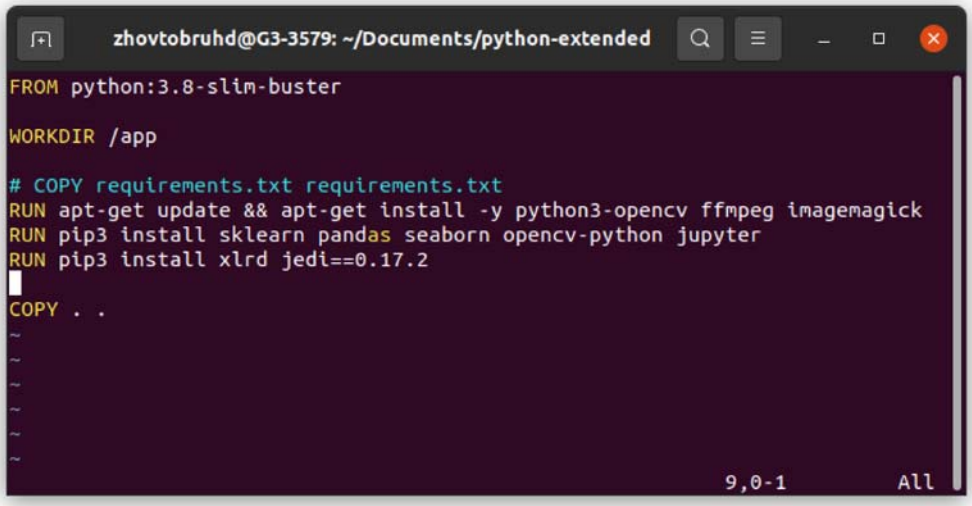


Рис. 3.2. Dockerfile

Docker image складається з набору файлів, які об'єднують усі необхідні елементи, такі як установочні файли, код програми та залежності, необхідні для налаштування повноцінного середовища контейнера. Docker image можливо створити за допомогою одного з двох методів:

- Інтерактивний: шляхом запуску контейнера з наявного образу Docker, ручне налаштування цього середовища контейнера за допомогою послідовно-

сті кроків і збереження отриманого стану як нового образу;

- Dockerfile: шляхом створення текстового файлу, відомого як Dockerfile, який надає специфікації для створення образу Docker, як показано на рис. 3.2

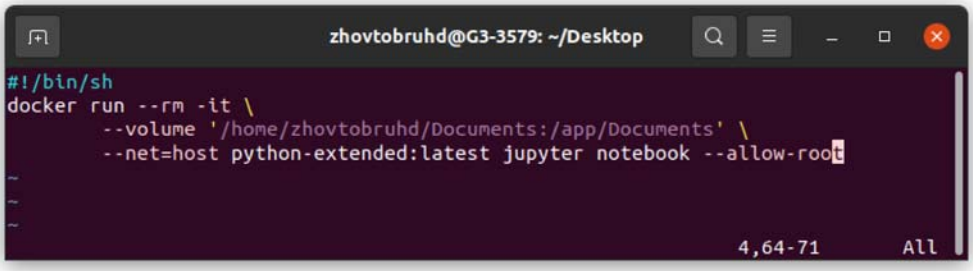


Рис. 3.3. Bash-скрипт для запуску контейнеру

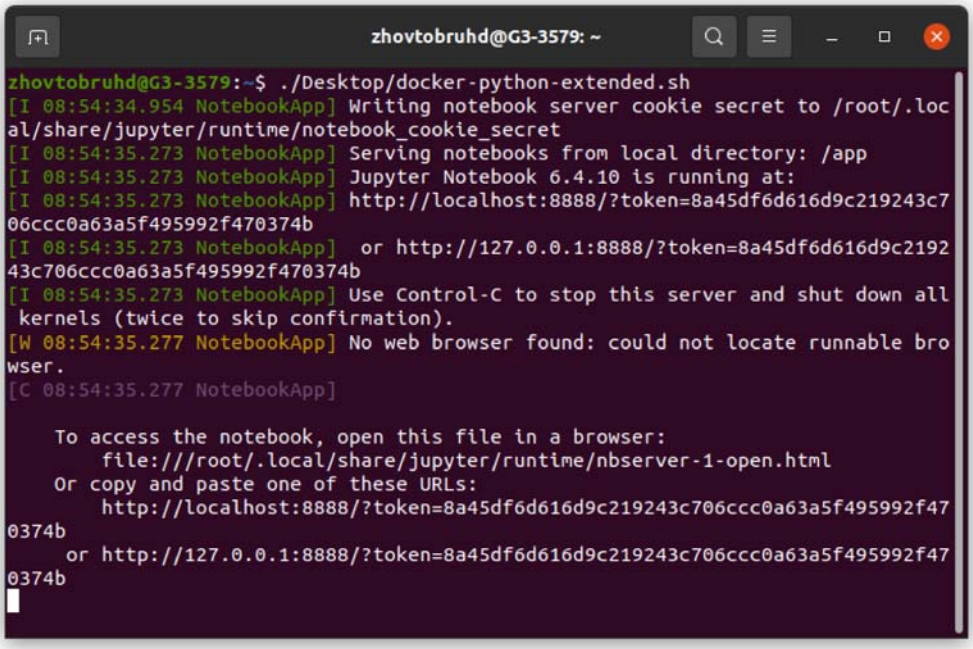


Рис. 3.4. Запуск контейнеру

Після того як було обрано попередньо створений образ або зібрано новий, необхідно запустити контейнер. Запуск контейнеру відбувається у командному рядку, причому передача різних параметрів визначає доступи для контейнеру, зокрема доступ до мережі та файлової системи хост-комп'ютера. Зазвичай відкриття доступу до файлової системи хост-комп'ютера є потенційно небезпечним тому в даному випадку доступ надається тільки до окремої папки. Доступ до мережі надається задля довстановлення необхідних пакетів у систему. Для того щоб не прописувати аргументи для запуску вручну,

команду для запуску контейнеру було обгорнуто у bash-скрипт, як показано на рис. 3.3.

Таким чином можно легко розгорнути середовище для досліджень за допомогою простої команди виконання скрипта і отримати результат, як на рис. 3.4

3.2. Дані

Набір даних (датасет) для цього дослідження було сформовано з Семестрових відомостей факультетету Інформаційних технологій ХНЕУ ім. С. Кузнеця та Єдиної державної електронної бази з питань освіти (ЄДЕБО) з застосуванням сервісу «Пошук абітурієнтів»[29]. Для датасету було обрано дані студентів 2018–2022 років, які навчаються за спеціальностями 121 «Інженерія програмного забезпечення» та 122 «Комп'ютерні науки», оскільки їх ОПП та предмети до вступу є максимально наближеними, що надасть змогу зберегти закономірності в даних при достатньому розмірі датасету.

Загалом в датасеті представлено інформацію про 449 студентів.

	4
name	Жовтобрюх Дмитро Андрійович
funding	1
zno_math	193.0
zno_ukr	189.0
zno_oth	186.0
gender	1
semester	1
s_min	93.0
s_max	100.0
s_avg	97.25
hours	690.0
pred	95.25
id	ee5c043c562a9d9d7ece1764eef1ebeb

Рис. 3.5. Приклад строки даних з датасету

На рис. 3.6 продемонстровано співвідношення жінок та чоловіків в зібраному датасеті. В цілому, можна вважати дане співвідношення відповідає співвідношенню жінок та чоловіків, що працюють в сфері IT і датасет відповідає реальним даним за ознакою статі [32].

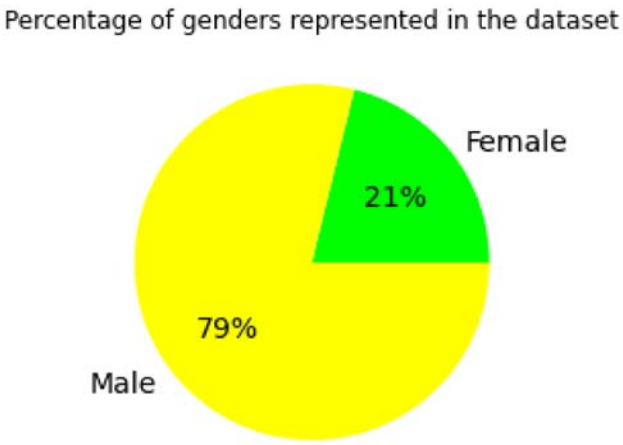


Рис. 3.6. Співвідношення гендерів представлених в датасеті

Для отримання попередніх знань про залежності в даних, було побудовано матрицю кореляції (рис. 3.7). Попередньо можна помітити значну кореляцію між цільовою змінною (pred) та мінімальною (s_min), середньою (s_avg) і максимальною (s_max) оцінкою за семестр.

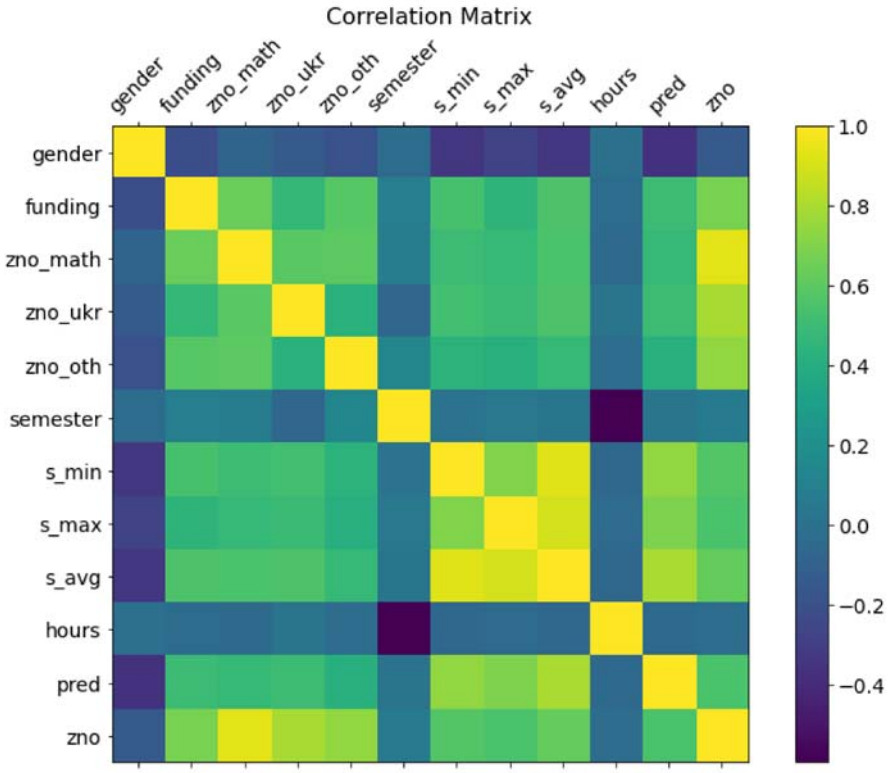


Рис. 3.7. Матриця кореляції

3.3. Оцінка ризиків

Потенційні ризики, що можуть виникнути під час роботи над проєктом:

• Мала кількість даних або фіч – модель працюватиме нестабільно, виникатиме underfitting або overfitting [20], проте можна зібрати більшу кількість інформації для тренування моделей.

• Розбалансований датасет – модель буде схильною до overfitting, можливо потрібно додати даних.

• Неправильно обрані фічі – модель навчається занадто довго, або погано працює на тестових даних.

• Неправильно обрана модель – низька точність роботи моделі, потрібно обрати іншу.

3.4. Обрання методології роботи

В ході аналізу джерел за обраною темою було знайдено наступну методологію (рис.3.8)[27].

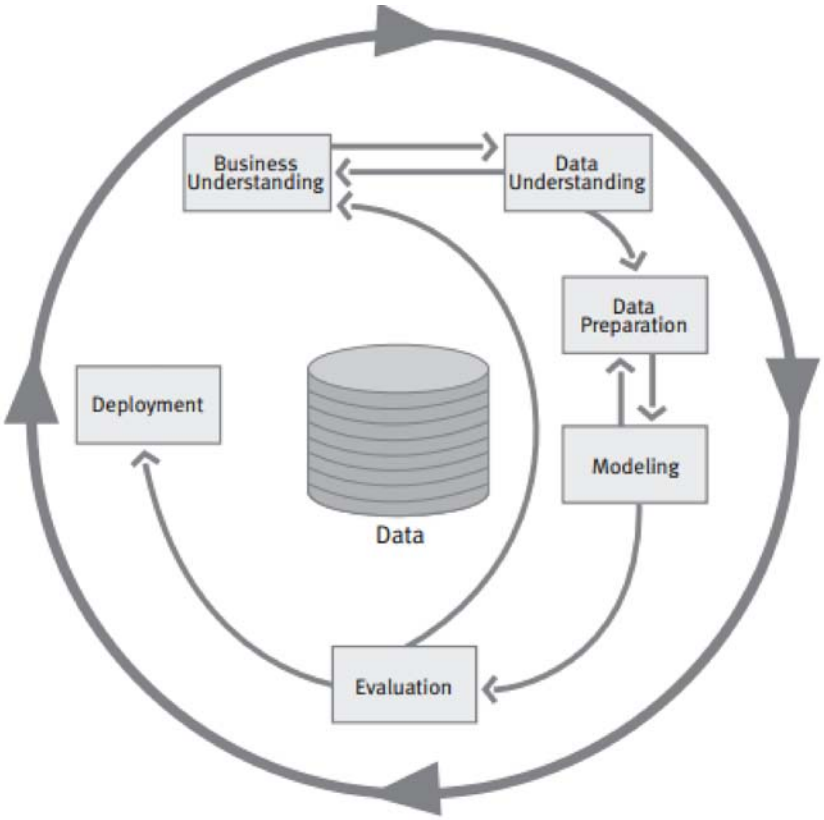


Рис. 3.8. CRISP-DM

Наведена модель процесу для DM демонструє життєвий цикл проекту DM. Вона містить фази проекту, їх відповідні завдання та взаємозв'язки між цими завданнями. Зв'язки можуть існувати між будь-якими завданнями інтелекту даних залежно від цілей передумови та інтересу користувача – і, головне, – до даних.

Життєвий цикл проекту інтелекту даних складається з шести фаз, як показано на рис. 3.8. Послідовність фаз не є жорсткою, а переміщення між різними фазами відбувається завжди на вимогу. Результат кожного етапу визначає, яка фаза, та яке конкретне завдання фази, має бути виконано наступним. Стрілки показують найважливіші та найчастіші залежності між фазами. Зовнішнє коло на рис. 3.8 символізує циклічність самого DM. DM не закінчується як тільки рішення розгорнуто. Знання, отримані під час процесу і з розгорнутого рішення можуть запускати нові, часто більш цілеспрямовані ділові процеси, подальші дані процесів DM покращуються від досвіду попередніх.

Далі наводиться коротка характеристика кожної фази на рис. 3.9 [27].

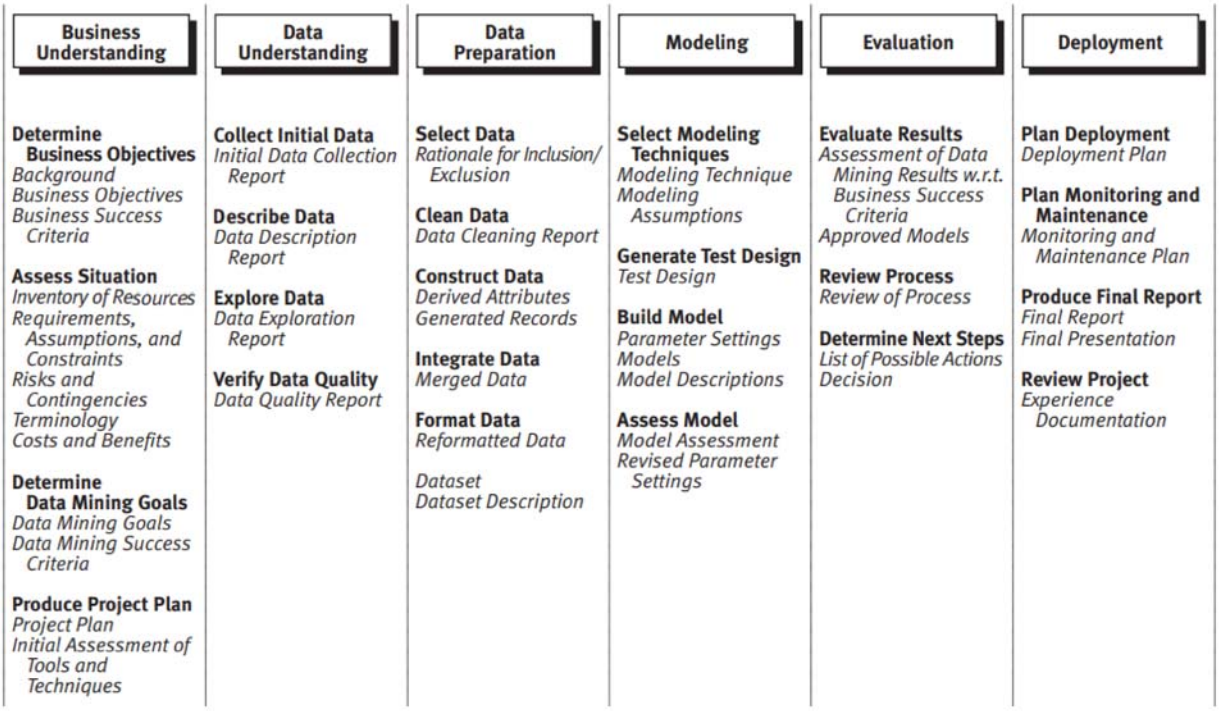


Рис. 3.9. Деталізація етапів CRISP-DM

Крім діаграми CRISP-DM, також існує діаграма методології DS яку зазвичай можна знайти в навчальних курсах IBM (рис.3.10) [28].

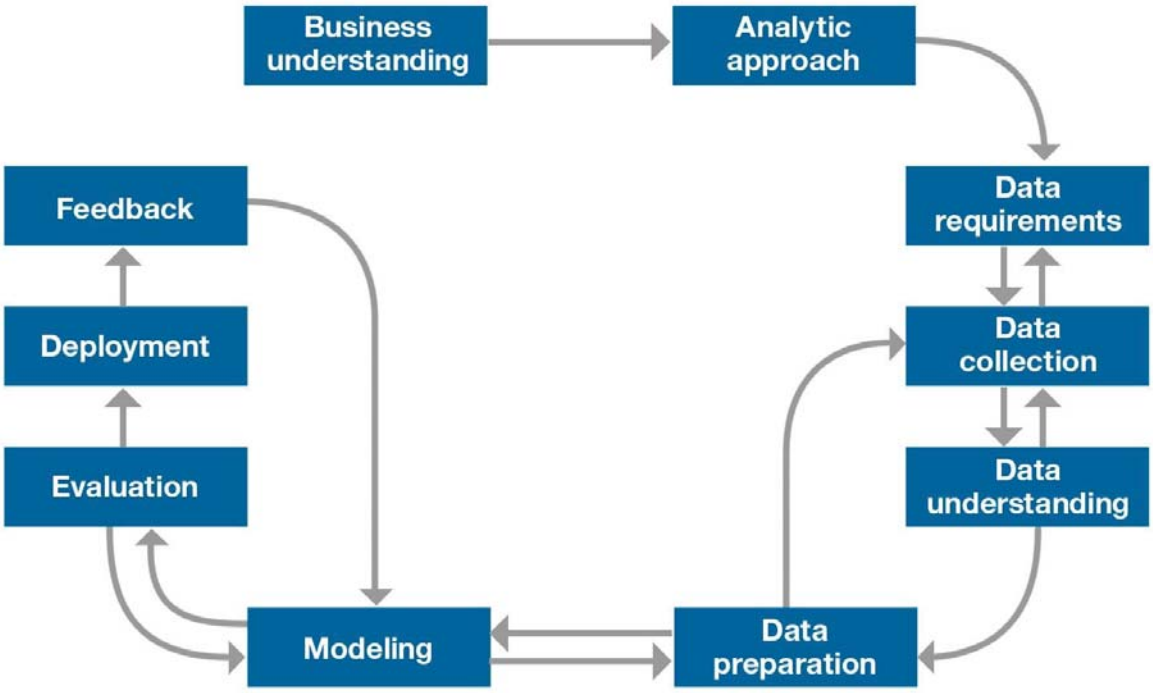


Рис. 3.10. DS methodology

Наведена схема є більш детальною і демонструє наочний варіант того, як в подальшому цей проєкт буде розвинутий. Слід зауважити, що для досягення задовільних результатів роботи моделі може знадобитися декілька проходів (ітерацій) за цією схемою. Це зумовлено тим, що визначення необхідних обсягів даних і правильний вибір моделі з першого разу навряд чи вдастся зробити повністю корректно.

Незважаючи на те, що методології трохи відрізняются, за схемою CRISP-DM буде в подальшому проводитися розробка проєкту через обмеженість ресурсів і масштабів дослідження.

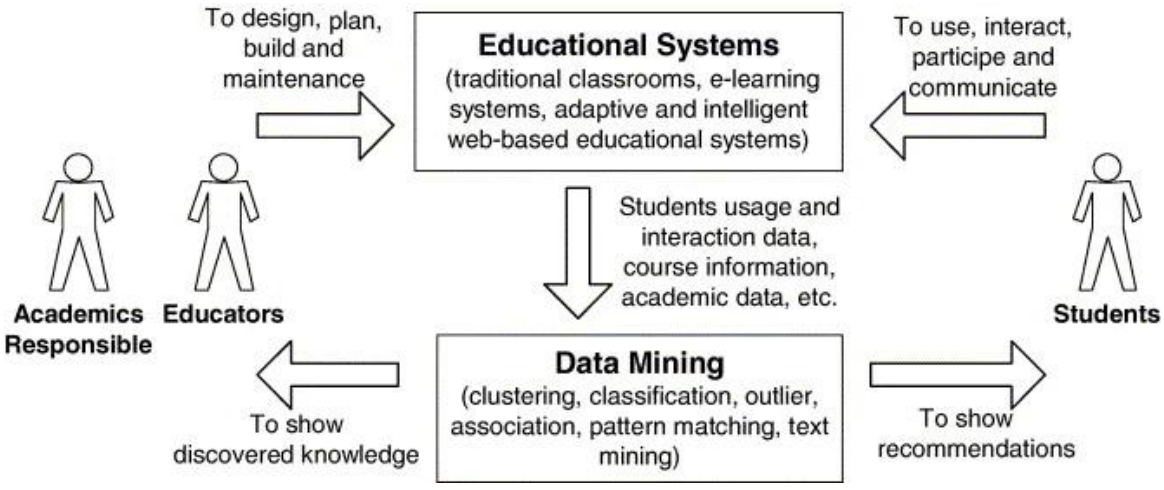


Рис. 3.11. Цикл використання DM в освітніх системах

Схема на цьому рисунку (рис.3.11) характеризує як саме наш готовий програмний модуль EDM буде працювати (взаємодіяти зі стейкхолдерами та учнями)[14]. Більшість дій, що виконуються студентами записуються в інформаційну систему ЗВО, потім ці дані експоруються з метою аналізу в наш програмний модуль і результати отримані таким чином мають бути проаналізовані, як і студентом так і керівництвом.

3.4.1. Business Understanding

За CRISP-DM методологією, першим завданням перед початком дослідження є отримання якйнайбільшого обсягу інформації та розуміння прикладного застосування даного дослідження в предметній галузі (визначити поточний стан проблеми, бізнес-цілі та ресурси).

В ході консультацій з керівником проєкту було визначено, що кожного року факультет приймає все більше студентів за спеціальностями 121 «Інженерія програмного забезпечення» та 122 «Комп'ютерні науки». Очікується, що збільшення кількості студентів може призвести до загострення певних системних проблем організації навчального процесу. В зв'язку з цим є необхідність у покращенні підходів до навчання та постійного дослідження поведінки студента по мірі того, як він виконує навчальний план.

Для цього пропонується дослідити дані на предмет можливої залежності між поточною та майбутньою успішністю студента, а також наявності закономірностей (поділення на умовні групи).

У якості ресурсів факультет має відомості про поточну успішність студентів, а також відомо, що про певну частину здатностей студента можна дізнатися спираючись на його результати зовнішнього незалежного оцінювання.

З точки зору бізнес-перспективи успіхом вважатиметься:

- створення моделі, яка використовує дані про поточну успішність, для передбачення успішності студента в наступному семестрі;

- присвоєння кожному студенту маркеру, на основі останніх даних про поточну успішність, для визначення студентів з задовільними та незадовільними академічними показниками.

3.4.2. Data Understanding

Важливим моментом при створенні датасету є тип змінної, який базується на тому які значення вона приймає:

- Continuous – змінна є неперервною і може приймати будь яке значення, наприклад бали ЗНО;

- Discrete – змінна є перервною і може приймати певні значення (зазвичай цілі) з діапазону, наприклад семестр навчання;

- Categorical – змінна приймає одно з двох значень, наприклад стать студента.

В ході розгляду бізнес-проблеми та аналізу наявних ресурсів було зібрано дані за ознаками наведеними в табл. 3.1.

Таблиця 3.1

Змінні, що досліджуються

Назва змінної	Значення змінної	Опис
gender	Categorical Female – 0 Male – 1	Стать студента
funding	Categorical Контракт – 0 Бюджет – 1	Джерело фінансування
zno_math	Continuous	Бали за ЗНО з математики
zno_ukr	Continuous	Бали за ЗНО з української мови
zno_oth	Continuous	Бали за ЗНО за третій предмет
semester	Discrete 1<=semester<=8	Номер семестру в якому студент навчається
s_min	Continuous	Мінімальна оцінка за профільну дисципліну у семестрі
s_max	Continuous	Максимальна оцінка за профільну дисципліну у семестрі
s_avg	Continuous	Середня оцінка за профільними дисциплінами у семестрі
hours	Continuous	Кількість годин за профільні дисципліни в семестрі
pred	Continuous	Середня оцінка у наступному семестрі

3.4.3. Data Preparation

Підготовка датасету – одна з найбільш витратних з точки зору часу процес дослідження [27]. Зазвичай, у аналізі даних власне дані групують в

набори – датасети, які вже потім легко модифікуються, оброблюються та використовуються алгоритмами обробки даних. Простими словами, датасет можна визначити як таблицю у якій кожна колонка відповідають якійсь змінній або ознаці предметної області (feature), а кожна строка – відповідає одному запису даних (record).

Одним з етапів підготовки датасету до використання з моделями є очищення датасету (data wrangling)[33]. Мета цього процесу визначити чи усі критично важливі для дослідження параметри заповнені належним чином. У випадку якщо в датасеті наявні пропуски (т.з. NaN-значення), використовується один з наступних методів:

- видалення строки з датасету;

- заповнення пропущеного значення середнім значенням по виборці або медіаною.

На рис. 3.12 можна побачити, що в студента в 7 рядку відсутні певні значення, що є прикладом даних з пропусками.

	id	gender	funding	zno_math	zno_ukr	zno_oth	semester	s_min	s_max	s_avg	hours	pred
0	af609c9a84cece367d5d54b287b8906b	1	1	193.0	182.0	178.0	1	81.0	100.0	89.50	690.0	88.00
1	2c7b91043b6ff937856aed4b2f7b3f07	1	0	166.0	161.0	146.0	1	60.0	82.0	74.00	690.0	72.00
2	151866af35d63facb5ad2150333b236c	1	0	138.0	135.0	171.0	1	61.0	77.0	67.00	690.0	68.75
3	478f3dce742612f00987242a808f38bb	1	0	186.0	181.0	185.0	1	81.0	96.0	90.25	690.0	87.25
4	ee5c043c562a9d9d7ece1764eef1ebeb	1	1	193.0	189.0	186.0	1	93.0	100.0	97.25	690.0	95.25
5	49c1e922887c0fed05e92d096220c58a	1	0	164.0	120.0	159.0	1	65.0	90.0	74.25	690.0	71.50
6	5fcdd7e7b7bb92d4c034a376dd51f7bc	1	0	168.0	131.0	158.0	1	66.0	81.0	72.50	690.0	73.00
7	6e10f3f0263598616cc674b332ec69be	1	0	138.0	155.0	162.0	1	NaN	NaN	NaN	690.0	NaN
8	4bca36b9c7a2e634d4292012153af276	1	0	150.0	173.0	136.0	1	60.0	88.0	74.00	690.0	71.00
9	1c9066e54b09ebb5a5cae1331e965ba0	1	1	200.0	181.0	193.0	1	91.0	100.0	97.00	690.0	93.25

Рис. 3.12. Дані з пропусками

Для вибору відповідного методу необхідно підрахувати кількість строк з пропущеними даними, як показано на рис. 3.13. Усього було знайдено 27 строк з пропущеними даними в датасеті з 1076 записів.

```
Entrée [25]: df_res.isnull().any(axis=1).sum()
    Out[25]: 27
```

Рис. 3.13. Підрахунок строк з пропущеними даними

Зважаючи на те, що кількість пропусків не є значною частиною від датасету, а про вплив тих чи інших параметрів на дослідження на даному етапі

не відомо було вирішено видалити з датасету строки з пропусками. Частина результату операції показана на рис. 3.14.

	id	gender	funding	zno_math	zno_ukr	zno_oth	semester	s_min	s_max	s_avg	hours	pred
0	af609c9a84cece367d5d54b287b8906b	1	1	193.0	182.0	178.0	1	81.0	100.0	89.50	690.0	88.00
1	2c7b91043b6ff937856aed4b2f7b3f07	1	0	166.0	161.0	146.0	1	60.0	82.0	74.00	690.0	72.00
2	151866af35d63facb5ad2150333b236c	1	0	138.0	135.0	171.0	1	61.0	77.0	67.00	690.0	68.75
3	478f3dce742612f00987242a808f38bb	1	0	186.0	181.0	185.0	1	81.0	96.0	90.25	690.0	87.25
4	ee5c043c562a9d9d7ece1764eef1ebeb	1	1	193.0	189.0	186.0	1	93.0	100.0	97.25	690.0	95.25
5	49c1e922887c0fed05e92d096220c58a	1	0	164.0	120.0	159.0	1	65.0	90.0	74.25	690.0	71.50
6	5fcdd7e7b7bb92d4c034a376dd51f7bc	1	0	168.0	131.0	158.0	1	66.0	81.0	72.50	690.0	73.00
8	4bca36b9c7a2e634d4292012153af276	1	0	150.0	173.0	136.0	1	60.0	88.0	74.00	690.0	71.00
9	1c9066e54b09ebb5a5cae1331e965ba0	1	1	200.0	181.0	193.0	1	91.0	100.0	97.00	690.0	93.25
10	8e5f5e9e99449828159abfea1ca7396d	1	1	195.0	165.0	185.0	1	82.0	99.0	87.75	690.0	89.25

Рис. 3.14. Очищені дані

3.4.4. Modeling

1. Регресія. Перед моделюванням при роботі з невідомим раніше датасетом необхідно визначити можливий вплив певних вхідних змінних на кінцевий результат. Як зазначено в попередньому розділі, з цією метою застосовуються непараметричні регресійні методи, в нашому випадку – Random Forest Regression. Слід згадати, що в результаті роботи методу отримується дерево, кожен вузол якого намагається якнайсильніше розділити вибірку, що до нього доходить за якимось критерієм, як показано на рис. 3.15.

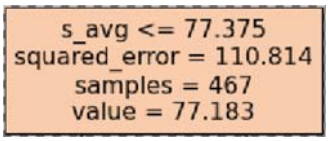


Рис. 3.15. Вузол дерева

Можна легко помітити, що таке представлення даних (рис. 3.16) зовсім нерепрезентативне, хоча теоретично дає змогу оцінити, які процеси відбуваються в такому дереві. З цією метою, модуль sklearn дозволяє отримати дані про кількісний вплив вхідних змінних на результат регресії, що продемонстровано на рис. 3.17.

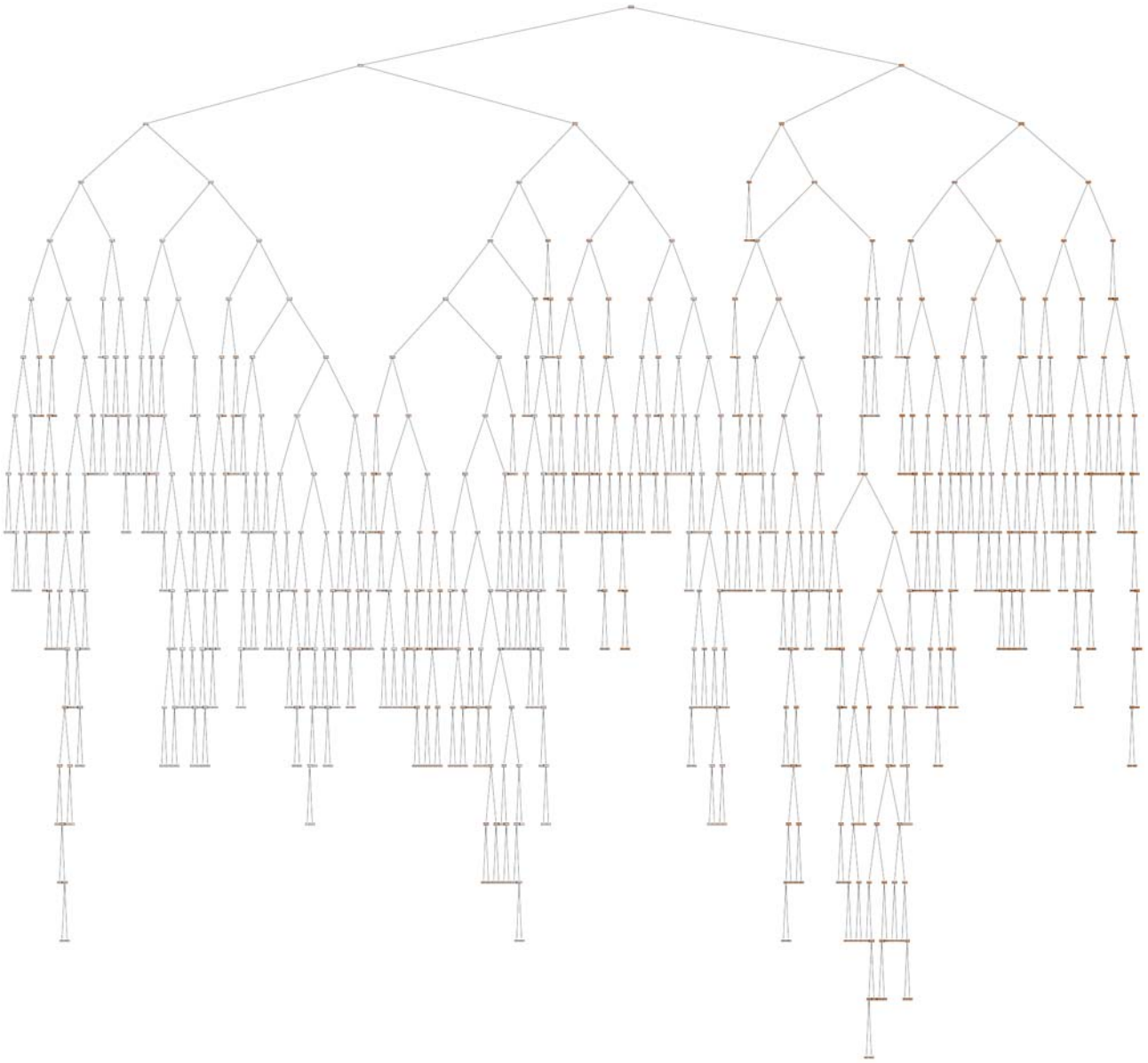


Рис. 3.16. Отримане дерево

	gender	funding	s_min	s_max	s_avg	zno_math	zno_ukr	zno_oth	hours	semester
1	0.011726	0.006685	0.042779	0.04185	0.689604	0.048017	0.053804	0.046224	0.026541	0.03277

Рис. 3.17. Кількісний вплив вхідних змінних на результат регресії

За цими даними (рис. 3.18), можемо зробити проміжний висновок, що більш за інших на успішність студента в наступному семестрі впливає його поточний середній бал, потім бали ЗНО та мінімальний і максимальний бал за предметами в поточному семестрі.

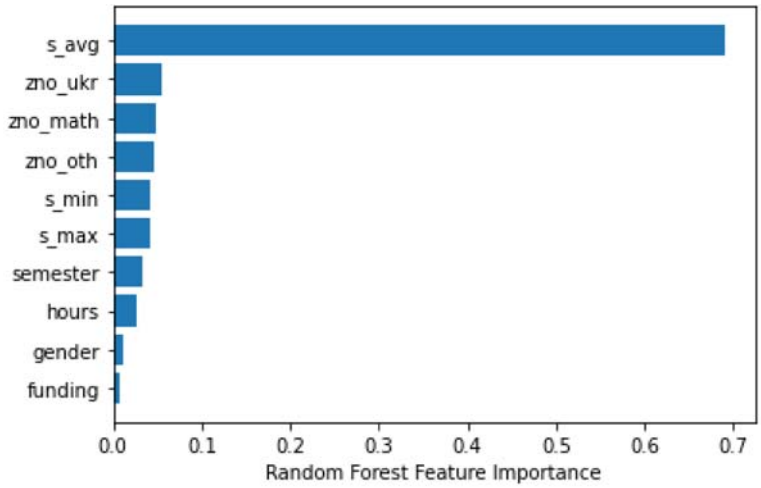


Рис. 3.18. Відсортований список впливів вхідних даних на результат регресії

2. Кластеризація. Для вирішення проблем кластеризації було обрано методи K-means та SVC.

Першочерговою і основною проблемою кластеризації є визначення кількості кластерів на які буде розділено датасет. В даному питанні необхідно витримати баланс між кількістю кластерів і їх значенням для предметної галузі, адже розділення на велику кількість малих кластерів хоча і дасть низьку помилку, але не матиме жодного змісту для предметної галузі.

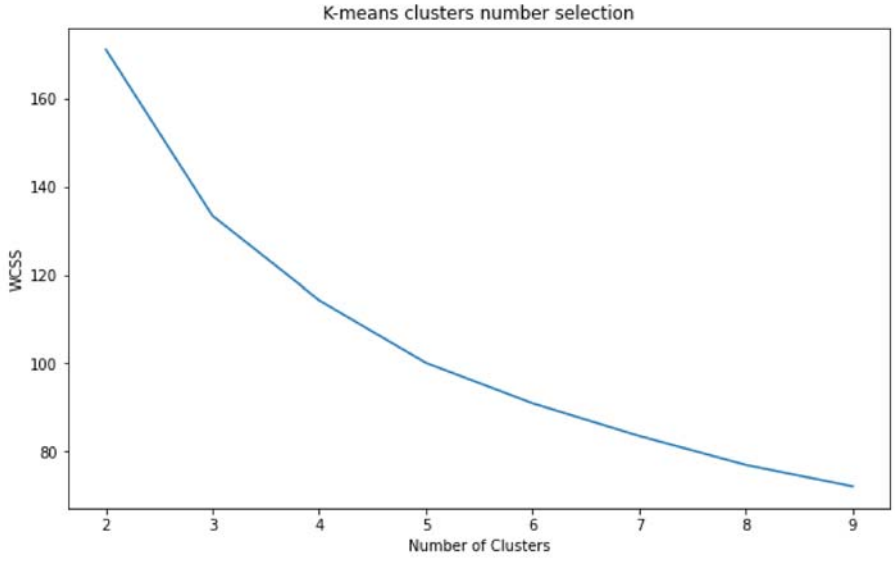


Рис. 3.19. K-means number of clusters selection

Обрання кількості кластерів у випадку k-means відбувається за допомогою правила ліктя (elbow rule), принцип якого полягає в тому, що на графіку залежності суми помилок алгоритму від кількості кластерів (рис. 3.19) спостерігається певний перелом, щодо якого можна зробити висновок про опти-

мальність кількості кластерів та суми помилок.

В наведеному випадку не спостерігається суттєвого перелому на графіку, проте можна помітити різке зниження помилки на кількості кластерів 3 та 4, але оцінка (DB-score) розподілення на 4 кластери виявилась кращою. В результаті розподілення було отримано 4 кластери, причому в датасеті спостерігається рівномірний розподіл екземплярів даних за кластерами (рис. 3.20).

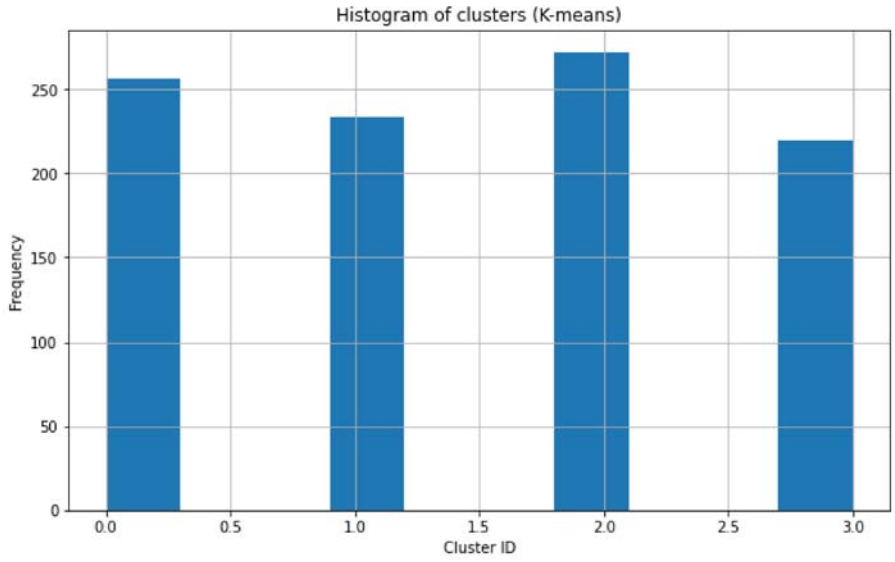


Рис. 3.20. K-means

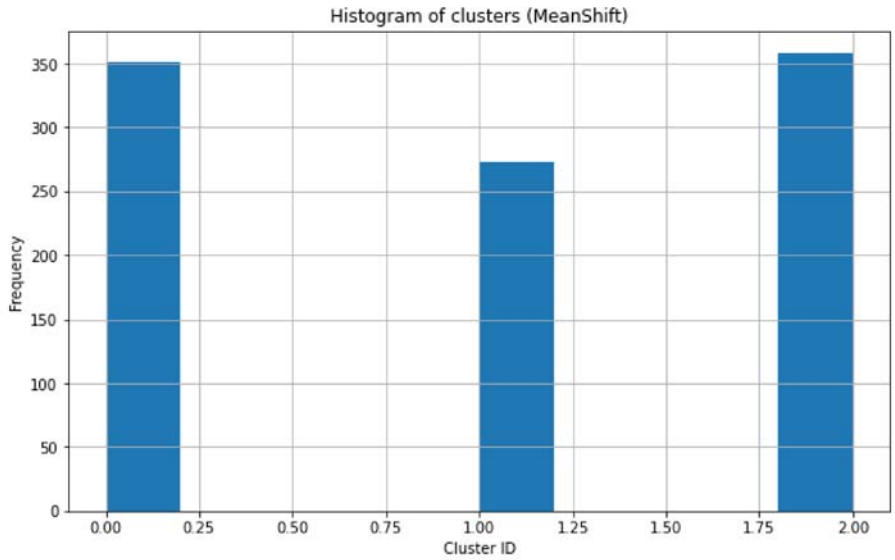


Рис. 3.21. MeanShift

Особливістю використання методу Meanshift є те, що він непараметричний, тобто кількість кластерів, на яку потрібно розділити датасет, не передається до методу. Натомість передається гіпепараметр bandwidth, який

використовується для визначення розміру кластеру. В ході підбору цього гіперпараметру та спираючись на знання предметної галузі алгоритм показав задовільне розподілення датасету на три кластери, причому розподілення відбулось доволі рівномірно (рис. 3.21).

Оскільки використаний для методів датасет є многовимірним і візуалізацію такого простору побудувати неможливо, тому було побудовано візуалізацію двовимірних (рис. 3.22, рис. 3.23, рис. 3.25, рис. 3.25) та трьохвимірних (рис. 3.24, рис. 3.27) просторів з розміченими кластерами, де вісями були найвагоміщі за результатами попереднього аналізу критеріями.

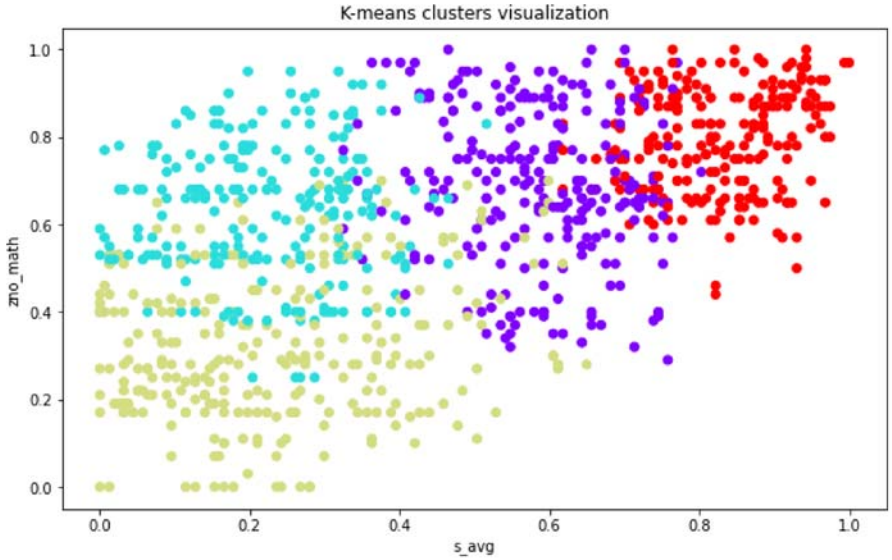


Рис. 3.22. Візуалізація кластерів на площині (s_min, zno_math)

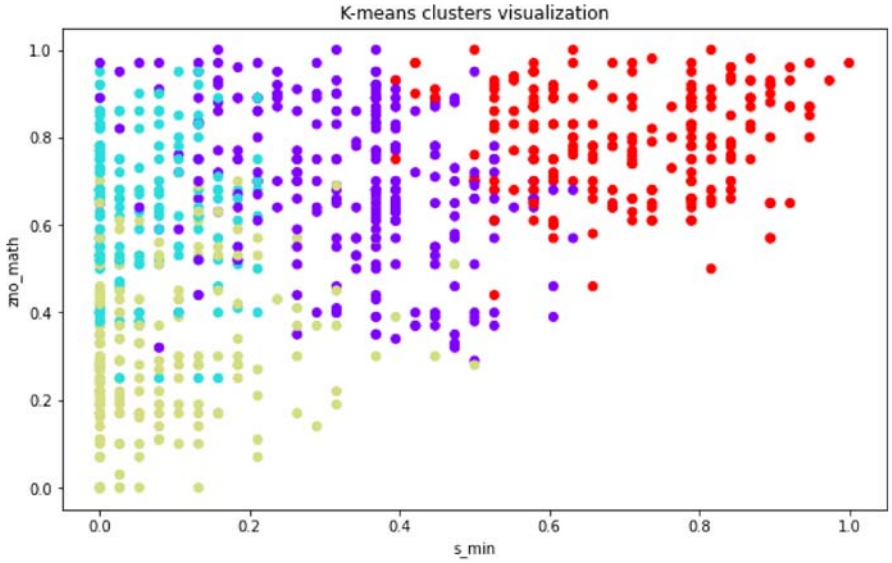


Рис. 3.23. Візуалізація кластерів на площині (s_avg, zno_math)

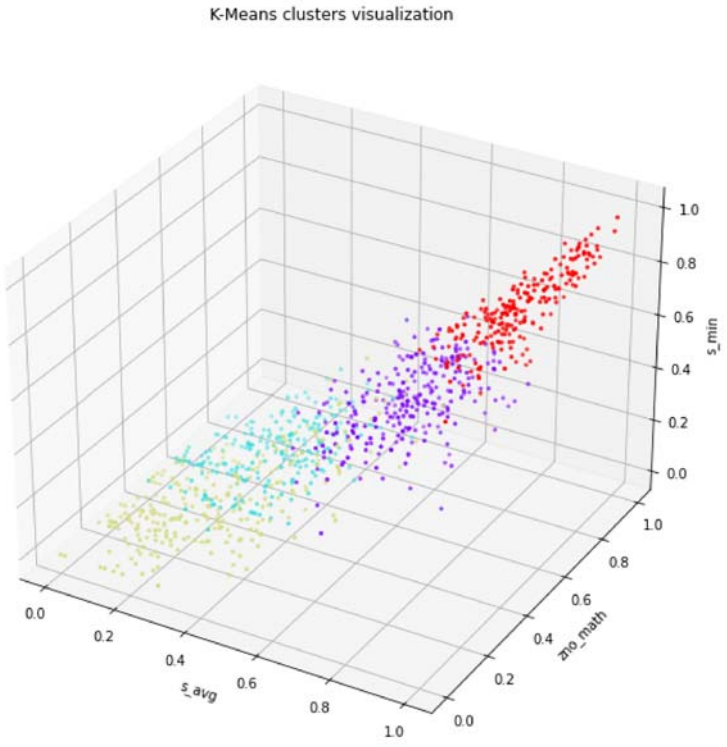


Рис. 3.24. Візуалізація кластерів отриманих за допомогою kmeans

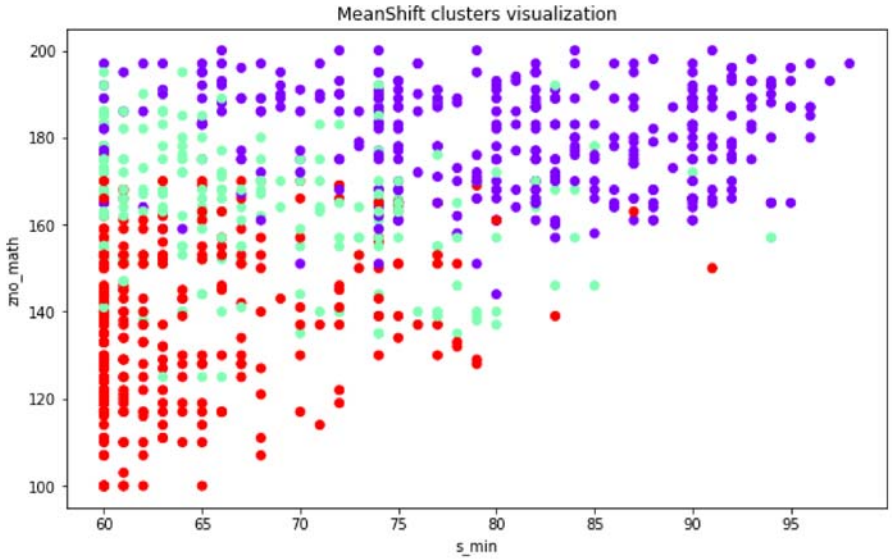


Рис. 3.25. Візуалізація кластерів на площині (s_min, zno_math)

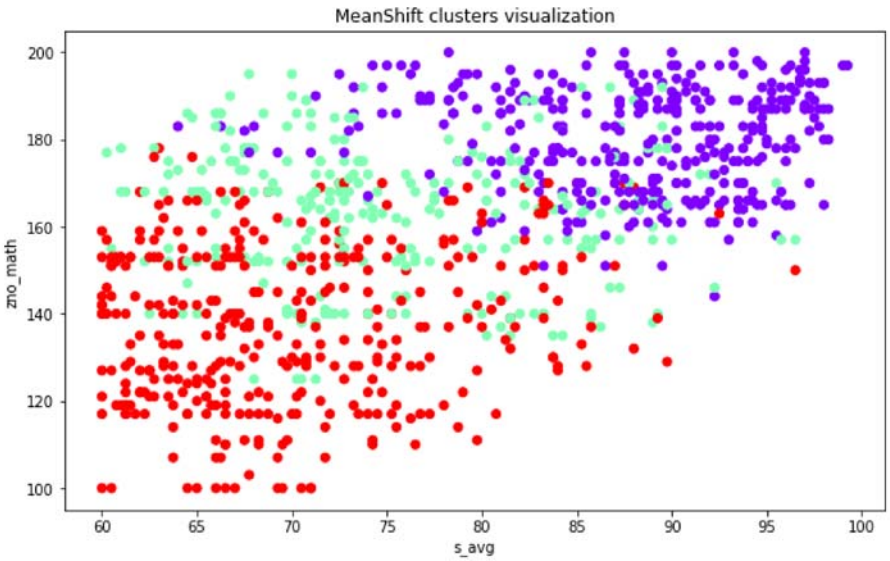


Рис. 3.26. Візуалізація кластерів на площині (s_avg, zno_math)

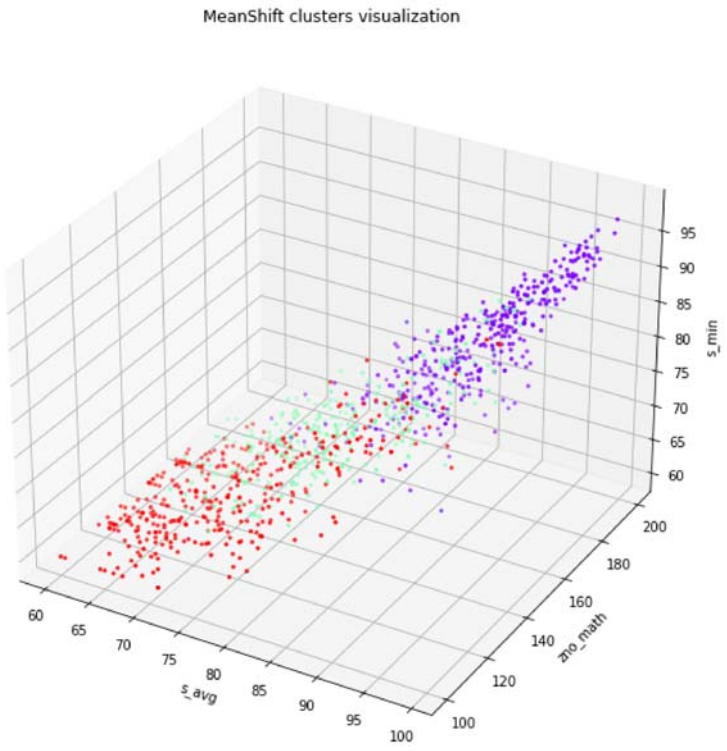


Рис. 3.27. Візуалізація кластерів отриманих за допомогою MeanShift

За вищезазначеними візуалізаціями можна зробити висновки, що метод k-means розподілив датасет на 4 кластери, які відображають студентів які демонструють високі академічні показники, вище середнього, середні та низькі відповідно. Метод meanshift розподілив датасет на три групи з високими, середніми та низькими академічними показниками. Негативними аспектами останього методу є те, що метрика використана для оцінгки показує трохи

гірші результати, ніж k-means на 4 кластерах, а також те, що розподілення кластерів показало, що студентів з середніми показниками меньше ніж тих, хто демонструє високі або низькі результати, а це не відповідає допустимому розподіленню предметної галузі. Також негативним аспектом meanshift є велике пересікання кластерів, що не є позитивною властивістю. Таким чином було зроблено висновок, що датасет буде розмічено за кластерізацією k-means на 4-ох кластерах.

3. Класифікація. Для класифікації було обрано методи DT, Naive Bayes, MLP та SVC.

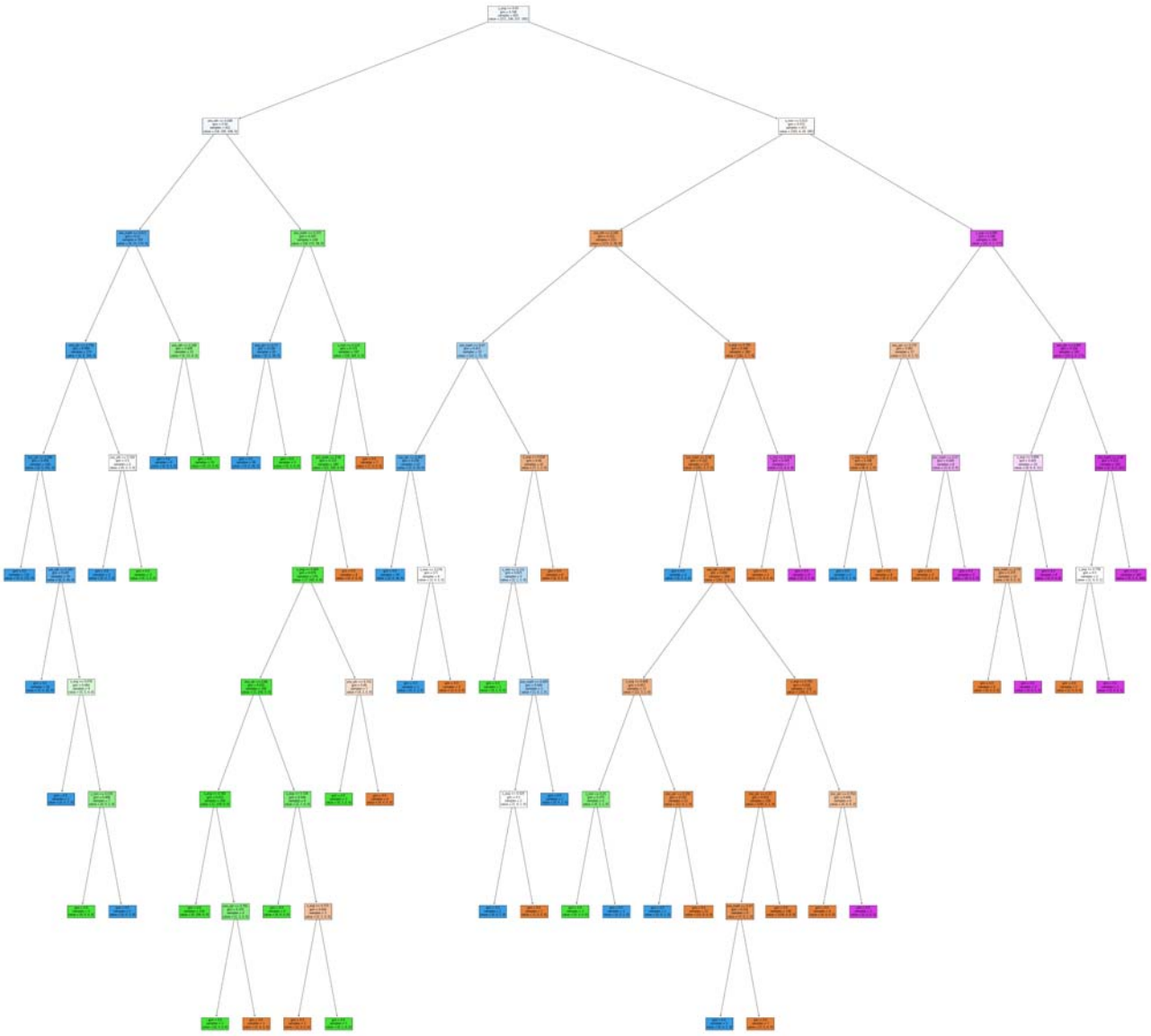


Рис. 3.28. Побудоване дерево рішень

Після проведених раніше етапів класифікація є відносно простішим за-

няттям і полягає в обранні найкращого методу класифікації, який можна було застосувати для розміченого на попередньому етапі датасеті.

Важливим аспектом для тренування та оцінки класифікаторів було розділення датасету в якому приблизно 20% було використано для тестування (рис. 3.29).

```
In [92]: test_size = 150

df0 = shuffle(dfc, random_state=0).reset_index()
df0.drop(columns=['index'],axis=1,inplace=True)
dfc_train = df0.head(982-test_size).reset_index(drop=True)
dfc_test = df0.tail(test_size).reset_index(drop=True)
```

Рис. 3.29. Розділення датасету для класифікації

Для оцінки класифікаторів було обрано метрики Accuracy та F1-score, а також побудовано confusion matrix для візуальної оцінки точності роботи класифікаторів (рис. 3.30, рис. 3.31, рис. 3.32, рис. 3.33).

Побудовані матриці показали задовільний результат класифікації та чітко розподілили тестувальну вибірку на 4 класи.

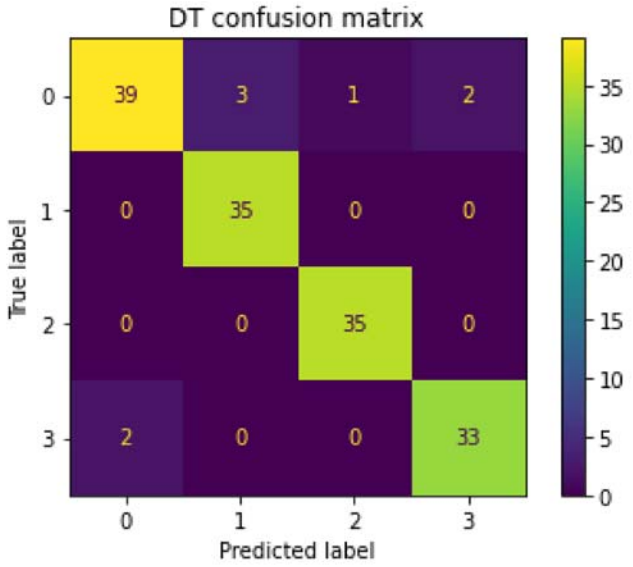


Рис. 3.30. Confusion matrix для побудованого дерева

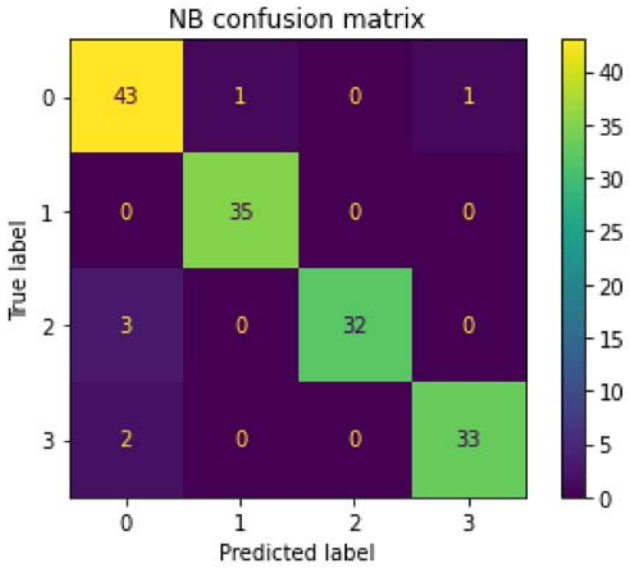


Рис. 3.31. Confusion matrix для побудованого Naive Bayes

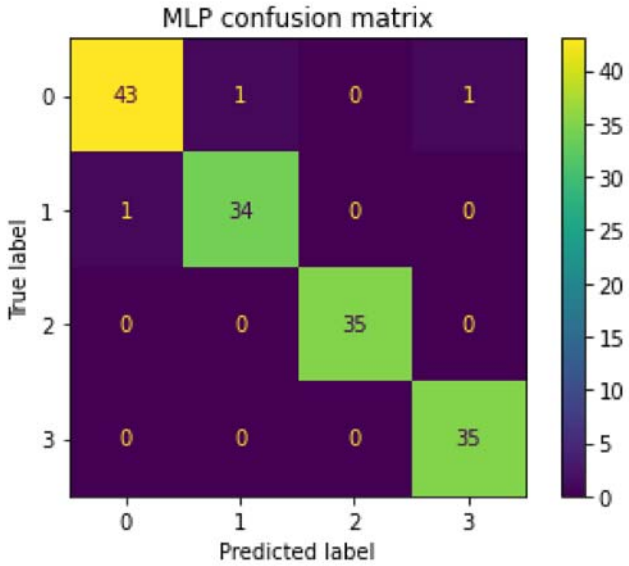


Рис. 3.32. Confusion matrix для побудованого MLP

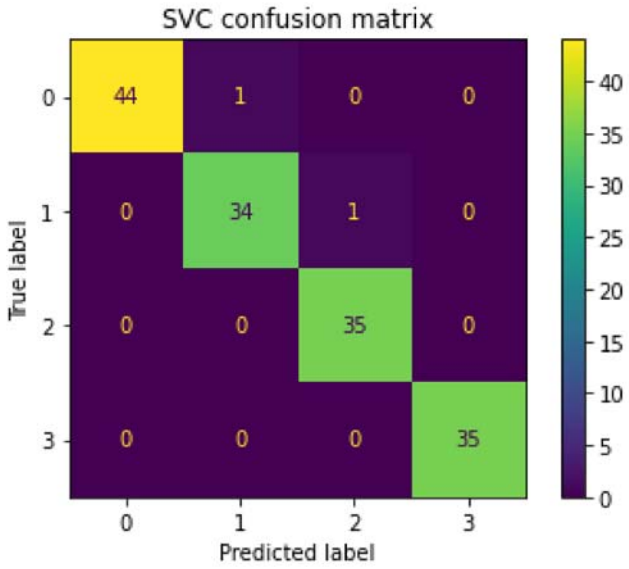


Рис. 3.33. Confusion matrix для побудованого SVC

3.4.5. Evaluation

Для вирішення задач регресії було застосовано методи RF, LR, MLP та SVR, для оцінки використовувалися методи MSE (меньше–краще) та R2-score (більше–краще).

Таблиця 3.2

Порівняння методів регресії

Назва методу	Train MSE	Test MSE	Train R2	Test R2
Random Forest	11.75	33.94	0.86	0.55
Linear Regression	39.96	31.81	0.41	0.54
Multi-Layer Perceptron	27.77	38.39	0.71	0.58
SVM Regression	30.63	32.39	0.67	0.66

Виходячи з наведених в таблиці 3.2 результатів, можна зробити висновок, що на використаному в дослідженні датасеті найкращий результат демонструє SVM Regression. Методи Random Forest та Multi-Layer Perceptron можливо демонструють проблему overfitting хоча показують задовільні результати. Метод Linear Regression для прогнозування успішності студента за даними наведеними в дослідженні використовувати не рекомендується.

Для вирішення задач кластеризації було застосовано методи K-means та MeanShift, для оцінки використовувався метод Davies-Bouldin score(меньше-краще).

Таблиця 3.3

Порівняння методів кластеризації

Назва методу	DB score
K-means	1.32
MeanShift	1.40

Виходячи з наведених в таблиці 3.3 результатів, можна зробити висновок, що обидва методи демонструють схожі показники, але K-means демонструє кращий результат.

Для вирішення задач класифікації було застосовано методи DT, SVC, Naive Bayes та MLP, для оцінки використовувалися методи Accuracy (більше–краще) та F1-score (більше-краще).

Таблиця 3.4

Порівняння методів класифікації

Назва методу	Train Acc	Test Acc	Train F1	Test F1
Decision Tree	1.0	0.95	1.0	0.95
SVM Classifier	0.99	0.99	0.99	0.99
Naive Bayes	0.94	0.95	0.94	0.95
Multi-Layer Perceptron	0.98	0.98	0.98	0.98

Виходячи з наведених в таблиці 3.4 результатів, можна зробити висновок, що на використаному в дослідженні датасеті найкращий результат демонструє SVM Classifier та Multi-Layer Perceptron. Методи Naive Bayes та Decision Tree теж демонструють добрі результати.

3.4.6. Deployment

Основною метою розгортування було використати попередньо натреновані моделі, які в даному випадку було збережено за допомогою модуля pickle (рис. 3.34) [39].

```
with open('mlpregressor.pkl', 'rb') as f:
    mlpregregressor = pickle.load(f)

with open('svrr.pkl', 'rb') as f:
    svrr = pickle.load(f)

with open('rfregressor.pkl', 'rb') as f:
    rfregressor = pickle.load(f)

with open('dtclassifier.pkl', 'rb') as f:
    dt = pickle.load(f)

with open('naivebayes.pkl', 'rb') as f:
    nb = pickle.load(f)

with open('svc.pkl', 'rb') as f:
    svc = pickle.load(f)
```

Рис. 3.34. Десеріалізація попередньо збережених моделей

Для перевірки правильності процесу серіалізації-десеріалізації, було виведено вагові коефіцієнти характеристик однієї з моделей (рис. 3.35). Оскільки вагові коефіцієнти після десеріалізації співпадають з тими, які були отримані під час навчання, можна зробити висновок, що десеріалізація моделей пройшла успішно.

```
rfregressor.feature_importances_
```

```
array([0.04606645, 0.05859859, 0.66104172, 0.05995599, 0.06075691,
       0.05735337, 0.05622696])
```

Рис. 3.35. Перевірка десеріалізованої моделі

Оскільки при тренуванні моделі навчалися на нормалізованому датасеті, таку ж операцію необхідно виконати для кожного нового запиту, для цього можна використовувати прості математичні формули спираючись на максимальні і мінімальні дані з початкового датасету (рис. 3.36).

```
vals = np.array([[1, 1, 70., 90., 80., 120., 170., 130., 2.]])
cols = np.array(['gender', 'funding', 's_min', 's_max', 's_avg', 'zno_math', 'zno_ukr', 'zno_oth', 'semester'])

df = pd.DataFrame(vals, columns=cols)
df['s_min'] = (df['s_min'] - 60.) / 40.
df['s_avg'] = (df['s_avg'] - 60.) / 40.
df['s_max'] = (df['s_max'] - 60.) / 40.

df['zno_math'] = (df['zno_math'] - 100.) / 100.
df['zno_ukr'] = (df['zno_ukr'] - 100.) / 100.
df['zno_oth'] = (df['zno_oth'] - 100.) / 100.

df['semester'] = df['semester'] / 8.

df
```

	gender	funding	s_min	s_max	s_avg	zno_math	zno_ukr	zno_oth	semester
0	1.0	1.0	0.25	0.75	0.5	0.2	0.7	0.3	0.25

Рис. 3.36. Нормалізація даних

Після десеріалізації можна легко передбачити середній семестровий бал

студента за допомогою моделей регресії (рис. 3.37).

```
p = pd.DataFrame(df[rfregressor.feature_names_in_])
rfregressor.predict(p)
array([75.096875])

p = pd.DataFrame(df[mlpregregressor.feature_names_in_])
mlpregregressor.predict(p)
array([86.24687141])

p = pd.DataFrame(df[svrr.feature_names_in_])
svrr.predict(p)
array([83.41263023])
```

Рис. 3.37. Передбачення успішності студента в наступному семестрі

В даному випадку спостерігається розбіжність між результатами передбачення семестрового балу, зумовлена невисокою точністю навчання моделей. Слід зауважити, що SVM Regression продемонстрував найнижчу помилку в ході перевірки на тестовому датасеті, тому цьому методу потрібно більше довіряти. Тим не меньш, дана проблема була зазначена при оцінці ризиків, тому предметом для наступного дослідження повинен стати набір даних з більшою кількістю характерстик, для чого необхідно додати в освітню ІС відстежування середньосеместрової успішності та відсутності студентів, як це було зроблено в дослідженнях з аналізу джерел.

Десеріалзовані моделі для класифікації можуть бути використані для визначення до якої категорії студентів відноситься даний результат, що може бути використано для визначення студентів, які можуть потрапити до групи ризику за їх академічними показниками (рис. 3.38).

```
p = pd.DataFrame(df[dt.feature_names_in_])
dt.predict(p)
array([2])

p = pd.DataFrame(df[nb.feature_names_in_])
nb.predict(p)
array([2])

p = pd.DataFrame(df[svc.feature_names_in_])
svc.predict(p)
array([2])
```

Рис. 3.38. Визначення належності студента до класу

В даному випадку висновки отримані з різних класифікаторів збігаються, що зумовлено високою точністю досягнутою при тренуванні моделей,

але оскільки при проведенні кластеризації не було отримано чіткого розділення датасету на кластери, все ж пропонується відстежувати додаткові академічні і демографічні характеристики в освітній ІС, щоб отримати більш цікаві результати.

Таким чином, в даному розділі було розглянуто методику, що використовувався при проведенні дослідження, зокрема обгрунтовано вибір програмних засобів, використаних для створення середовища для дослідження, зібрано датасет та визначено перспективні характеристики для дослідження. Спираючись на попередньо проведений аналіз джерел було обрано актуальну методологію для проведення досліджень в даній галузі, а також оцінено ризики, які можуть негативно впливати на очікувані результати дослідження.

Проведене за методологією CRISP-DM дослідження дозволило створити моделі з задовільною точністю та провести регресійний аналіз, розділення датасету на кластери та створити класифікатори, що визначають приналежність студента до однієї з чотирьох категорій. Для покращення майбутніх досліджень запропоновано відстежувати додаткові характеристики в освітній ІС університету.

ВИСНОВКИ

В рамках дипломного бакалаврського проєкту було отримано дослідницьке завдання на розроблення модуля інформаційної системи для аналізу успішності студентів на основі методів Data mining.

Відповідно до завдання було визначено проблему, розглянуто її актуальність шляхом пошуку в базах наукових статей та журналів, підтверджено актуальність завдання та розглянуто перспективи для подальшого дослідження і розробки програмного модуля, який підсумує результати проведеної роботи. По відповідній темі було знайдено та проаналізовано статті з ScienceDirect, IEEE Xplore та ін. Аналіз показав, що за темою проводяться активні дослідження протягом останніх 15 років, як закордонними, так і вітчизняними вченими. Зміст статей по темі показав наявність широковикористовуваних методів для вирішення проблеми передбачення успішності студентів, але усталених рішень поставленої проблеми не визначено.

Було визначено підзавдання та методи і програмні інструменти, які використовуввались при подальшому дослідженні. Дослідження було виконано за однією з найпоширеніших методологій для проведення дослідженнь CRISP-DM. Імплементація та оцінювання методів продемонструвало ефективність K-means для кластеризації, SVR для регресії та SVC для класифікації.

Таким чином, виконання дипломного проєкту за вищезазначеною темою дійсно має науковий та практичний потенціал, практичну значимість та актуальність для суспільства, оскільки на дослідження з даної теми є попит в науковому та професійному середовищі. Позитивним результатом роботи є визначення найбільшого впливу середнього семестрового балу за семестр на подальшу успішність студента, незначного впливу балів зовнішнього незалежного оцінювання на успішність студента, а також створення програмного модуля з передбачення успішності студентів.

В подальшому рекомендується враховувати додаткові показники студента в університетських ІС, такі як середньосеместрова успішність та відвідуваність занять.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сугоняк І. І. OLAP-технології для моніторингу успішності студентів за умов рейтингової системи оцінювання знань [Текст] / І. І. Сугоняк. // Інформаційні технології і засоби навчання, 2013, Том 38, №6. — Київ: [б. в.], 2013. – С. 245–256.

2. Тимофєєв В. І. Використання технології data mining для оцінки якості навчання у вищому навчальному закладі [Текст] / В. І. Тимофєєв, О. В. Лисенко // Вісник НУК № 1, 2012. — Миколаїв : НУК імені адмірала Макарова, 2012. – С. 191–193.

3. Analytics & ML Will Transform Education (Cloud Next '18) [Електронний ресурс] / Google Cloud // YouTube. — Електрон. відеодан. – Режим доступу: https://www.youtube.com/watch?v=JewJZR_YtZE. – Назва з екрану. – Дата перегляду: 01.04.2022.

4. Using Data to Analyze Learning [Електронний ресурс] / Education at Illinois // YouTube. — Електрон. відеодан. – Режим доступу: https://www.youtube.com/watch?v=KZkhCxAPBVQ. – Назва з екрану. – Дата перегляду: 01.04.2022.

5. Cassano, R.; Costa, V.; Fornasari, T. An Effective National Evaluation System of Schools for Sustainable Development: A Comparative European Analysis. Sustainability 2019, 11, 195. http://doi.org/10.3390/su11010195

6. Digging Deep into Educational Data [Електронний ресурс] / WPI // YouTube. — Електрон. відеодан. – Режим доступу: https://www.youtube.com/watch?v=33NHytSfPoY. – Назва з екрану. – Дата перегляду: 01.04.2022.

7. David J. Lemay, Clare Baek, Tenzin Doleck, Comparison of learning analytics and educational data mining: A topic modeling approach, Computers and Education: Artificial Intelligence, Volume 2, 2021, 100016, ISSN 2666-920X, https://doi.org/10.1016/j.caeai.2021.100016.

8. educationaldatamining.org [Електронний ресурс] // educationaldatamining.org. – Режим доступу : https://educationaldatamining.org/. — Назва з екрану. – Дата перегляду : 14.04.2022.

9. F. J. Kaunang and R. Rotikan, "Students' Academic Performance Prediction using Data Mining,"2018 Third International Conference on Informatics and Computing (ICIC), 2018, pp. 1-5, doi: 10.1109/IAC.2018.8780547.

10. Thaher, T.; Zaguia, A.; Al Azwari, S.; Mafarja, M.; Chantar, H.; Abuham-

dah, A.; Turabieh, H.; Mirjalili, S.; Sheta, A. An Enhanced Evolutionary Student Performance Prediction Model Using Whale Optimization Algorithm Boosted with Sine-Cosine Mechanism. Appl. Sci. 2021, 11, 10237.
https://doi.org/ 10.3390/app112110237

11. Educational Data Mining (EDM) 2020 Presentation [Електронний ресурс] / Hamid Karimi // YouTube. — Електрон. відеодан. – Режим доступу: https://www.youtube.com/watch?v=xrwxIALtHrc. – Назва з екрану. – Дата перегляду: 01.04.2022.

12. Mudasir Ashrafa, Majid Zamanbuhee Ahmedc. An Intelligent Prediction System for Educational Data Mining Based on Ensemble and Filtering approaches Author links open overlay panel

13. What is Learning Analytics [Електронний ресурс] // Society for Learning Analytics Research. – Режим доступу:
https://www.solaresearch.org/about/what-is-learning-analytics/. — Назва з екрану. – Дата перегляду : 14.04.2022.

14. Romero, Cristóbal & Ventura, Sebastian. (2007). Educational data mining: A survey from 1995 to 2005. Expert Systems with Applications. 33. 135-146. 10.1016/j.eswa.2006.04.005.

15. AlQaheri, H.; Panda, M. An Education Process Mining Framework: Unveiling Meaningful Information for Understanding Students' Learning Behavior and Improving Teaching Quality. Information 2022, 13, 29.
https://doi.org/10.3390/info13010029

16. A systematic literature review on student performance predictions Hasnah Nawang, Mokhairi Makhtar and Wan Mohd Amir Fazamin Wan Hamzah. International Journal of Advanced Technology and Engineering Exploration, Vol 8(84) ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2021.874521

17. Вергун В. Р. Прогнозування успішності студентів на основі методів інтелектуального аналізу даних навчальних програм [Текст] / В. Р. Вергун // НАУКОВІ ЗАПИСКИ * 2019 / 1 (58) — Київ : НаУКМА, 2019. – С. 71–78.

18. Zafari, M.; Sadeghi-Niaraki, A.; Choi, S.-M.; Esmaeily, A. A Practical Model for the Evaluation of High School Student Performance Based on Machine Learning. Appl. Sci. 2021, 11, 11534. https://doi.org/10.3390/app112311534

19. Kenan Zengin, Necmi Esgi, Ergin Erginer, Mehmet Emin Aksoy, A sample study on applying data mining research techniques in educational science: Developing a more meaning of data, Procedia - Social and Behavioral Sciences, Volume

15, 2011, Pages 4028-4032, ISSN 1877-0428,

20. K. Govindasamy and T. Velmurugan. A Study on Classification and Clustering Data Mining Algorithms based on Students Academic Performance Prediction. International Journal of Control Theory and Applications ISSN : 0974–5572 Vol 10, N 23, 2017

21. Santosa, Lukito, & Chrismanto. Classification and Prediction of Students' GPA Using KMeans Clustering Algorithm to Assist Student Admission Process. Journal of Information Systems Engineering and Business Intelligence, 2021, 7 (1), 1-10

22. Osmanbegovic, Edin; Suljic, Mirza (2012) : Data Mining Approach for Predicting Student Performance, Economic Review: Journal of Economics and Business, ISSN 1512-8962, University of Tuzla, Faculty of Economics, Tuzla, Vol. 10, Iss. 1, pp. 3-12

23. Alboaneen, D.; Almelihi, M.; Alsubaie, R.; Alghamdi, R.; Alshehri, L.; Alharthi, R. Development of a Web-Based Prediction System for Students' Academic Performance. Data 2022, 7, 21. https://doi.org/10.3390/data7020021

24. Zhang Y, Yun Y, An R, Cui J, Dai H and Shang X (2021) Educational Data Mining Techniques for Student Performance Prediction: Method Review and Comparison Analysis. Front. Psychol. 12:698490. doi: 10.3389/fpsyg.2021.698490.

25. Supervised vs. Unsupervised Learning: What's the Difference? [Електронний ресурс] // IBM. – Режим доступу:
https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning. — Назва з екрану. – Дата перегляду : 18.04.2022.

26. Regression in data mining [Електронний ресурс] // Javatpoint. – Режим доступу: https://www.javatpoint.com/regression-in-data-mining . — Назва з екрану. – Дата перегляду : 18.04.2022.

27. IBM SPSS Modeler CRISP-DM Guide [Електронний ресурс]. – Режим доступу: https://www.ibm.com/docs/en/spss-modeler/18.1.1

28. Data Science Methodology 101 [Електронний ресурс]. – Режим доступу: https://towardsdatascience.com/data-science-methodology-101-ce9f0d660336

29. Пошук абітурієнтів [Електронний ресурс]. – Режим доступу:
https://abit-poisk.org.ua/ — Назва з екрану. — Дата перегляду: 10.04.2022.

30. Enterprise Open Source and Linux [Електронний ресурс] // Ubuntu. – Режим доступу: https://ubuntu.com/. — Назва з екрану. – Дата перегляду : 14.04.2022.

31. Home [Електронний ресурс] // Docker. – Режим доступу:

https://www.docker.com/. — Назва з екрану. – Дата перегляду : 14.04.2022.

32. Исследование: Женщины в украинском IT — кто они, кем работают и сколько зарабатывают [Електронний ресурс] // ITC.ua . – Режим доступу: https://itc.ua/news/zhenshhiny-v-ukrainskom-it-kto-oni-kem-rabotayut-i-skolko-zarabatyvayut/. — Назва з екрану. – Дата перегляду : 14.04.2022.

33. Data Wrangling: What It Is & Why It's Important [Електронний ресурс] // Harvard Business School Online . – Режим доступу: https://online.hbs.edu/blog/post/data-wrangling. — Назва з екрану. – Дата перегляду : 18.04.2022.

34. Project Jupyter | Home [Електронний ресурс] // Project Jupyter. – Режим доступу: https://jupyter.org/. — Назва з екрану. – Дата перегляду : 18.04.2022.

35. What is Linear Regression [Електронний ресурс] // Towards Data Science. – Режим доступу: https://towardsdatascience.com/the-concepts-behind-linear-regression-and-its-implementation-ffbab5a4d65e . — Назва з екрану. – Дата перегляду : 10.04.2022.

36. Random Forest Regression [Електронний ресурс] // Chaya Bakshi. – Режим доступу: https://levelup.gitconnected.com/random-forest-regression-209c0f354c84. — Назва з екрану. – Дата перегляду : 10.04.2022.

37. All you need to know about Polynomial Regression [Електронний ресурс] // Analytics Vidhya. – Режим доступу: https://www.analyticsvidhya.com/blog/2021/07/all-you-need-to-know-about-polynomial-regression/. — Назва з екрану. – Дата перегляду : 10.04.2022.

38. Karalar, H., Kapucu, C. & Guruler, H. Predicting students at risk of academic failure using ensemble model during pandemic in a distance learning system. Int J Educ Technol High Educ 18, 63 (2021). https://doi.org/10.1186/s41239-021-00300-y

39. pickle – Python object serialization – Python 3.10.4 documentation [Електронний ресурс] // Python 3.10.4 documentation. – Режим доступу: https://docs.python.org/3/library/pickle.html. — Назва з екрану. – Дата перегляду : 18.04.2022.

ДОДАТОК А

In [ ]:
```
pip install pydotplus dtreeviz
```

Importing modules

In [3]:
```
import matplotlib.pyplot as plt
import matplotlib.image as pltimg
from mpl_toolkits.mplot3d import Axes3D
from dtreeviz.trees import dtreeviz
import pydotplus
import numpy as np
import pandas as pd
import hashlib
from sklearn import tree
from sklearn import linear_model
from sklearn import preprocessing
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import f1_score, r2_score, mean_squared_error, \
davies_bouldin_score, confusion_matrix, multilabel_confusion_matrix, \
balanced_accuracy_score, accuracy_score
from sklearn.neural_network import MLPRegressor, MLPClassifier
from sklearn.model_selection import train_test_split, cross_val_score, \
RepeatedKFold, KFold, GridSearchCV, cross_validate
from sklearn.utils import shuffle
from sklearn.svm import SVR, SVC
from scipy import stats
from sklearn.cluster import KMeans, DBSCAN, MeanShift
import pickle
```

Preparing the dataset

In [13]:
```
folder = "data"

paths = [
    "121-2018",
    "121-2019",
    "121-2020",
    "121-2021",
    "122-2018",
    "122-2019",
    "122-2020",
    "122-2021",
]

semesters = [
    [1, 2, 3, 4, 5, 6, 7],
    [1, 2, 3, 4, 5],
    [1, 2, 3],
    [1],
    [1, 2, 3, 4, 5, 6, 7],
    [1, 2, 3, 4, 5],
    [1, 2, 3],
```

```
    [1],
]
```

In [14]:
```
dataframes = list()
```

In [15]:
```
for i in range(len(paths)):
    for semester in (semesters[i][:-1]):
        df0 = pd.read_csv(folder + "/" + paths[i] + ".csv")
        df_s1 = pd.read_csv(folder + "/" + paths[i] + "-" + str(semester) + ".c
        df_s2 = pd.read_csv(folder + "/" + paths[i] + "-" + str(semester + 1) +

        df0["semester"] = semester
        df0["s_min"] = df_s1[["s1", "s2", "s3", "s4"]].min(axis=1).astype("floa
        df0["s_max"] = df_s1[["s1", "s2", "s3", "s4"]].max(axis=1).astype("floa
        df0["s_avg"] = df_s1[["s1", "s2", "s3", "s4"]].mean(axis=1).astype("flo
        df0["hours"] = df_s1[["c1", "c2", "c3", "c4"]].sum(axis=1).astype("floa
        df0["pred"] = df_s2[["s1", "s2", "s3", "s4"]].mean(axis=1).astype("floa

        dataframes.append(df0)
```

In [16]:
```
df_overall = pd.concat(dataframes,
                       axis=0,
                       ignore_index=True)
df_overall.rename(columns={"z1": "zno_math",
                           "z2": "zno_ukr",
                           "z3":"zno_oth"}, inplace=True)
df_overall["zno_math"] = df_overall["zno_math"].astype("float64")
df_overall["zno_ukr"] = df_overall["zno_ukr"].astype("float64")
df_overall["zno_oth"] = df_overall["zno_oth"].astype("float64")
df_overall["gender"] = df_overall["gender"].astype("category")
df_overall["funding"] = df_overall["funding"].astype("category")
df_overall["id"] = df_overall["name"].apply(
    lambda x: hashlib.md5(x.encode()).hexdigest()
)

df_res = df_overall.reindex(columns=["id", "name", "gender",
                                     "funding", "zno_math",
                                     "zno_ukr", "zno_oth",
                                     "semester", "s_min",
                                     "s_max", "s_avg",
                                     "hours", "pred"])

df_res.dtypes
```

Out[16]:
```
id            object
name          object
gender      category
funding     category
zno_math     float64
zno_ukr      float64
zno_oth      float64
semester       int64
s_min        float64
s_max        float64
s_avg        float64
hours        float64
pred         float64
dtype: object
```

In [17]:
```
df_res.iloc[4].to_frame()
```

Out[17]:

	4
id	ee5c043c562a9d9d7ece1764eef1ebeb
name	Жовтобрюх Дмитро Андрійович
gender	1
funding	1
zno_math	193.0
zno_ukr	189.0
zno_oth	186.0
semester	1
s_min	93.0
s_max	100.0
s_avg	97.25
hours	690.0
pred	95.25

Anonymize the dataset

In [18]:
```
df_res.drop(columns=["name"], inplace=True)
df_res.head(10)
```

Out[18]:

	id	gender	funding	zno_math	zno_ukr	zno_oth	semester	s_
0	af609c9a84cece367d5d54b287b8906b	1	1	193.0	182.0	178.0	1	8
1	2c7b91043b6ff937856aed4b2f7b3f07	1	0	166.0	161.0	146.0	1	6
2	151866af35d63facb5ad2150333b236c	1	0	138.0	135.0	171.0	1	6
3	478f3dce742612f00987242a808f38bb	1	0	186.0	181.0	185.0	1	8
4	ee5c043c562a9d9d7ece1764eef1ebeb	1	1	193.0	189.0	186.0	1	9
5	49c1e922887c0fed05e92d096220c58a	1	0	164.0	120.0	159.0	1	6
6	5fcdd7e7b7bb92d4c034a376dd51f7bc	1	0	168.0	131.0	158.0	1	6
7	6e10f3f0263598616cc674b332ec69be	1	0	138.0	155.0	162.0	1	N
8	4bca36b9c7a2e634d4292012153af276	1	0	150.0	173.0	136.0	1	6
9	1c9066e54b09ebb5a5cae1331e965ba0	1	1	200.0	181.0	193.0	1	9

Data Wrangling

In [25]:
```
df_res.isnull().any(axis=1).sum()
```

Out[25]:
27

In [8]:
```
df_res.dropna(axis=0, inplace=True)
df_res.head(10)
```

Out[8]:

	id	gender	funding	zno_math	zno_ukr	zno_oth	semester	s
0	af609c9a84cece367d5d54b287b8906b	1	1	193.0	182.0	178.0	1	
1	2c7b91043b6ff937856aed4b2f7b3f07	1	0	166.0	161.0	146.0	1	
2	151866af35d63facb5ad2150333b236c	1	0	138.0	135.0	171.0	1	
3	478f3dce742612f00987242a808f38bb	1	0	186.0	181.0	185.0	1	
4	ee5c043c562a9d9d7ece1764eef1ebeb	1	1	193.0	189.0	186.0	1	
5	49c1e922887c0fed05e92d096220c58a	1	0	164.0	120.0	159.0	1	
6	5fcdd7e7b7bb92d4c034a376dd51f7bc	1	0	168.0	131.0	158.0	1	
8	4bca36b9c7a2e634d4292012153af276	1	0	150.0	173.0	136.0	1	
9	1c9066e54b09ebb5a5cae1331e965ba0	1	1	200.0	181.0	193.0	1	
10	8e5f5e9e99449828159abfea1ca7396d	1	1	195.0	165.0	185.0	1	

Saving the dataset

In [9]:
```
df_res.to_csv("dataset.csv", index=False)
```

Load the dataset

In [4]:
```
df = pd.read_csv("dataset.csv")
#df["zno"] = df["zno_math"]*0.5+df["zno_ukr"]*0.3+df["zno_oth"]*0.2
df.drop(columns=["id"], inplace=True)
df = df[(df.s_min > 59) & (df.pred > 59)]
df.reset_index(drop=True, inplace=True)
df
```

Out[4]:

	gender	funding	zno_math	zno_ukr	zno_oth	semester	s_min	s_max	s_avg	hours	pred
0	1	1	193.0	182.0	178.0	1	81.0	100.0	89.50	690.0	88.00
1	1	0	166.0	161.0	146.0	1	60.0	82.0	74.00	690.0	72.00
2	1	0	138.0	135.0	171.0	1	61.0	77.0	67.00	690.0	68.75
3	1	0	186.0	181.0	185.0	1	81.0	96.0	90.25	690.0	87.25
4	1	1	193.0	189.0	186.0	1	93.0	100.0	97.25	690.0	95.25
...	...	...	...	...	...	...	...	...	...	...	...
977	1	0	100.0	104.0	109.0	2	61.0	76.0	66.50	750.0	63.25
978	1	0	147.0	166.0	181.0	2	61.0	87.0	68.50	750.0	67.50
979	1	0	163.0	157.0	177.0	2	75.0	93.0	80.75	750.0	84.25
980	1	0	143.0	148.0	100.0	2	62.0	77.0	66.00	750.0	68.00
981	0	1	153.0	110.0	139.0	2	61.0	90.0	75.00	750.0	79.75

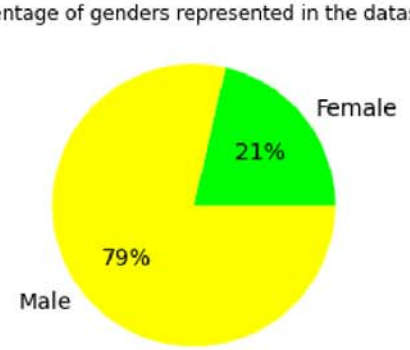
982 rows × 11 columns

Exploring the data

In [9]:
```
df.groupby('gender').size().plot(
    kind='pie',
```

```
    labels=['Female', 'Male'],
    textprops={'fontsize': 14},
    colors=['lime', 'yellow'],
    autopct=(lambda val: f'{val:.0f}%'),
    ylabel="",
    title="Percentage of genders represented in the dataset")
```

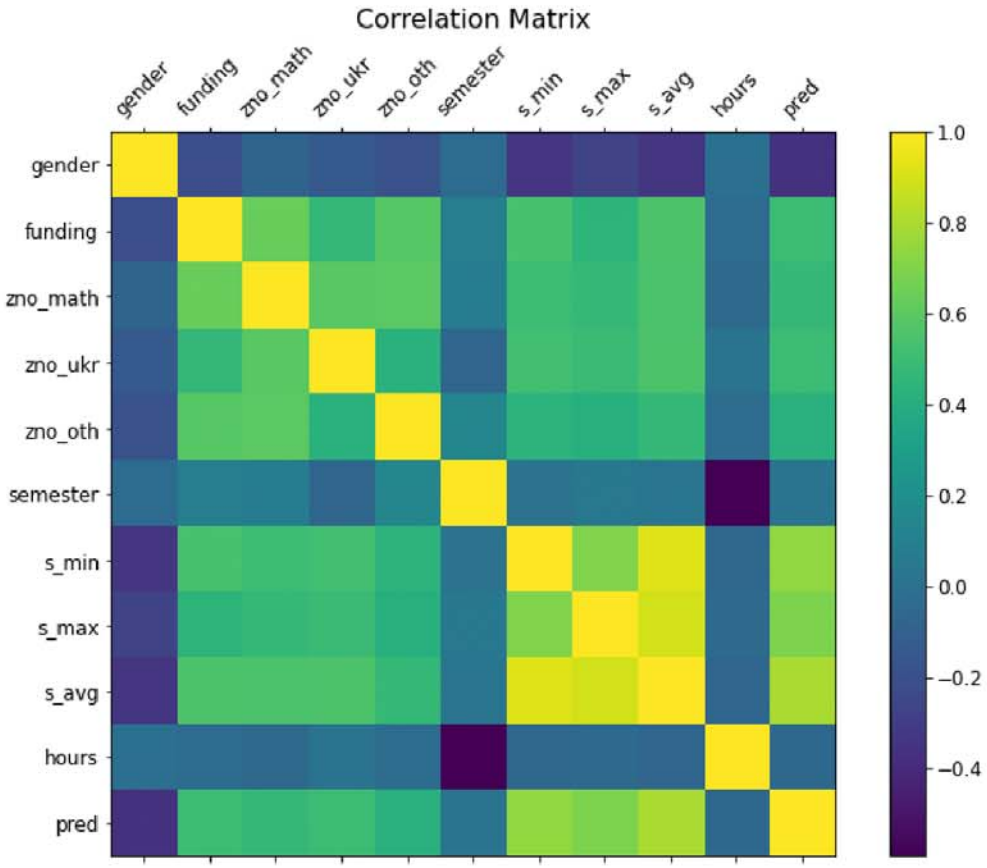
Out[9]:
```
<AxesSubplot:title={'center':'Percentage of genders represented in the datase
t'}>
```



Plot Correlation Matrix Heatmap

In [5]:
```
f = plt.figure(figsize=(12, 8))
plt.matshow(df.corr(), fignum=f.number)
plt.xticks(range(df.select_dtypes(['number']).shape[1]), df.select_dtypes(['num
plt.yticks(range(df.select_dtypes(['number']).shape[1]), df.select_dtypes(['num
cb = plt.colorbar()
cb.ax.tick_params(labelsize=12)
plt.title('Correlation Matrix', fontsize=16);
print(df.corr())
```

```
            gender   funding  zno_math   zno_ukr   zno_oth  semester  \
gender    1.000000 -0.214000 -0.081502 -0.142140 -0.190373 -0.031759
funding  -0.214000  1.000000  0.634833  0.467201  0.587135  0.089151
zno_math -0.081502  0.634833  1.000000  0.594600  0.596436  0.081721
zno_ukr  -0.142140  0.467201  0.594600  1.000000  0.423973 -0.069650
zno_oth  -0.190373  0.587135  0.596436  0.423973  1.000000  0.137346
semester -0.031759  0.089151  0.081721 -0.069650  0.137346  1.000000
s_min    -0.338832  0.535324  0.500134  0.514384  0.437781  0.011103
s_max    -0.268514  0.441033  0.475286  0.493800  0.411547  0.045903
s_avg    -0.335760  0.554260  0.545321  0.555041  0.472400  0.022604
hours    -0.007488 -0.028835 -0.041092  0.019425 -0.027160 -0.595269
pred     -0.362637  0.500259  0.474667  0.498771  0.419348  0.019317

              s_min     s_max     s_avg     hours      pred
gender   -0.338832 -0.268514 -0.335760 -0.007488 -0.362637
funding   0.535324  0.441033  0.554260 -0.028835  0.500259
zno_math  0.500134  0.475286  0.545321 -0.041092  0.474667
zno_ukr   0.514384  0.493800  0.555041  0.019425  0.498771
zno_oth   0.437781  0.411547  0.472400 -0.027160  0.419348
semester  0.011103  0.045903  0.022604 -0.595269  0.019317
s_min     1.000000  0.697833  0.921488 -0.055954  0.743863
s_max     0.697833  1.000000  0.888711 -0.035292  0.692439
s_avg     0.921488  0.888711  1.000000 -0.060343  0.795778
hours    -0.055954 -0.035292 -0.060343  1.000000 -0.051636
pred      0.743863  0.692439  0.795778 -0.051636  1.000000
```



Split the dataset

In [7]:
```
dfx = df[['gender', 'funding', 's_min', 's_max', 's_avg', 'zno_math', 'zno_ukr'
dfy = df['pred']
x = dfx.values
cols = dfx.columns
min_max_scaler = preprocessing.MinMaxScaler()
x_scaled = min_max_scaler.fit_transform(x)
df_scaled = pd.DataFrame(x_scaled, columns=cols)
df_scaled['pred'] = dfy

test_size = 150

df0 = shuffle(df_scaled, random_state=0).reset_index()
df0.drop(columns=['index'],axis=1,inplace=True)
df_train = df0.head(982-test_size).reset_index(drop=True)
df_test = df0.tail(test_size).reset_index(drop=True)
```

In [8]:
```
df_train.head()
```

Out[8]:

	gender	funding	s_min	s_max	s_avg	zno_math	zno_ukr	zno_oth	semester	pred
0	1.0	1.0	0.000000	0.600	0.273885	0.89	0.345550	0.773196	0.4	82.25
1	1.0	0.0	0.131579	0.675	0.394904	0.30	0.219895	0.041237	0.2	83.75
2	0.0	1.0	0.657895	0.750	0.700637	0.75	0.806283	0.711340	0.0	86.00
3	1.0	1.0	0.289474	0.925	0.547771	0.75	0.680628	0.618557	0.0	67.00
4	1.0	0.0	0.078947	0.800	0.324841	0.70	0.544503	0.061856	0.4	65.25

In [9]:
```
df_test.head()
```

Out[9]:

	gender	funding	s_min	s_max	s_avg	zno_math	zno_ukr	zno_oth	semester	pred
0	1.0	0.0	0.026316	0.50	0.286624	0.37	0.345550	0.432990	0.2	76.00
1	0.0	1.0	0.605263	0.75	0.662420	0.75	0.806283	0.711340	0.2	79.25
2	0.0	0.0	0.368421	0.75	0.522293	0.44	0.785340	0.824742	0.2	76.00
3	1.0	0.0	0.131579	1.00	0.592357	0.63	0.680628	0.164948	0.4	92.50
4	1.0	1.0	0.368421	0.75	0.585987	0.95	0.753927	0.721649	0.0	83.75

Regression

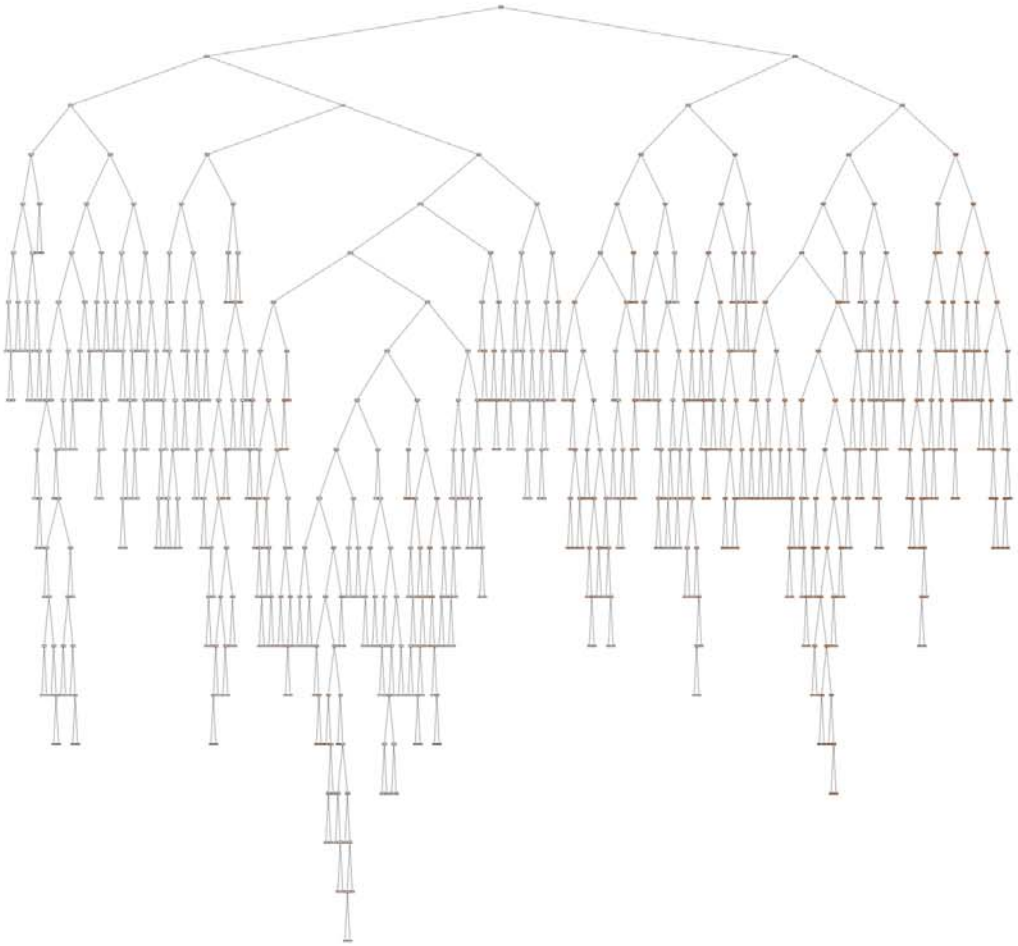
Build a Random Forest Regressor

In [10]:
```
features = ['s_min', 's_max', 's_avg', 'zno_math', 'zno_ukr', 'zno_oth', 'semes

X_train = df_train[features]
y_train = df_train['pred']

X_test = df_test[features]
y_test = df_test['pred']

model = RandomForestRegressor(n_estimators=80, random_state=0, max_depth=25)

cv = KFold(n_splits=5)

for train_index, test_index in cv.split(X_train):

    X_tr, X_te = X_train.iloc[train_index,:],X_train.iloc[test_index,:]
    y_tr, y_te = y_train[train_index], y_train[test_index]

    model.fit(X_tr, y_tr)

    y_pred = model.predict(X_te)

    print(f'MSE: {mean_squared_error(y_te, y_pred)} R2: {r2_score(y_te, y_pred)


y_pred_train = model.predict(X_train)

print(f'TRAIN MSE: {mean_squared_error(y_pred_train, y_train)} R2: {r2_score(y_

y_pred_test = model.predict(X_test)

print(f'TEST MSE: {mean_squared_error(y_pred_test, y_test)} R2: {r2_score(y_pre
```

```
MSE: 51.89982386788923 R2: 0.466106084711861
MSE: 45.11621058663922 R2: 0.5923377074319085
MSE: 34.33437682370106 R2: 0.688804993315107
MSE: 41.68949524661144 R2: 0.6714727968096523
MSE: 33.94130600527108 R2: 0.6783153193706783
TRAIN MSE: 11.753304478571964 R2: 0.864145898841588
TEST MSE: 33.93499713541667 R2: 0.5517925196544549
```

Plotting the result

In [11]:
```
plt.figure(figsize=(60,60))
_ = tree.plot_tree(model.estimators_[0], feature_names=features, filled=True)
```



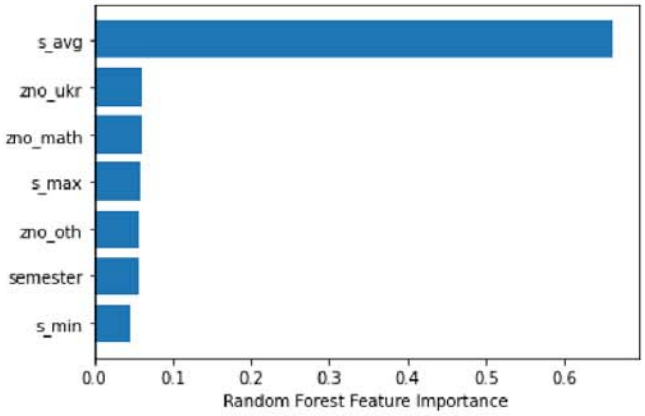
In [12]:
```
stats = pd.DataFrame([np.array(model.feature_names_in_), np.array(model.feature
stats.rename(columns=stats.iloc[0], inplace = True)
stats.drop(stats.index[0], inplace = True)
stats
```

Out[12]:

	s_min	s_max	s_avg	zno_math	zno_ukr	zno_oth	semester
1	0.046066	0.058599	0.661042	0.059956	0.060757	0.057353	0.056227

In [13]:
```
sorted_idx = model.feature_importances_.argsort()
plt.barh(model.feature_names_in_[sorted_idx], model.feature_importances_[sorted
plt.xlabel("Random Forest Feature Importance")
```

Out[13]:
```
Text(0.5, 0, 'Random Forest Feature Importance')
```



In [14]:
```
with open('rfregressor.pkl','wb') as f:
    pickle.dump(model,f)
```

Linear Regression

In [19]:
```
features = ['gender', 'funding', 's_min', 's_avg', 'zno_math', 'zno_ukr', 'zno_

X_train = df_train[features]
y_train = df_train['pred']

X_test = df_test[features]
y_test = df_test['pred']
```

In [20]:
```
regression = linear_model.LinearRegression()

cv = KFold(n_splits=4)

for train_index, test_index in cv.split(X_train):
    # Splitting the dataset
    X_tr, X_te = X_train.iloc[train_index,:],X_train.iloc[test_index,:]
    y_tr, y_te = y_train[train_index], y_train[test_index]

    # For training, fit() is used
    regression.fit(X_tr, y_tr)

    # For other metrics, we need the predictions of the model
    y_pred_test = regression.predict(X_te)
    print(f'MSE_test: {mean_squared_error(y_te, y_pred_test)} R2_test: {r2_scor

#regression.fit(X_train, y_train)

y_pred_train = regression.predict(X_train)

print(f'TRAIN MSE: {mean_squared_error(y_pred_train, y_train)} R2: {r2_score(y_

y_pred_test = regression.predict(X_test)

print(f'TEST MSE: {mean_squared_error(y_pred_test, y_test)} R2: {r2_score(y_pre
```

```
MSE_test: 46.8547508465935 R2_test: 0.537440511429673
MSE_test: 46.99408862162545 R2_test: 0.5850615221020523
MSE_test: 36.72971675889268 R2_test: 0.6859465252951928
MSE_test: 33.186429785976486 R2_test: 0.6975455864386995
TRAIN MSE: 39.960508635020304 R2: 0.4149355974724308
TEST MSE: 31.8142361214029 R2: 0.541669695509721
```

In [21]:
```
stats = pd.DataFrame([np.array(regression.feature_names_in_), np.array(regressi
stats.rename(columns=stats.iloc[0], inplace = True)
stats.drop(stats.index[0], inplace = True)
stats
```

Out[21]:

	gender	funding	s_min	s_avg	zno_math	zno_ukr	zno_oth	semester
1	-2.711879	1.694437	1.147484	24.288484	-0.688872	2.893487	0.227572	-0.190907

In [22]:
```
with open('linreg.pkl','wb') as f:
    pickle.dump(regression,f)
```

SVR

In [23]:
```
features = ['gender', 'funding', 's_min', 's_avg', 'zno_math', 'zno_ukr', 'zno_

X_train = df_train[features]
y_train = df_train['pred']

X_test = df_test[features]
y_test = df_test['pred']

regressor = SVR(kernel = 'rbf', C=300, gamma=0.5, epsilon=0.1)

regressor.fit(X_train, y_train)

y_pred_train = regressor.predict(X_train)

print(f'TRAIN MSE: {mean_squared_error(y_pred_train, y_train)} R2: {r2_score(y_

y_pred_test = regressor.predict(X_test)

print(f'TEST MSE: {mean_squared_error(y_pred_test, y_test)} R2: {r2_score(y_pre
```

```
TRAIN MSE: 30.634707398663085 R2: 0.6704471097289042
TEST MSE: 32.38683092990945 R2: 0.6564113321344736
```

In [24]:
```
with open('svrr.pkl','wb') as f:
    pickle.dump(regressor,f)
```

Multi-Layer Perceptron

In [339…
```
features = ['gender', 'funding', 's_min', 's_avg', 'zno_math', 'zno_ukr', 'zno_

X_train = df_train[features]
y_train = df_train['pred']

X_test = df_test[features]
y_test = df_test['pred']
```

In [360…
```
def mlp_model(X, Y):

    estimator=MLPRegressor()
```

```
    param_grid = {'hidden_layer_sizes': [(40,40,40,40,40,40,40,40)],
              'activation': ['relu'],
              'alpha': [0.1, 0.05],
              'learning_rate': ['constant','adaptive'],
              'solver': ['adam']}

    gsc = GridSearchCV(
        estimator,
        param_grid,
        cv=5, scoring='neg_mean_squared_error', verbose=0, n_jobs=-1)

    grid_result = gsc.fit(X, Y)


    best_params = grid_result.best_params_

    best_mlp = MLPRegressor(hidden_layer_sizes = best_params["hidden_layer_size
                            activation =best_params["activation"],
                            solver=best_params["solver"],
                            max_iter= 1500, n_iter_no_change = 100
                  )

    scoring = {
               'abs_error': 'neg_mean_absolute_error',
               'squared_error': 'neg_mean_squared_error',
               'r2':'r2'}

    scores = cross_validate(best_mlp, X, Y, cv=5, scoring=scoring, return_train
    return scores
```

In [ ]:
```
scores = mlp_model(X_train,y_train)
```

In [364…
```
scores
```

Out[364]:
```
{'fit_time': array([19.53427386, 21.1274929 , 20.99080062, 22.3155365 , 22.046
54193]),
 'score_time': array([0.0043807 , 0.00432897, 0.00444007, 0.00453877, 0.004575
49]),
 'estimator': [MLPRegressor(hidden_layer_sizes=(40, 40, 40, 40, 40, 40, 40, 4
0), max_iter=1500,
               n_iter_no_change=100),
  MLPRegressor(hidden_layer_sizes=(40, 40, 40, 40, 40, 40, 40, 40), max_iter=1
500,
               n_iter_no_change=100),
  MLPRegressor(hidden_layer_sizes=(40, 40, 40, 40, 40, 40, 40, 40), max_iter=1
500,
               n_iter_no_change=100),
  MLPRegressor(hidden_layer_sizes=(40, 40, 40, 40, 40, 40, 40, 40), max_iter=1
500,
               n_iter_no_change=100),
  MLPRegressor(hidden_layer_sizes=(40, 40, 40, 40, 40, 40, 40, 40), max_iter=1
500,
               n_iter_no_change=100)],
 'test_abs_error': array([-5.92712845, -5.47072116, -4.69566754, -5.37182851,
-5.09027066]),
 'train_abs_error': array([-3.35713767, -4.09190116, -4.03335857, -3.48392061,
-3.64271492]),
 'test_squared_error': array([-59.48741707, -48.8451336 , -36.30755473, -48.18
283758,
        -43.09155576]),
 'train_squared_error': array([-19.79968892, -28.2463561 , -27.28603689, -20.4
5852362,
        -22.3335894 ]),
 'test_r2': array([0.38805245, 0.5586438 , 0.67092079, 0.62030308, 0.5915922
2]),
 'train_r2': array([0.82559685, 0.74372088, 0.75302277, 0.8076102 , 0.7998897
9])}
```

In [365…
```
for model in scores['estimator']:

    y_pred_train = model.predict(X_train)

    print(f'TRAIN MSE: {mean_squared_error(y_pred_train, y_train)} R2: {r2_scor

    y_pred_test = model.predict(X_test)

    print(f'TEST MSE: {mean_squared_error(y_pred_test, y_test)} R2: {r2_score(y
```

```
TRAIN MSE: 27.765855508991777 R2: 0.7061992996344186
TEST MSE: 38.39779695919623 R2: 0.5783149253589901
TRAIN MSE: 32.38096648839824 R2: 0.6286828954481148
TEST MSE: 40.20063598331831 R2: 0.5469608236900734
TRAIN MSE: 29.086003191823554 R2: 0.6431362054984493
TEST MSE: 41.34417750230823 R2: 0.489397199902462
TRAIN MSE: 25.99005741532619 R2: 0.6684235159858949
TEST MSE: 45.59756920739578 R2: 0.40207114598945537
TRAIN MSE: 26.47520288453034 R2: 0.7253861769454882
TEST MSE: 41.90307232238418 R2: 0.565045756688344
```

In [367…
```
with open('mlpregressor.pkl','wb') as f:
    pickle.dump(scores['estimator'][0],f)
```

In [368…
```
with open('mlpregressor.pkl', 'rb') as f:
    mlpregregressor = pickle.load(f)
```

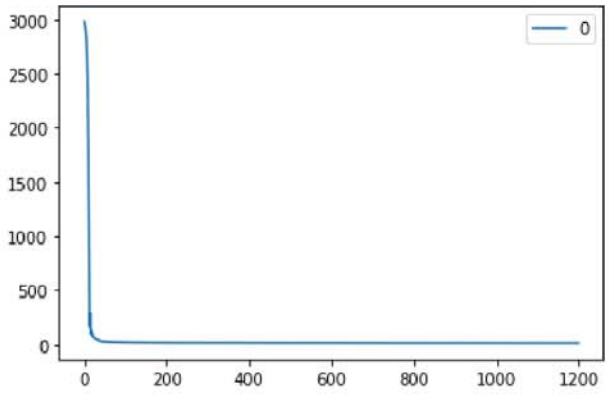
In [369…
```
y_pred_train = mlpregregressor.predict(X_train)

print(f'TRAIN MSE: {mean_squared_error(y_pred_train, y_train)} R2: {r2_score(y_
```

```
y_pred_test = mlpregregressor.predict(X_test)

print(f'TEST MSE: {mean_squared_error(y_pred_test, y_test)} R2: {r2_score(y_pre
```

```
TRAIN MSE: 27.765855508991777 R2: 0.7061992996344186
TEST MSE: 38.39779695919623 R2: 0.5783149253589901
```

In [345…
```
pd.DataFrame(scores["estimator"][0].loss_curve_).plot()
```

Out[345]:
```
<AxesSubplot:>
```



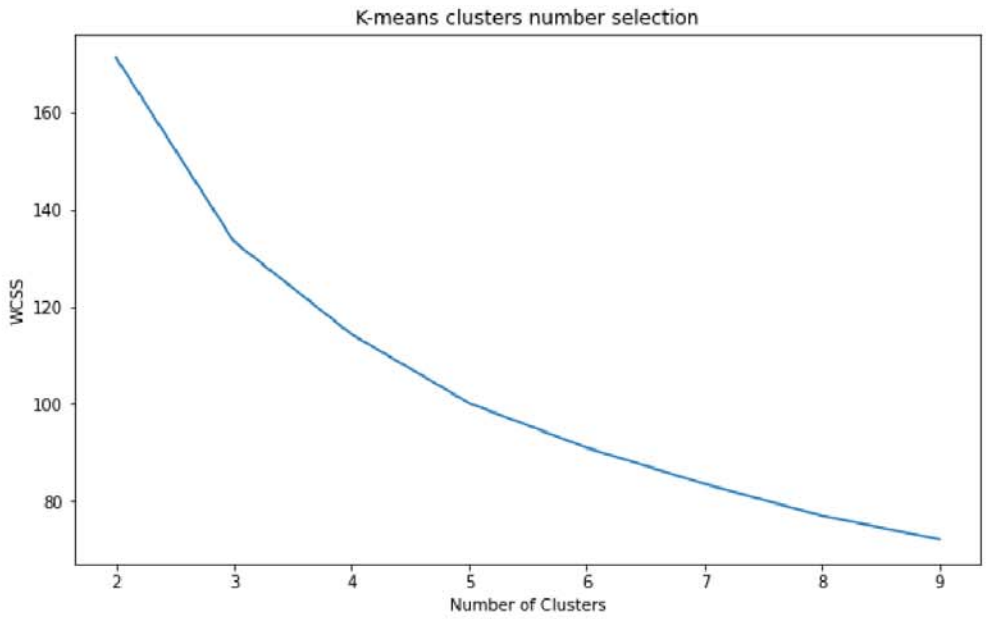
In [ ]:
```
# dtree = DecisionTreeClassifier()
# dtree = dtree.fit(X, y)
# data = tree.export_graphviz(dtree, out_file=None, feature_names=features)
# graph = pydotplus.graph_from_dot_data(data)
# graph.write_png('mydecisiontree.png')

# img=pltimg.imread('mydecisiontree.png')
# imgplot = plt.imshow(img)
# plt.show()
```

Clustering

K-means

In [25]:
```
features = ['s_min', 's_avg', 'zno_math', 'zno_ukr', 'zno_oth']
X = df0[features]
num = 10
history = list()

for i in range(2,num):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=0).fit(X)
    history.append(kmeans.inertia_)

plt.figure(figsize=(10,6))
plt.plot(np.array(range(2,num)), np.array(history))
plt.xlabel("Number of Clusters")
plt.ylabel("WCSS")
plt.title("K-means clusters number selection")
```

Out[25]:
```
Text(0.5, 1.0, 'K-means clusters number selection')
```

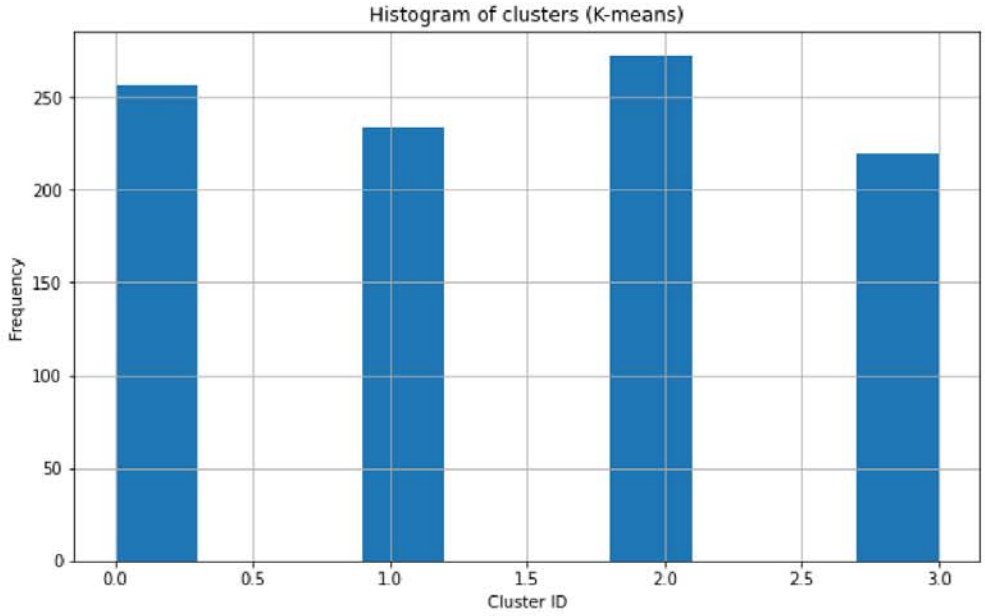


In [26]:
```
kmeans = KMeans(n_clusters=4, random_state=0).fit(X)
print(f"DB score: {davies_bouldin_score(X, kmeans.labels_)}")
X["cluster"] = kmeans.labels_

pd.DataFrame(kmeans.labels_).hist(figsize=(10,6))
plt.title("Histogram of clusters (K-means)")
plt.xlabel("Cluster ID")
plt.ylabel("Frequency")
X.head()
```

```
DB score: 1.3227220541001667
/tmp/ipykernel_42/2086331995.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/sta
ble/user_guide/indexing.html#returning-a-view-versus-a-copy
  X["cluster"] = kmeans.labels_
```

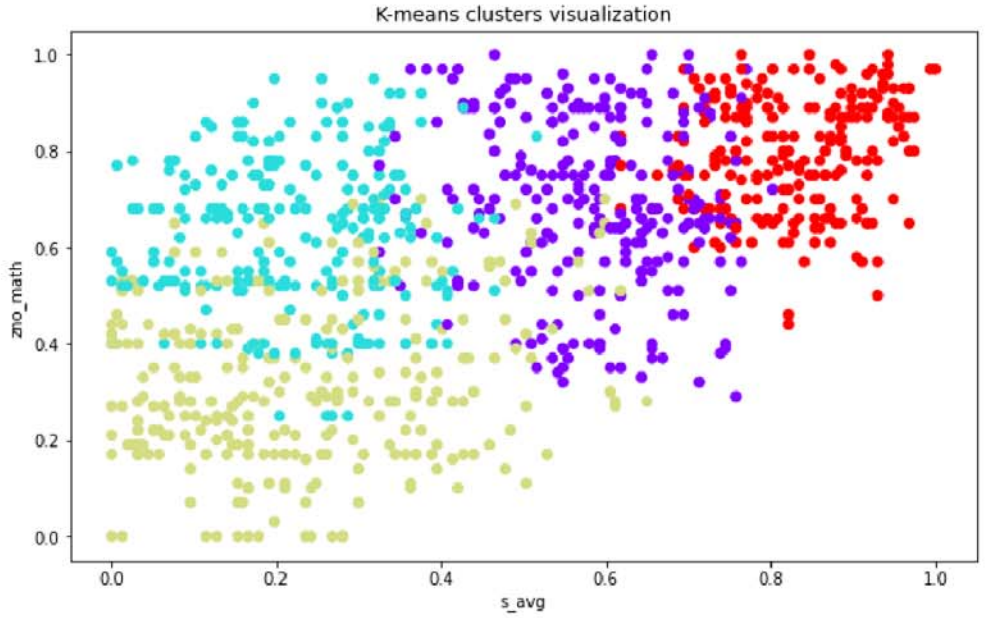
Out[26]:

	s_min	s_avg	zno_math	zno_ukr	zno_oth	cluster
0	0.000000	0.273885	0.89	0.345550	0.773196	1
1	0.131579	0.394904	0.30	0.219895	0.041237	2
2	0.657895	0.700637	0.75	0.806283	0.711340	3
3	0.289474	0.547771	0.75	0.680628	0.618557	0
4	0.078947	0.324841	0.70	0.544503	0.061856	2



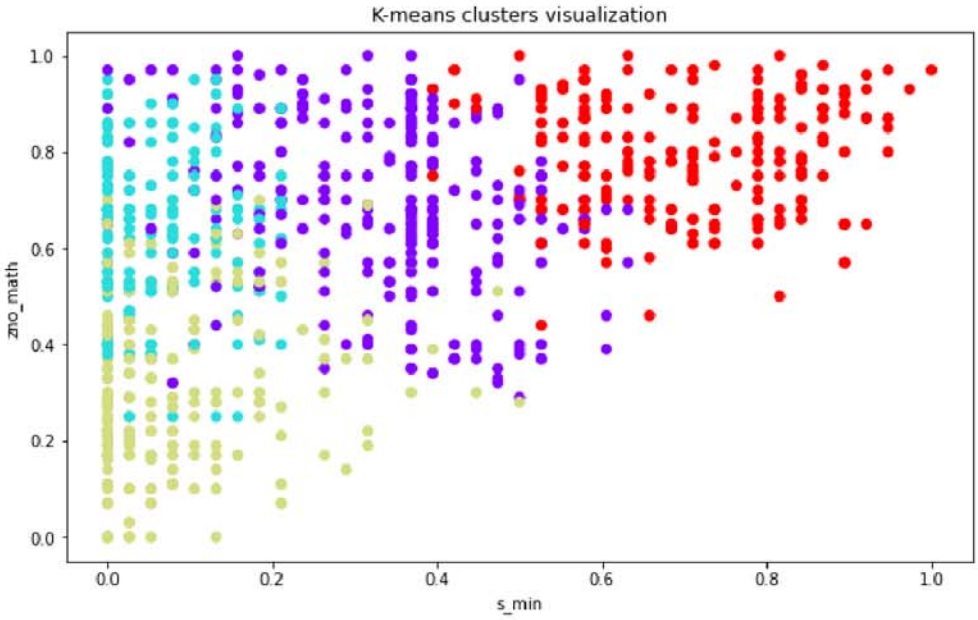
In [27]:
```
plt.figure(figsize=(10,6))
plt.scatter(X["s_avg"],X["zno_math"], c=kmeans.labels_, cmap='rainbow')
plt.title("K-means clusters visualization")
plt.xlabel("s_avg")
plt.ylabel("zno_math")
```

Out[27]:
```
Text(0, 0.5, 'zno_math')
```

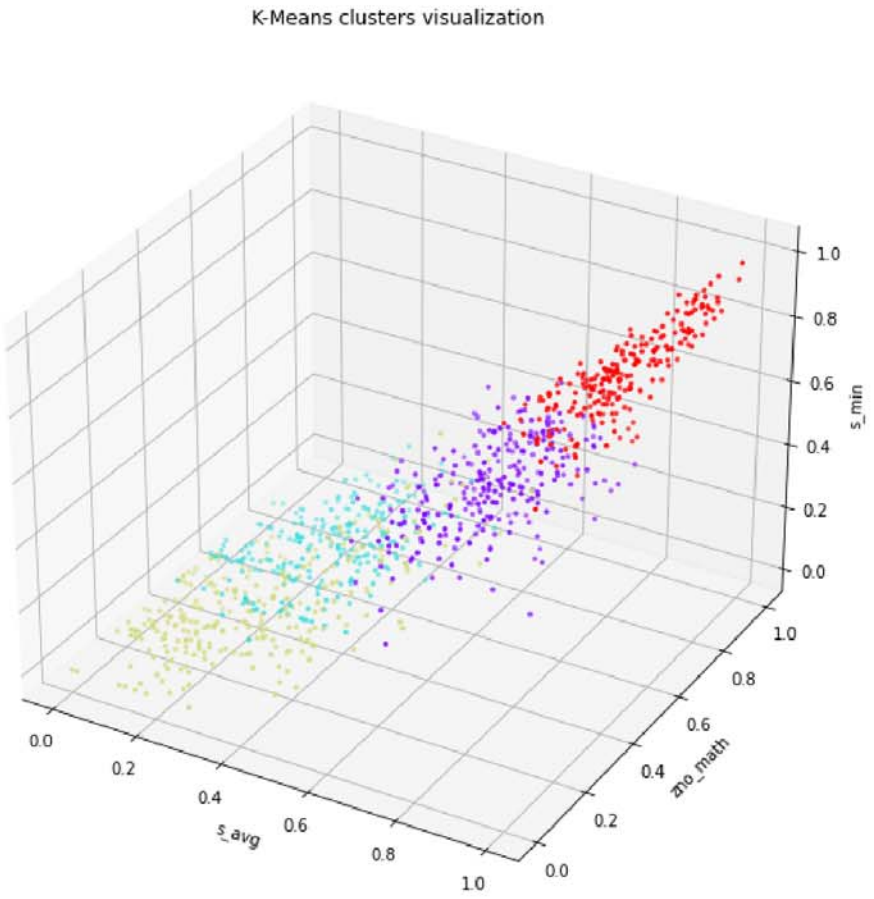


In [28]:
```
plt.figure(figsize=(10,6))
plt.scatter(X["s_min"],X["zno_math"], c=kmeans.labels_, cmap='rainbow')
plt.title("K-means clusters visualization")
plt.xlabel("s_min")
plt.ylabel("zno_math")
```

Out[28]:
```
Text(0, 0.5, 'zno_math')
```



In [29]:
```
fig = plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')

ax.scatter(X["s_avg"], X["zno_math"], X["s_min"], c=kmeans.labels_, marker=".",

ax.set_title('K-Means clusters visualization')
ax.set_xlabel('s_avg')
ax.set_ylabel('zno_math')
ax.set_zlabel('s_min')

plt.show()
```



In [30]:
```
kmeans.cluster_centers_
```

Out[30]:
```
array([[0.33789062, 0.58541501, 0.69347656, 0.66749836, 0.71717945],
       [0.04374719, 0.22633241, 0.63564103, 0.54821676, 0.65296502],
       [0.07246517, 0.2360903 , 0.32360294, 0.37877271, 0.28839448],
       [0.70909091, 0.82918356, 0.79518182, 0.80604474, 0.77253983]])
```

In [31]:
```
with open('kmeans.pkl','wb') as f:
    pickle.dump(kmeans,f)
```

In [36]:
```
labeled_df = df0.copy()
labeled_df["cluster"] = kmeans.labels_
labeled_df.drop(columns=["pred"], axis=1, inplace=True)

labeled_df.to_csv('labeled.csv', index=False)
```

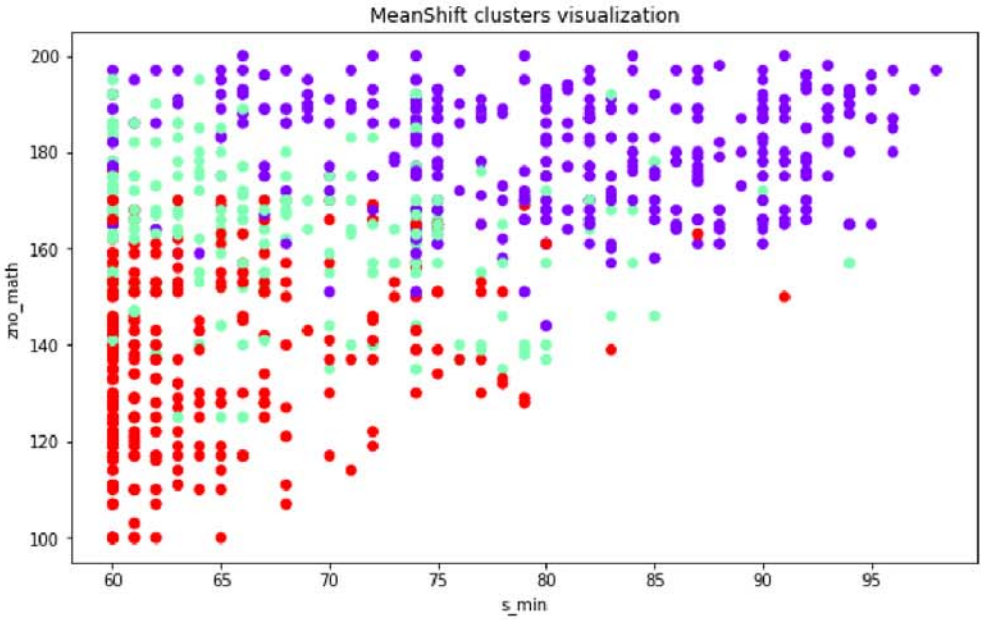
Meanshift

In [557…
```
features = ['s_min', 's_max', 's_avg', 'zno_math', 'zno_ukr', 'zno_oth']
X = df[features]
meanshift = MeanShift(bandwidth=28).fit(X)
print(f"Clusters found: {max(meanshift.labels_)+1}")
print(f"DB score: {davies_bouldin_score(X, meanshift.labels_)}")

plt.figure(figsize=(10,6))
plt.scatter(X["s_min"],X["zno_math"], c=meanshift.labels_, cmap='rainbow')
```

```
plt.title("MeanShift clusters visualization")
plt.xlabel("s_min")
plt.ylabel("zno_math")
```

```
Clusters found: 3
DB score: 1.400923264655348
```

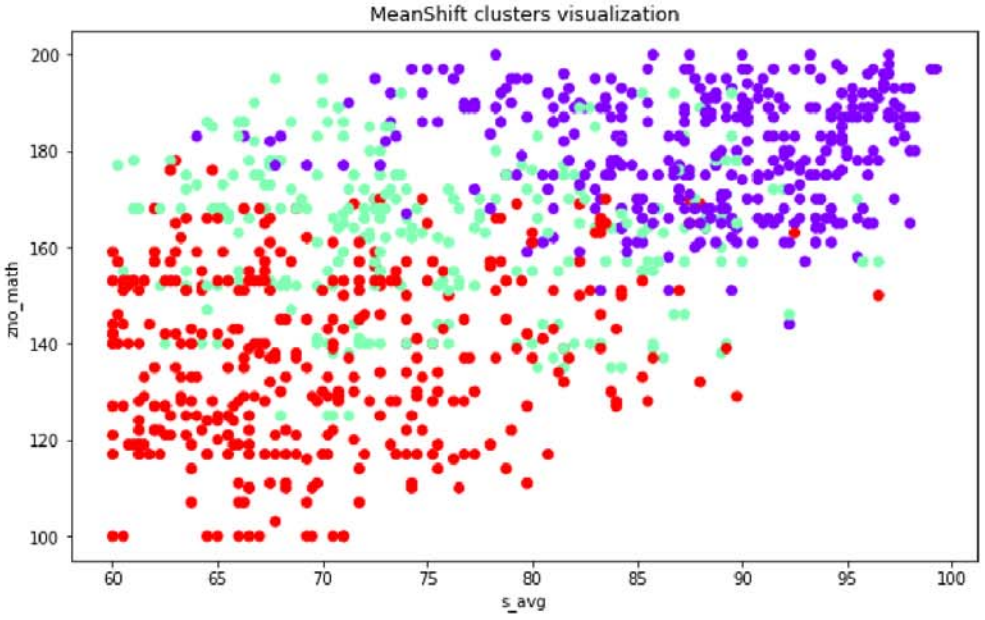
Out[557]: Text(0, 0.5, 'zno_math')



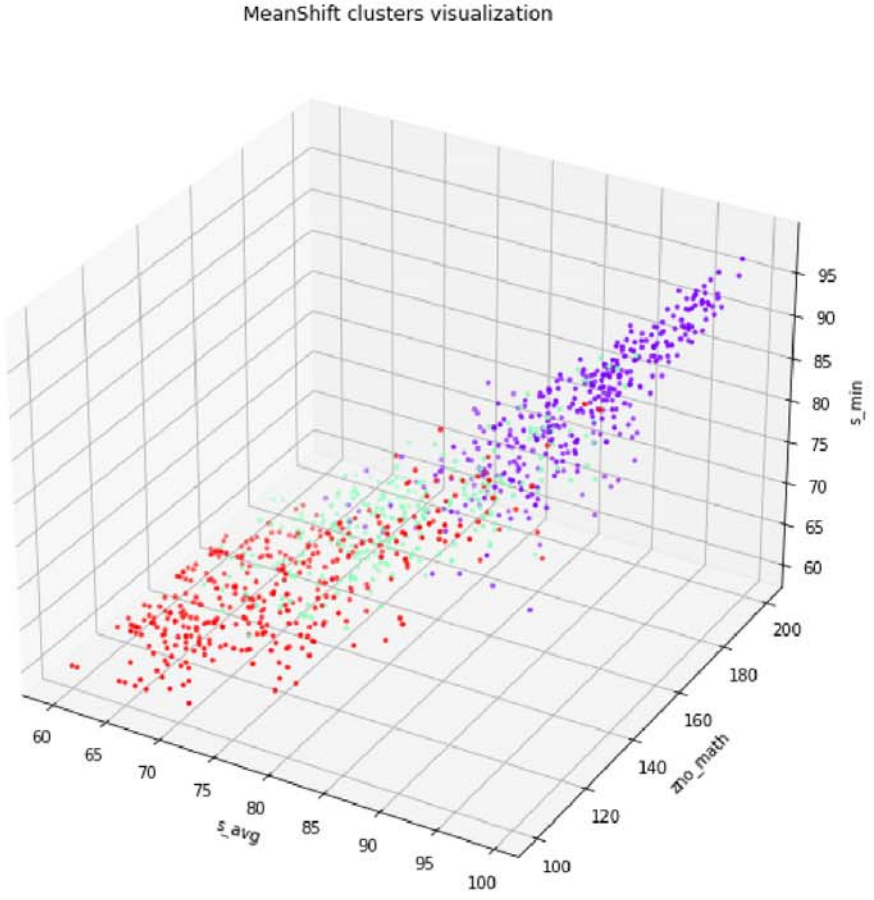
In [558…

```
plt.figure(figsize=(10,6))
plt.scatter(X["s_avg"],X["zno_math"], c=meanshift.labels_, cmap='rainbow')
plt.title("MeanShift clusters visualization")
plt.xlabel("s_avg")
plt.ylabel("zno_math")
```

Out[558]: Text(0, 0.5, 'zno_math')

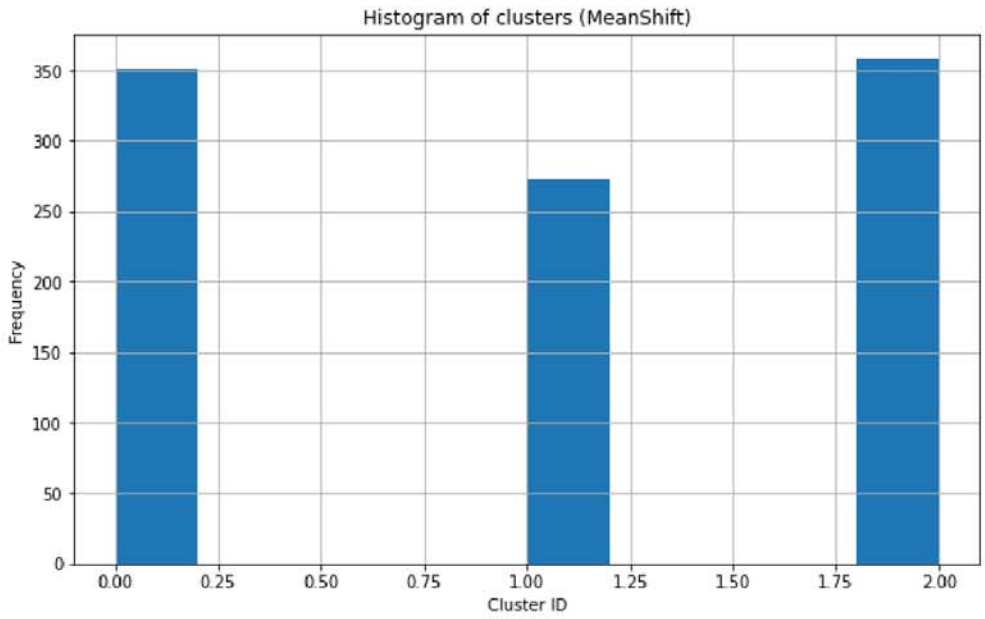


In [559…
```
fig = plt.figure(figsize=(10,10))
ax = fig.add_subplot(111, projection='3d')

ax.scatter(X["s_avg"], X["zno_math"], X["s_min"], c=meanshift.labels_, marker="

ax.set_title('MeanShift clusters visualization')
ax.set_xlabel('s_avg')
ax.set_ylabel('zno_math')
ax.set_zlabel('s_min')

plt.show()
```



In [560…
```
pd.DataFrame(meanshift.labels_).hist(figsize=(10,6))
plt.title("Histogram of clusters (MeanShift)")
plt.xlabel("Cluster ID")
plt.ylabel("Frequency")
```

Out[560]:
```
Text(0, 0.5, 'Frequency')
```



In [561…
```
meanshift.cluster_centers_
```

Out[561]:
```
array([[ 80.24110672,  95.29249012,  87.91600791, 181.35968379,
        180.06719368, 177.53359684],
       [ 64.95683453,  84.30215827,  73.51978417, 164.75539568,
        159.41726619, 162.41726619],
       [ 62.89928058,  78.32374101,  69.51978417, 145.66906475,
        151.60431655, 138.5971223 ]])
```

In [562…
```
with open('meanshift.pkl','wb') as f:
    pickle.dump(meanshift,f)
```

Classification

Load labeled dataset

In [91]:
```
dfc = pd.read_csv('labeled.csv')
dfc.head()
```

Out[91]:

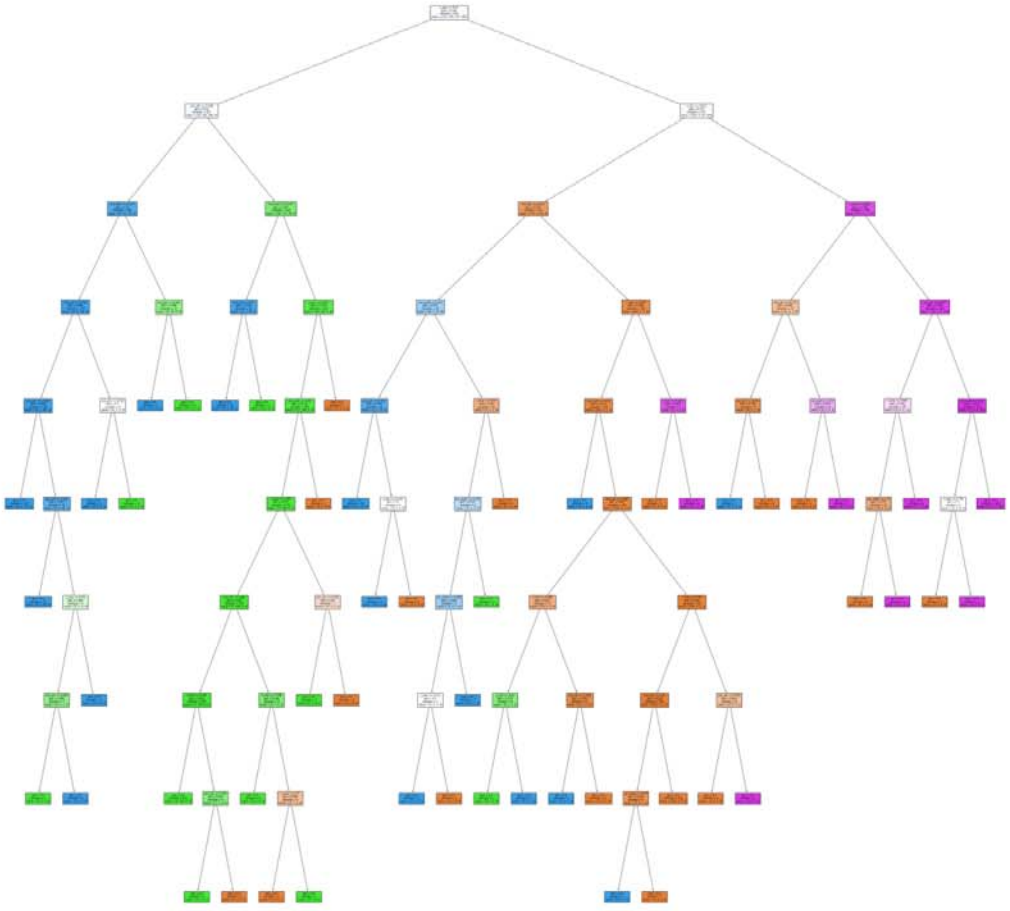
	gender	funding	s_min	s_max	s_avg	zno_math	zno_ukr	zno_oth	semester	cluster
0	1.0	1.0	0.000000	0.600	0.273885	0.89	0.345550	0.773196	0.4	1
1	1.0	0.0	0.131579	0.675	0.394904	0.30	0.219895	0.041237	0.2	2
2	0.0	1.0	0.657895	0.750	0.700637	0.75	0.806283	0.711340	0.0	3
3	1.0	1.0	0.289474	0.925	0.547771	0.75	0.680628	0.618557	0.0	0
4	1.0	0.0	0.078947	0.800	0.324841	0.70	0.544503	0.061856	0.4	2

In [92]:
```
test_size = 150

df0 = shuffle(dfc, random_state=0).reset_index()
df0.drop(columns=['index'],axis=1,inplace=True)
dfc_train = df0.head(982-test_size).reset_index(drop=True)
dfc_test = df0.tail(test_size).reset_index(drop=True)
```

Decision Tree

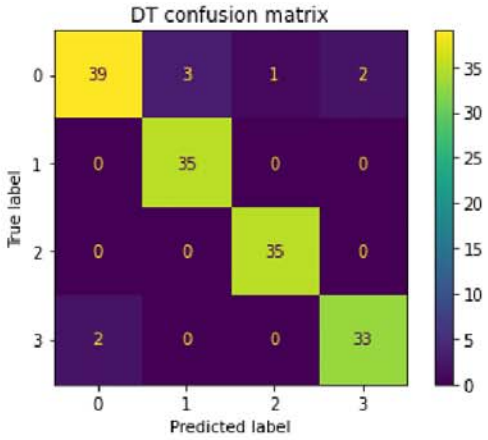
In [93]:
```
features = ['s_min', 's_max', 's_avg', 'zno_math', 'zno_ukr', 'zno_oth']

X_train, y_train = dfc_train[features], dfc_train['cluster']
X_test, y_test = dfc_test[features], dfc_test['cluster']

clf = DecisionTreeClassifier()
clf = clf.fit(X_train, y_train)
```

In [94]:
```
plt.figure(figsize=(60,60))
_ =tree.plot_tree(clf, feature_names=features, filled=True)
```



In [113…
```
plot_confusion_matrix(clf, X_test, y_test)
plt.title("DT confusion matrix")
plt.show()
```

```
/usr/local/lib/python3.8/site-packages/sklearn/utils/deprecation.py:87: FutureW
arning: Function plot_confusion_matrix is deprecated; Function `plot_confusion_
matrix` is deprecated in 1.0 and will be removed in 1.2. Use one of the class m
ethods: ConfusionMatrixDisplay.from_predictions or ConfusionMatrixDisplay.from_
estimator.
  warnings.warn(msg, category=FutureWarning)
```



In [98]:
```
y_pred_train = clf.predict(X_train)
y_pred_test = clf.predict(X_test)
print(f"F1 train: {f1_score(y_pred_train, y_train, average='weighted')}")
print(f"F1 test: {f1_score(y_pred_test, y_test, average='weighted')}")
print(f"Acc train: {balanced_accuracy_score(y_pred_train, y_train)}")
print(f"Acc test: {balanced_accuracy_score(y_pred_test, y_test)}")
```

```
F1 train: 1.0
F1 test: 0.947449069483269
Acc train: 1.0
Acc test: 0.9468378772133585
```

In [131…
```
with open('dtclassifier.pkl','wb') as f:
    pickle.dump(clf,f)
```

SVM Classifier

In [108…
```
features = ['s_min', 's_max', 's_avg', 'zno_math', 'zno_ukr', 'zno_oth']

X_train, y_train = dfc_train[features], dfc_train['cluster']
X_test, y_test = dfc_test[features], dfc_test['cluster']

svc = SVC(gamma=1.1)
svc.fit(X_train, y_train)
```

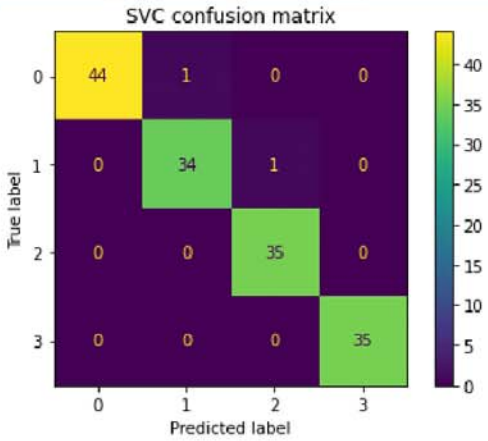
Out[108]:
```
SVC(gamma=1.1)
```

In [109…
```
y_pred_train = svc.predict(X_train)
y_pred_test = svc.predict(X_test)
print(f"F1 train: {f1_score(y_pred_train, y_train, average='weighted')}")
print(f"F1 test: {f1_score(y_pred_test, y_test, average='weighted')}")
print(f"Acc train: {balanced_accuracy_score(y_pred_train, y_train)}")
print(f"Acc test: {balanced_accuracy_score(y_pred_test, y_test)}")
```

```
F1 train: 0.986772613261244
F1 test: 0.9866571714933798
Acc train: 0.987049799543754
Acc test: 0.9859126984126985
```

In [112…
```
plot_confusion_matrix(svc, X_test, y_test)
plt.title("SVC confusion matrix")
plt.show()
```

```
/usr/local/lib/python3.8/site-packages/sklearn/utils/deprecation.py:87: FutureW
arning: Function plot_confusion_matrix is deprecated; Function `plot_confusion_
matrix` is deprecated in 1.0 and will be removed in 1.2. Use one of the class m
ethods: ConfusionMatrixDisplay.from_predictions or ConfusionMatrixDisplay.from_
estimator.
  warnings.warn(msg, category=FutureWarning)
```



In [132…
```
with open('svc.pkl','wb') as f:
    pickle.dump(svc,f)
```

Naive Bayes

In [114…
```
from sklearn.naive_bayes import GaussianNB
```

In [115…
```
features = ['s_min', 's_max', 's_avg', 'zno_math', 'zno_ukr', 'zno_oth']

X_train, y_train = dfc_train[features], dfc_train['cluster']
X_test, y_test = dfc_test[features], dfc_test['cluster']

#Create a Gaussian Classifier
gnb = GaussianNB()

#Train the model using the training sets
gnb.fit(X_train, y_train)
```

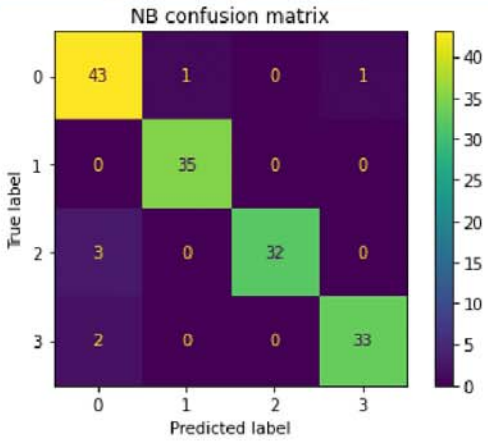
Out[115]:
```
GaussianNB()
```

In [119…
```
y_pred_train = gnb.predict(X_train)
y_pred_test = gnb.predict(X_test)
print(f"F1 train: {f1_score(y_pred_train, y_train, average='weighted')}")
print(f"F1 test: {f1_score(y_pred_test, y_test, average='weighted')}")
print(f"Acc train: {accuracy_score(y_pred_train, y_train)}")
print(f"Acc test: {accuracy_score(y_pred_test, y_test)}")
```

```
F1 train: 0.9439387552478229
F1 test: 0.9531263855347445
Acc train: 0.9447115384615384
Acc test: 0.9533333333333334
```

In [120…
```
plot_confusion_matrix(gnb, X_test, y_test)
plt.title("NB confusion matrix")
plt.show()
```

```
/usr/local/lib/python3.8/site-packages/sklearn/utils/deprecation.py:87: FutureW
arning: Function plot_confusion_matrix is deprecated; Function `plot_confusion_
matrix` is deprecated in 1.0 and will be removed in 1.2. Use one of the class m
ethods: ConfusionMatrixDisplay.from_predictions or ConfusionMatrixDisplay.from_
estimator.
  warnings.warn(msg, category=FutureWarning)
```



In [133…
```
with open('naivebayes.pkl','wb') as f:
    pickle.dump(gnb,f)
```

MLPClassifier

In [126…
```
features = ['s_min', 's_max', 's_avg', 'zno_math', 'zno_ukr', 'zno_oth']

X_train, y_train = dfc_train[features], dfc_train['cluster']
X_test, y_test = dfc_test[features], dfc_test['cluster']

mlpc = MLPClassifier(solver='lbfgs', alpha=1e-5,
                     hidden_layer_sizes=(10, 10, 10), random_state=0)

mlpc.fit(X_train, y_train)
```

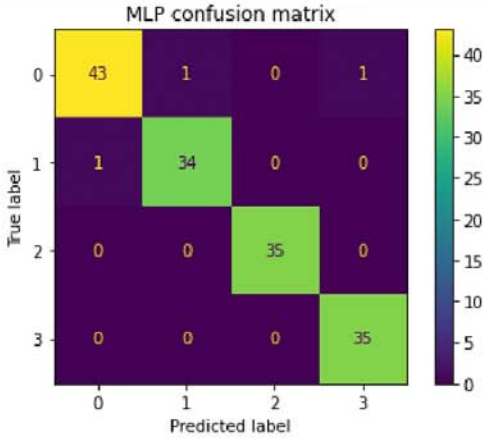
Out[126]:
```
MLPClassifier(alpha=1e-05, hidden_layer_sizes=(10, 10, 10), random_state=0,
              solver='lbfgs')
```

In [127…
```
y_pred_train = mlpc.predict(X_train)
y_pred_test = mlpc.predict(X_test)
print(f"F1 train: {f1_score(y_pred_train, y_train, average='weighted')}")
print(f"F1 test: {f1_score(y_pred_test, y_test, average='weighted')}")
print(f"Acc train: {accuracy_score(y_pred_train, y_train)}")
print(f"Acc test: {accuracy_score(y_pred_test, y_test)}")
```

```
F1 train: 0.9819923144464717
F1 test: 0.9800654111937543
Acc train: 0.9819711538461539
Acc test: 0.98
```

In [130…
```
plot_confusion_matrix(mlpc, X_test, y_test)
plt.title("MLP confusion matrix")
plt.show()
```

```
/usr/local/lib/python3.8/site-packages/sklearn/utils/deprecation.py:87: FutureW
arning: Function plot_confusion_matrix is deprecated; Function `plot_confusion_
matrix` is deprecated in 1.0 and will be removed in 1.2. Use one of the class m
ethods: ConfusionMatrixDisplay.from_predictions or ConfusionMatrixDisplay.from_
estimator.
  warnings.warn(msg, category=FutureWarning)
```



In [134…
```
with open('mlpclassifier.pkl','wb') as f:
    pickle.dump(mlpc,f)
```

In [ ]: