КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій
Кафедра технологій управління

Спеціальність 122 – Комп'ютерні науки, освітня програма «Інформаційна аналітика та впливи»

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему:

" Розробка технології прогнозування успішності завершення курсу студентом школи програмування методами Data Science"

Студента 2-го курсу групи ІАВ-21

Гамзін Нікіти Сергійовича

(прізвище, ім'я, по батькові)

Науковий керівник:

к.т.н, асистент

(науковий ступінь, вчене звання)

Хлевний Андрій Олександрович

(прізвище, ім'я, по батькові)

(підпис студента) (дата (підпис)

Попередній захист:

(Висновок: «До захисту в Екзаменаційній комісії»)

Завідувач кафедри
технологій управління

(прізвище, ініціали) (дата)

Київ – 2025

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра технологій управління

Освітньо-кваліфікаційний рівень Магістр
Спеціальність 122 - Комп'ютерні науки
Освітня програма Інформаційна аналітика та впливи

ЗАТВЕРДЖУЮ
Завідувач кафедри
професор Морозов
В.В.

_____
«____» _____20___року

З А В Д А Н Н Я
НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Студент: Гамзін Нікіта Сергійович

Група: ІАВ-21

1. **Тема кваліфікаційної роботи**

Розробка технології прогнозування успішності завершення курсу студентом школи програмування методами Data Science

Затверджена наказом по від « ___ »_____ 2025р. No

2. **Строк подання студентом готової роботи –** «__» _____ 2025 р.

3. **Цільова установка та вихідні дані до роботи**

Настільний додаток для прогнозування успішності завершення навчального курсу студентом школи програмування на основі історичних освітніх даних з використанням методів Data Science. Система реалізована як модуль прогнозної аналітики, що обробляє дані з Excel-файлів та автоматизує процес побудови прогнозу. Інтерфейс забезпечує вибір характеристик студентів, побудову моделей та візуалізацію результатів.

4. Зміст роботи

Розробка технології Data Science для передбачення академічної успішності студентів, зокрема в контексті освітніх програм з програмування. Зокрема, поставлено задачі прогнозування запису на курс, виключення з курсу, впродовж навчання та знаходження найкращих кандидатів для переходу на наступний курс. Особливу увагу приділено аналізу факторів, які впливають на завершення навчального курсу, включаючи відвідуваність, результати тестів, динаміку виконання завдань та інші навчальні показники.

Результати моделювання інтегровано у настільний додаток, створений як інструмент для прийняття рішень у сфері освітнього менеджменту. Програма дозволяє користувачеві завантажити дані, обрати параметри моделі, отримати прогноз завершення курсу конкретним студентом і візуалізувати результати у зрозумілому форматі впливають на завершення навчального курсу, включаючи відвідуваність, результати тестів, динаміку виконання завдань та інші навчальні показники.

5. Перелік графічного матеріалу (слайдів)

Загалом робота містить 55 рисунків, 20 слайдів.

Перелік слайдів: мета (1 слайд), задачі дослідження (1 слайд), об'єкт і предмет (1 слайд), наукова новизна та практична цінність (1 слайд), побудова та оцінювання моделей (7 слайдів), опис настільного додатку (7 слайди), відгук з підприємства (1 слайд), висновки (1 слайд).

6. Календарний план виконання роботи:

№ з/п	Назва частин роботи	%	Виконання роботи	
			За планом	Фактично
1	Вибір теми дипломної роботи	5	01.10.24	01.10.24
2	Протокол кафедри ТУ про затвердження тем дипломних робіт та призначення наукових керівників	5	27.12.24	27.12.24
3	Формування переліку нормативних матеріалів, літератури з проблематики дипломної роботи	6	24.02.25	24.02.25

4	Складання розгорнутого плану кваліфікаційної роботи	8	25.02.25	25.02.25
5	Ознайомлення наукового керівника з розгорнутим планом кваліфікаційної роботи. Внесення змін.	4	27.02.25	27.02.25
6	Підготовка розділу 1 «Постановка задачі та аналіз рішення».	10	09.04.25	09.04.25
7	Підготовка розділу 2 «Аналіз методів вирішення задачі»	10	18.04.25	18.04.25
8	Підготовка розділу 3 «Реалізація настільного додатку на основі реальних даних»	20	30.04.25	30.04.25
9	Оформлення кваліфікаційної роботи. Підготовка висновків і пропозицій	12	09.05.25	09.05.25
10	Передача кваліфікаційної роботи науковому керівникові	5	10.05.25	10.05.25
11	Передача кваліфікаційної роботи рецензенту для рецензування	8	12.05.25	12.05.25
12	Попередній захист кваліфікаційної роботи	7	12.05.25	12.05.25

Дата видачі завдання «__» _____ 20__ р.

Керівник роботи к.т.н., ас. Андрій ХЛЕВНИЙ

_____
(підпис)

Завдання прийняв до виконання:

Здобувач освіти групи ІАВ-21 Гамзін Нікіта

_____
(підпис)

ЗМІСТ

АНОТАЦІЯ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра технологій управління
Спеціальність 122 - Комп'ютерні науки,
освітня програма "Інформаційна аналітика та впливи"

Дипломна робота магістра Гамзіна Нікіти Сергійовича.

Тема роботи – «Розробка технології прогнозування успішності завершення курсу студентом школи програмування методами Data Science».

Мета дипломної роботи магістра – розробити додаток для прогнозування завершення курсу студентом школи програмування.

Об'єкт дослідження – прогнозування завершення курсу програмування на основі результатів навчання.

Предмет дослідження – інформаційна технологія прогнозування завершення навчального курсу студентом школи програмування на основі аналізу історичних даних.

Наукова новизна роботи – розроблено додаток, що дозволяє прогнозувати завершення курсу програмування студентом.

У роботі досліджуються існуючі підходи використання штучного інтелекту. Розробляється нова методика його використання, а також проводиться обґрунтування доцільності впровадження рішення.

Дипломна робота складається зі вступу, основної частини, яка включає три розділи, висновків, 1 додатку та списку використаних джерел. Всього налічує 100 сторінок та перелік посилань з 20 джерел на 2 сторінках.

Ключові слова: прогнозування успішності, освітні дані, машинне навчання, школа програмування, класифікація.

ВСТУП

У сучасному світі спостерігається стрімкий розвиток технологій, що супроводжується появою та еволюцією професій, безпосередньо пов'язаних з технологічним прогресом. Ці професії викликають значний інтерес серед потенційних спеціалістів, приваблюючи як інноваційною складовою та можливістю долучитися до передових технологій, так і високим рівнем матеріальної винагороди.

Однак, статистичні дані демонструють, що значний відсоток людей, які починають освоєння технологічних професій, зіштовхуються з розчаруванням при глибшому ознайомленні з практичними аспектами роботи. Це призводить до того, що суттєва частка початківців припиняє навчання та повертається до більш традиційних сфер діяльності.

Дана тенденція створює комплексну проблему для освітніх закладів та компаній, що спеціалізуються на підготовці фахівців у галузі інформаційних технологій. Перед ними постає критичне завдання: розробка ефективних методів відбору та оцінки потенційних студентів, які з високою ймовірністю завершать повний цикл навчання та успішно інтегруються в професійне середовище.

Маючи безпосередній досвід роботи на посаді викладача та методиста в одній з провідних приватних шкіл програмування для дітей, можу підтвердити актуальність даної проблематики. Варто підкреслити, що успішне завершення навчальних програм студентами має вагоме значення не лише з комерційної точки зору, але й з позиції якісного розвитку галузі в цілому. Ефективне навчання можливе лише за умови високої вмотивованості та щирого інтересу студента до обраної спеціальності.

Аналогічна проблематика спостерігається і в державному секторі освіти, де передчасне припинення навчання студентами контрактної форми має негативний вплив як на фінансові показники закладу, так і на ефективність використання освітніх ресурсів.

Враховуючи вищезазначене, метою даного дослідження є розробка комплексної технології прогнозування успішності завершення навчального курсу студентами шкіл програмування, що базується на багатофакторному аналізі особистісних характеристик, когнітивних здібностей та мотиваційних аспектів потенційних здобувачів освіти.

З метою практичної реалізації результатів дослідження, передбачається розробка програмного забезпечення з інтуїтивно зрозумілим користувацьким інтерфейсом для настільних комп'ютерів. Даний інструментарій надасть можливість менеджерам освітніх закладів ефективно здійснювати аналіз та прогнозування успішності студентів на основі розробленої технології.

Настільний програмний комплекс представляє собою спеціалізоване програмне забезпечення, що функціонує в локальному середовищі персонального комп'ютера. Дане рішення характеризується високим рівнем продуктивності та безпеки, оскільки основна обробка даних відбувається на стороні клієнта. Сучасні настільні додатки здатні забезпечувати широкий спектр функціональних можливостей: від складних бізнес-операцій та аналітичних інструментів до систем управління навчальним процесом та інтегрованих середовищ розробки.

Ключові функціональні вимоги до розроблюваного програмного забезпечення включають:

- Ергономічний користувацький інтерфейс, що відповідає сучасним стандартам UX/UI дизайну та забезпечує максимальну ефективність роботи користувача.
- Комплексна система інтеграції з існуючими інформаційними системами навчального закладу, включаючи CRM-системи, системи управління навчальним процесом та аналітичні платформи.
- Розширені можливості для внесення, редагування та категоризації даних про студентів, включаючи їх академічні показники, психологічні характеристики та поведінкові патерни.

- Функціонал для створення та управління ієрархічною структурою навчальних груп з можливістю призначення відповідальних викладачів та менеджерів.
- Багатофакторна система прогнозування, що включає аналіз потенціалу абітурієнтів, ризиків відрахування та ймовірності успішного завершення навчальної програми, базуючись на розроблених математичних моделях та алгоритмах машинного навчання.

Для проведення дослідження буде використано два фундаментальних типи бізнес-моделей освітніх установ, кожна з яких має свої специфічні характеристики та цільові показники ефективності.

Перша модель – "модель утримання" базується на стратегії максимізації тривалості навчання студента на курсі програмування. Ключовим аспектом даної моделі є рання ідентифікація потенційно ризикових студентів та впровадження превентивних заходів щодо їх підтримки. Це включає моніторинг академічної успішності, рівня залученості та психоемоційного стану студента, а також своєчасне застосування коригувальних педагогічних інтервенцій.

Друга модель – "модель селективного відбору" орієнтована на виявлення та залучення студентів з найвищим потенціалом до успішного завершення повного курсу навчання. Дана модель передбачає комплексну оцінку когнітивних здібностей, мотиваційних факторів та попереднього досвіду кандидатів. Особлива увага приділяється аналізу природних схильностей до програмування та здатності до засвоєння складного технічного матеріалу.

Важливо зазначити, що обидві моделі мають свої переваги та обмеження, і їх ефективність значною мірою залежить від специфіки навчального закладу, цільової аудиторії та ринкових умов.

Для досягнення мети визначено наступні ключові задачі:

1. Проаналізувати існуючі системи прогнозування успішності студентів, виділити їх основні характеристики, переваги та недоліки.
2. Описати функціональні вимоги до майбутньої системи.
3. Обрати технології та інструменти для розробки настільного додатку.

4. Спроектувати архітектуру системи та користувацький інтерфейс відповідно до визначених вимог.
5. Реалізувати систему прогнозування з використанням обраних технологій.
6. Провести тестування розробленої системи.

Об'єкт дослідження – процес прогнозування успішності студентів в освітніх закладах.

Предмет дослідження – методи та алгоритми прогнозування успішності студентів на основі їх характеристик та показників.

Мета дослідження – визначити найкращі методи для прогнозування успішності студентів.

Для вирішення поставлених завдань використовувалися такі методи дослідження:

- аналіз наукової літератури та існуючих рішень;
- порівняльний аналіз;
- статистичний аналіз;
- тестування.

Наукова новизна одержаних результатів –одержано програмний продукт, що дозволяє спрогнозувати, чи завершить студент навчання на курсі програмування.

Практичне значення одержаних результатів полягає у вирішені бізнес проблеми підсвічування проблемних студентів чи вибору найкращих студентів для навчання.

РОЗДІЛ 1. ПОСТАНОВКА ЗАДАЧІ ТА АНАЛІЗ РІШЕННЯ

1.1 Цикл життя студента в школі програмування

Дослідження базується на наборі даних, наданих провідною школою програмування для дітей в Україні, що спеціалізується на навчанні інформаційним технологіям. Згідно з цим, методологічна база та концепт дослідження формується з урахуванням специфіки бізнес процесів цього конкретного освітнього закладу та його організаційної структури, проте, сам результат роботи, буде можливим до впровадження і в інших освітніх закладах.

В процесах школи, найпершим кроком є реєстрація студента на курс. Цю роботу виконує менеджер кол центру. Важливо, що дана комунікація на даному етапі розвитку системи прогнозування ніяк не відслідковується, що може вносити значний хаос в оцінку студента. Наразі, система прогнозування успішності інтегрується в освітній процес з моменту першого контакту потенційного студента з викладачем на пробному занятті.

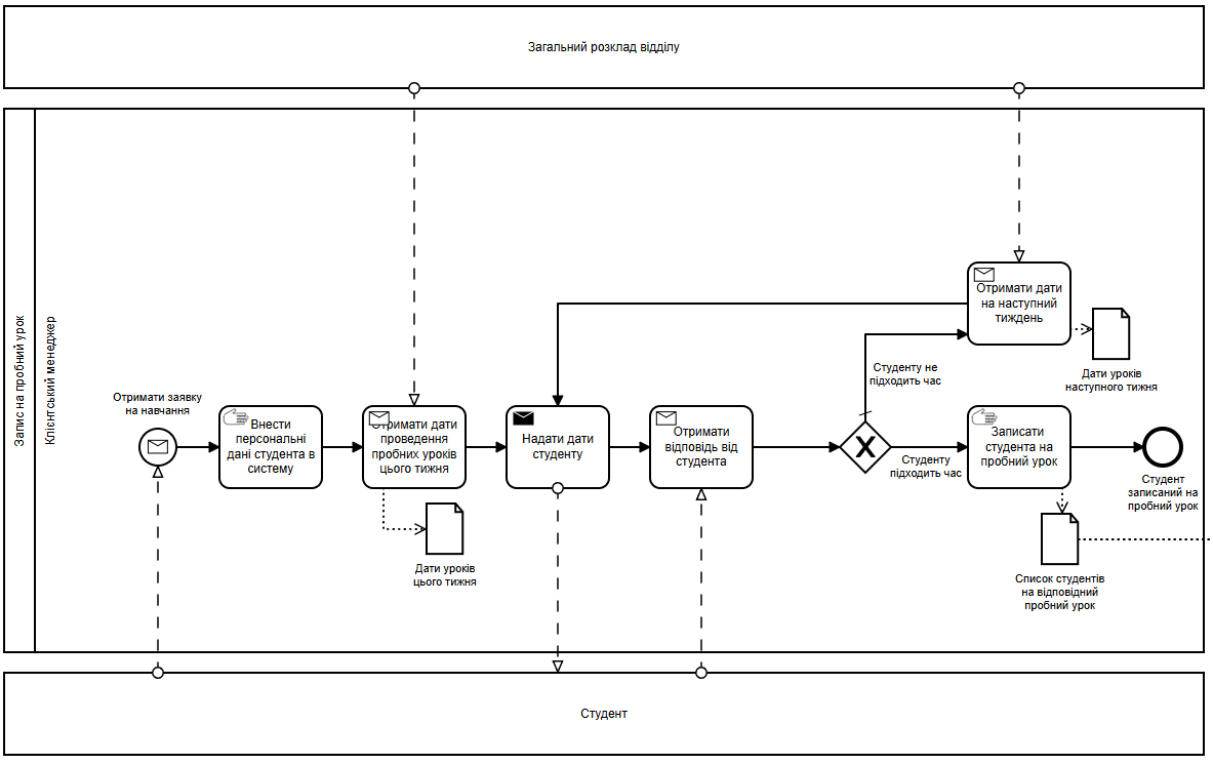


Рисунок 1.1. Приклад процесу комунікації клієнтського менеджера зі студентом

Після запису на навчальний курс, наступним кроком студента є участь в пробному занятті, де відбувається безпосередня взаємодія з викладачем, внаслідок чого, студента буде оцінено.

Навчальний процес реалізується на спеціалізованій освітній платформі, що розроблена школою, де викладач проводить заняття та іншу педагогічну діяльність(перевірка домашнього завдання, опис фідбеку уроку, тощо) та подальшу комунікацію з тьютором, надаючи коротку характеристику кожного потенційного студента, що відвідав пробне заняття. Після цього, настає останній крок очікування зворотного зв'язку та прийняття рішення від самих студентів щодо подальшого навчання в школі.

Використання власної навчальної платформи для проведення занять надає широкі можливості для імплементації системи збору та аналізу даних про студентів під час проходження першого уроку. В рамках пілотного етапу дослідження було впроваджено ряд модифікацій у існуючу платформу, що дозволило набрати первинний масив даних про поведінкові патерни, когнітивні здібності та навчальну активність потенційних студентів. Отримані дані становлять емпіричну базу для подальшого аналізу та розробки моделей.

Після позитивного рішення від студента та успішного проходження вступного тестування, його офіційно зараховують до навчальної програми. На цьому етапі відбувається важлива трансформація статусу – від потенційного абітурієнта до повноцінного студента школи програмування. Організаційна структура школи має свої особливості, однією з яких є те, що викладач, який проводить пробне заняття, може відрізнятися від основного викладача курсу. Це створює певну варіативність у процесі оцінювання та адаптації студента, оскільки різні викладачі можуть мати власні методологічні підходи та педагогічні техніки.

Після офіційного зарахування студенти починають регулярне навчання за встановленим розкладом. Школа використовує інноваційну систему мотивації: за кожен урок студент має можливість заробити бали, які накопичуються протягом всього курсу. Ця система побудована за принципом "gamification" (з англ. ігорфікація). В кінці навчального періоду накопичені бали можна

обміняти на різноманітні подарунки від школи, що стимулює активну участь та старанність. Оцінювання кожного заняття складається з двох компонентів: оцінка за виконане домашнє завдання (максимум 5 балів) та оцінка за активну роботу безпосередньо на уроці (максимум 3 бали). Така система дозволяє комплексно оцінити як самостійну роботу студента, так і його залученість до навчального процесу.

Структура кожного уроку ретельно продумана та складається з трьох взаємопов'язаних частин, кожна з яких має свою специфічну мету та значення:

1. Теоретична частина. Це фундаментальний блок заняття, під час якого студенти активно взаємодіють з викладачем, відповідають на запитання та засвоюють новий матеріал. На цьому етапі використовуються різноманітні інтерактивні методи навчання, включаючи дискусії, презентації та демонстрації практичного застосування теоретичного матеріалу.
2. Перерва. Цей етап є не просто технічною паузою, а важливою частиною навчального процесу. Під час перерви студенти мають можливість неформально спілкуватися між собою та з викладачем, беруть участь у спеціально розроблених навчальних іграх, які сприяють розвитку комунікаційних навичок та покращують групову динаміку. Такий підхід допомагає створити позитивну атмосферу та підтримувати високий рівень залученості студентів[1].
3. Практична частина. Заключний етап уроку присвячений безпосередньому застосуванню отриманих знань. Студенти працюють над практичними завданнями, які базуються на матеріалі, вивченому в теоретичній частині. Викладач виступає в ролі ментора, надаючи індивідуальні консультації та допомогу при виникненні складнощів. Це дозволяє закріпити теоретичні знання та розвинути практичні навички програмування.

Важливим елементом навчального процесу є регулярні презентації проєктів, які проводяться кожні вісім уроків. Під час цих презентацій студенти демонструють свої досягнення батькам, представляючи власні проєкти та розробки на задану викладачем тематику. Це не лише дозволяє продемонструвати прогрес у навчанні, але й розвиває навички публічних

виступів, презентації та захисту своїх ідей. Такі заходи також сприяють формуванню професійного портфоліо студента та підвищують його впевненість у власних силах.

Також, важливим аспектом навчального процесу є щотижневі зустрічі викладача з тьютором, під час яких детально обговорюється прогрес та поведінка кожного студента. Ці зустрічі мають структурований характер та включають декілька ключових компонентів аналізу.

По-перше, викладач надає детальний звіт про академічну успішність кожного студента, включаючи оцінки за виконані завдання, активність на заняттях та загальне розуміння матеріалу. Особлива увага приділяється студентам, які демонструють ознаки складнощів у навчанні або втрати інтересу до курсу.

По-друге, тьютор ділиться своїми спостереженнями щодо соціальної адаптації студентів, їхньої взаємодії з однолітками та загального емоційного стану під час навчання. Це допомагає створити більш повну картину про кожного студента та вчасно виявити потенційні проблеми[2].

На основі цих обговорень формується список студентів, які потребують додаткової уваги або підтримки. Для таких студентів розробляється індивідуальний план підтримки, який може включати додаткові консультації, зміну формату подачі матеріалу або роботу з психологом.

Варто зазначити, що поточна система оцінювання базується переважно на суб'єктивних факторах – особистому досвіді та спостереженнях викладача та тьютора. Хоча цей підхід має свої переваги, зокрема можливість врахування індивідуальних особливостей кожного студента, він також створює певні обмеження. Основним недоліком є відсутність формалізованої системи оцінювання, яка б враховувала об'єктивні показники успішності та прогресу студента.

Щотижневі дзвінки завершуються формуванням конкретних рекомендацій та плану дій для покращення навчального процесу. Це може включати зміни в методиці викладання, додаткові матеріали для окремих

студентів або організацію додаткових заходів для підвищення мотивації та залученості студентів.

Для забезпечення безперервності та ефективності навчального процесу впроваджено систему індивідуальних консультацій та відпрацювань пропущеного матеріалу. Ця система функціонує на двох рівнях: компенсаторному та корекційному. На компенсаторному рівні здійснюється відновлення пропущеного навчального матеріалу для студентів, які не змогли відвідати планові заняття. На корекційному рівні проводяться індивідуальні консультації зі студентами, які потребують додаткової педагогічної підтримки або демонструють труднощі у засвоєнні певних тематичних блоків.

Особливої уваги заслуговує аналіз фінансово-організаційної структури навчального процесу, а саме системи оплати, яка базується на восьмиурочних модулях (що відповідає двомісячному періоду навчання). Статистичний аналіз демонструє цікаву, проте логічну, закономірність у спостереженні циклічності показниках відсіву студентів, яка корелює з завершенням кожного восьмиурочного модуля. Це явище можна пояснити комплексом факторів:

- Психологічний аспект: завершення модуля створює природну «точку виходу», коли студенти схильні переоцінювати свої цілі та мотивацію.
- Фінансовий аспект: необхідність внесення оплати за наступний модуль спонукає до критичної оцінки доцільності продовження навчання.
- Академічний аспект: накопичення певного об'єму знань може створювати ілюзію достатності отриманих навичок для самостійного розвитку.

Розуміння цієї закономірності дозволяє розробляти та впроваджувати превентивні заходи для підтримки мотивації студентів та зменшення відсіву. Зокрема, особлива увага приділяється підготовці студентів до переходу між модулями, демонстрації перспектив подальшого навчання та індивідуальному супроводу тих, хто виявляє ознаки зниження зацікавленості у продовженні курсу.

По завершенні основного навчального курсу студенти проходять комплексну кваліфікаційну атестацію, яка включає виконання практичного

проекту та теоретичне обґрунтування застосованих функціональностей. Результати даної атестації слугують основою для подальшої траєкторії навчання, де студент має можливість обрати один з двох напрямків професійного розвитку: перехід до поглибленого курсу або завершення базової програми навчання.

У випадку продовження навчання, студент інтегрується в нове, складніше, освітнє середовище, що може передбачати зміну викладача. Така ротація педагогічних кадрів забезпечує диверсифікацію методологічних підходів та розширення світогляду студентів.

Особливо важливим етапом є успішне проходження трьох базових курсів, що формують фундаментальну основу для подальшої спеціалізації. За умови демонстрації високих академічних показників та стабільної мотивації, студенти отримують можливість долучитися до програми підвищеної складності – професійного курсу, що характеризується значно більшою інтенсивністю навчання та об'ємом залучених освітніх ресурсів.

Професійний курс відрізняється підвищеними вимогами до якості підготовки студентів, оскільки базується на принципах проектно-орієнтованого навчання та інтенсивної командної взаємодії. Успішність реалізації навчальних проектів безпосередньо залежить від злагодженості роботи всіх учасників освітнього процесу, їхньої здатності до ефективної комунікації та спільного вирішення складних професійних завдань.

З огляду на специфіку даного етапу навчання, особливої актуальності набуває впровадження комплексної системи селекції студентів, що базується на багатофакторному аналізі їхньої попередньої успішності, особистісних якостей та потенціалу до професійного зростання. Така система повинна включати оцінку не лише технічних компетенцій, але й розвинутих особистісних навичок, зокрема здатності до командної роботи, критичного мислення та самоорганізації.

1.2 Постановка задачі магістерської роботи

На початку XXI століття відбулася кардинальна трансформація ролі персональних комп'ютерів у професійній та освітній діяльності. Якщо на межі тисячоліть комп'ютерні системи здебільшого використовувалися для виконання рутинних офісних операцій, таких як створення текстових документів, робота з електронними таблицями, базова обробка графічних матеріалів, то сучасний етап характеризується суттєвим розширенням їхнього функціонального потенціалу. Варто відзначити, що програмне забезпечення того періоду мало суттєві обмеження щодо можливостей обробки та аналізу даних, а складні обчислювальні операції були доступні для виключно потужних серверних систем та спеціалізованих обчислювальних комплексів[3].

Революційні зміни в галузі апаратного забезпечення, зокрема збільшення обчислювальної потужності процесорів, розширення об'ємів оперативної пам'яті та впровадження твердотільних накопичувачів, у поєднанні з активним розвитком технологій Data Science та машинного навчання, створили передумови для якісно нового етапу в еволюції програмних застосунків. Особливо помітним став попит на аналітичні програмні рішення, здатні здійснювати комплексний аналіз освітніх даних в автономному режимі, без необхідності постійної інтеграції з хмарними сервісами чи зовнішніми обчислювальними ресурсами.

У сучасному розумінні, програмний застосунок представляє собою складну програмну екосистему, що розробляється з урахуванням специфіки конкретної операційної системи та встановлюється безпосередньо на локальний комп'ютер користувача. Архітектура таких додатків передбачає можливість як повністю автономної роботи, так і гібридного функціонування з вибірковим доступом до онлайн-ресурсів. Ключовими технічними характеристиками сучасних програмних рішень є глибока інтеграція з файловою системою операційної системи, оптимізовані алгоритми обробки локальних даних та здатність підтримувати високу продуктивність при роботі з масштабними інформаційними масивами[4].

В контексті освітньої аналітики програмні застосунки набули особливого значення як інструменти комплексного аналізу та моніторингу навчального

процесу. На даний момент часу, аналітичні системи включають збір та роботу з даними з найрізноманітніших джерел, включаючи електронні журнали успішності, експортовані звіти з систем управління навчанням (LMS), результати проміжного та підсумкового контролю знань[]. На основі підсумкових даних такі системи дозволяють будувати предиктивні моделі академічної успішності, здійснювати глибинний аналіз факторів ризику відсіву студентів, оцінювати ефективність різних педагогічних методик та формувати індивідуалізовані освітні траєкторії[5].

Серед ключових функціональних можливостей сучасних освітніх аналітичних систем варто відзначити наступні:

- Високоефективні алгоритми обробки локальних даних, що забезпечують миттєвий доступ до інформації та її аналіз без затримок, пов'язаних із мережевою комунікацією чи серверною обробкою.
- Розвинуті системи візуалізації даних, що включають інтерактивні дашборди, динамічні графіки та адаптивні інформаційні панелі, оптимізовані для відображення як індивідуальної успішності студентів, так і агрегованих показників навчальних груп.
- Гнучкі механізми інтеграції з існуючими інформаційними системами навчальних закладів, що не потребують розгортання додаткової серверної інфраструктури та мінімізують витрати на впровадження.
- Посилені системи захисту персональних даних та конфіденційної інформації, що відповідають сучасним вимогам до безпеки освітніх даних та забезпечують надійне зберігання оцінок, особистих файлів та інших чутливих матеріалів.
- Розширені можливості автономної роботи, що дозволяють педагогічному складу та адміністративному персоналу здійснювати аналітичну діяльність незалежно від доступності мережевого з'єднання, що особливо актуально для віддалених локацій та виїзних освітніх заходів.

Порівняно з веб-орієнтованими рішеннями, які часто характеризуються уніфікованим функціоналом та обмеженими можливостями налаштування,

програмні застосунки, в контексті настільних програмних застосунків, аналітичні системи пропонують значно ширші можливості для кастомізації. Це включає глибоке налаштування користувацького інтерфейсу, розробку спеціалізованих аналітичних модулів та адаптацію системи під специфічні потреби конкретного навчального закладу. Така гнучкість робить розробку програмних аналітичних систем особливо перспективним напрямком для освітніх установ, що прагнуть впровадити інноваційні технології управління якістю освітнього процесу.

У контексті стрімкого розвитку інформаційних технологій та зростаючої потреби у ефективній обробці даних, автоматизації бізнес-процесів та оптимізації продуктивності праці, особливу увагу привертають десктопні додатки як інструменти професійного призначення. Незважаючи на активне впровадження веб-орієнтованих та хмарних технологій, настільні програмні рішення зберігають свою актуальність завдяки унікальному поєднанню надійності, продуктивності та безпеки.

Сучасний ринок програмного забезпечення демонструє стійку тенденцію до диверсифікації рішень у сфері фінансового обліку та аналітики. Провідні розробники пропонують спеціалізовані системи, що забезпечують комплексну автоматизацію бухгалтерського обліку, формування фінансової звітності та проведення глибокого аналізу економічних показників. Важливо відзначити, що такі системи, як M.E.Doc та SendPulse, стали галузевими стандартами, демонструючи високу ефективність при роботі з масштабними масивами фінансових даних.

У сфері управління проєктами та ресурсами підприємства спостерігається значний прогрес у розробці інтегрованих рішень. Сучасні системи проєктного менеджменту надають можливість здійснювати комплексне планування, моніторинг та контроль ресурсів організації. Особливо варто відзначити розвиток функціоналу для автоматизації процесів управління людськими ресурсами, що включає модулі планування робочого навантаження, оцінки ефективності та професійного розвитку персоналу[6].

Аналітика даних та системи бізнес-аналізу представляють особливий інтерес у контексті сучасних вимог до обробки та візуалізації корпоративної інформації. Одними з найпоширеніших програмних рішень є Power BI Desktop і Tableau Desktop, які пропонують розширений функціонал для аналізу даних, їх візуалізації та побудови прогностичних моделей. Ці інструменти дозволяють організаціям приймати обґрунтовані управлінські рішення на основі комплексного аналізу бізнес показників.

Системи управління взаємовідносинами з клієнтами у форматі програмних застосунків набувають особливого значення для організацій з підвищеними вимогами до інформаційної безпеки. Локальне зберігання та обробка клієнтських даних забезпечують додатковий рівень захисту конфіденційної інформації, що є критично важливим для фінансових установ, медичних закладів та інших організацій, що працюють з чутливими даними.

У галузі логістики та управління складськими операціями програмні застосунки демонструють високу ефективність завдяки можливості автономної роботи та глибокої інтеграції з локальними системами обліку. Такі рішення забезпечують безперервність бізнес-процесів навіть в умовах обмеженого доступу до мережі Інтернет, що особливо актуально для віддалених складських приміщень та виробничих об'єктів.

Аналізуючи переваги програмних бізнес-застосунків, необхідно відзначити їх високу продуктивність при роботі з великими обсягами даних, що досягається завдяки оптимізації під конкретну апаратну платформу та можливості використання локальних обчислювальних ресурсів. Незалежність від якості інтернет-з'єднання забезпечує стабільність роботи та мінімізує ризики переривання критично важливих бізнес-процесів. Підвищений рівень захисту конфіденційної інформації досягається за рахунок локального зберігання даних та можливості впровадження додаткових заходів безпеки на рівні робочих станцій[7].

Особливу увагу слід приділити питанням гнучкості налаштування програмних застосунків, що дозволяє адаптувати програмне забезпечення під специфічні вимоги конкретної організації. Можливість тонкого налаштування

користувацького інтерфейсу, модифікації бізнес-логіки та інтеграції з існуючими інформаційними системами підприємства створює передумови для максимально ефективного використання програмного забезпечення.

Розглядаючи перспективи розвитку програмних застосунків, важливо враховувати певні обмеження, пов'язані з масштабованістю рішень та складністю процесів оновлення програмного забезпечення. Проте, активний розвиток гібридних архітектур, що поєднують переваги програмних та хмарних технологій, відкриває нові можливості для подолання цих обмежень. Інтеграція локальних додатків з хмарними сервісами дозволяє створювати гнучкі рішення, що відповідають сучасним вимогам до масштабованості та доступності корпоративних інформаційних систем.

Аналізуючи функціональне призначення та сфери застосування, можна виділити кілька ключових категорій програмного забезпечення, кожна з яких має свої унікальні характеристики та особливості використання.

Офісне програмне забезпечення становить базовий інструментарій сучасного професіонала. Наприклад, Microsoft Office і Microsoft365 надають вичерпний функціонал для введення, заповнення, зберігання та аналізу документації. При цьому, даній програмні пакети забезпечують належний рівень інструментарію для реалізації документації, включаючи складні текстові і графічні структури, складні розрахунки, моделювання процесів чи створення презентаційних матеріалів. Особливу увагу варто приділити інтегрованим можливостям спільної роботи та хмарної синхронізації, які суттєво розширюють функціональність цих інструментів.

У фінансовому секторі провідну роль відіграють спеціалізовані бухгалтерські системи. Наприклад, M.E.Doc і SendPulse надають широкі можливості, що включають, але не обмежуються введенням бухгалтерського обліку, інтеграцію з зовнішніми системами, автоматизацію звітності чи підготовку аналітичних звітів. Інтеграція з банківськими системами та можливість електронного документообігу значно підвищують ефективність фінансового менеджменту.

Системи бізнес-аналітики представляють собою потужний інструментарій для обробки та візуалізації даних. Power BI Desktop, Tableau Desktop та інші аналогічні системи дозволяють трансформувати великі масиви інформації у зрозумілі візуальні представлення, що сприяє прийняттю обґрунтованих управлінських рішень. Особливо важливою є можливість інтеграції з різноманітними джерелами даних та створення інтерактивних дашбордів.

У галузі управління проєктами програмні інструменти на кшталт Microsoft Project і спеціалізованих версій Jira забезпечують всебічний підхід до планування, моніторингу та контролю виконання завдань. Ці системи надають можливість детального ресурсного планування, відстеження прогресу та управління ризиками проєктів.

Системи класу CRM та ERP, що входять до складу корпоративних інформаційних рішень, надають широкі можливості для ведення, аналізу бізнес-процесів і формування різноманітних типів звітності. Bitrix24, SAP ERP та подібні системи надають інструменти для управління взаємовідносинами з клієнтами, автоматизації бізнес-процесів та оптимізації ресурсів підприємства[8].

Професійні графічні редактори та мультимедійні додатки складають окрему категорію програмного забезпечення, орієнтовану на створення та обробку цифрового контенту. Продукти Adobe Creative Suite, CorelDRAW та професійні аудіо-редактори надають повний спектр інструментів для реалізації творчих задач.

Інженерні та CAD-системи представляють собою спеціалізовані рішення для проєктування та моделювання. AutoCAD, SolidWorks та подібні системи забезпечують можливість створення складних технічних креслень та 3D-моделей з високою точністю та деталізацією.

Середовища розробки програмного забезпечення, такі як Visual Studio та IntelliJ IDEA, надають розробникам повноцінний інструментарій для створення, тестування та відлагодження програмних продуктів. Інтеграція з системами

контролю версій та можливості автоматизації процесів розробки значно підвищують продуктивність програмістів.

Системи управління базами даних та адміністративні інструменти забезпечують надійне зберігання та обробку даних. Такі рішення як DBeaver та HeidiSQL надають зручний інтерфейс для роботи з різними типами баз даних та їх адміністрування.

Безпека інформаційних систем забезпечується спеціалізованими додатками, включаючи антивірусне програмне забезпечення, системи моніторингу та контролю доступу. Ці інструменти є критично важливими для забезпечення конфіденційності та цілісності корпоративних даних[9].

Розглядаючи проєкт з розробки програмного забезпечення для аналізу завершення студентом курсу програмування, можна класифікувати його як гібридне рішення, що поєднує елементи аналітичних систем та освітніх платформ. Такий додаток можна віднести до категорії спеціалізованого програмного забезпечення для освітньої галузі з елементами бізнес-аналітики, оскільки він включає функціонал для збору та аналізу даних про схильності користувачів, а також надає рекомендації щодо професійного розвитку в сфері інформаційних технологій.

У контексті розробки програмного забезпечення для прогнозування успішності завершення навчального курсу студентами школи програмування, методологія створення такої системи потребує комплексного та науково обґрунтованого підходу. Дана робота присвячена детальному опису процесу розробки спеціалізованого програмного забезпечення, що базується на методах машинного навчання та аналізу даних.

Первинним етапом розробки є формування концептуальної моделі системи та визначення ключових метрик успішності. На цьому етапі необхідно провести ґрунтовний аналіз предметної області, визначити основні фактори впливу на успішність студентів та сформулювати математичну модель оцінювання. Важливим аспектом є встановлення порогових значень для класифікації результатів навчання та визначення критеріїв успішного завершення курсу.

Наступним критичним етапом є збір та попередня обробка даних. Цей процес включає формування репрезентативної вибірки даних про студентів, що містить демографічні показники, освітній бекграунд, результати пробного тестування та проміжних оцінювань, а також показники активності в навчальній системі. Особлива увага приділяється забезпеченню якості даних через застосування методів нормалізації, стандартизації та усунення викидів. Важливим аспектом є також дотримання принципів анонімізації персональних даних відповідно до чинного законодавства.

Процес відбору значущих ознак для моделі машинного навчання базується на застосуванні методів функціональної інженерії та статистичного аналізу. Проводиться кореляційний аналіз для виявлення взаємозалежностей між різними характеристиками, застосовуються методи зменшення розмірності даних та створюються композитні ознаки, що відображають комплексні характеристики навчального процесу.

Розробка та навчання предиктивної моделі є ключовим етапом створення системи. На основі аналізу існуючих алгоритмів машинного навчання обирається оптимальний підхід до вирішення задачі класифікації. Проводиться експериментальне порівняння різних моделей, включаючи метод k найближчих сусідів, методи на основі дерев рішень, методи аналізу виживання та нейронні мережі. Особлива увага приділяється валідації моделей та оцінці їх узагальнюючої здатності.

Архітектурне рішення десктопного додатку базується на принципах модульності та масштабованості. Графічний інтерфейс розробляється з урахуванням вимог ергономіки та користувацького досвіду, забезпечуючи інтуїтивно зрозумілу взаємодію з системою. Реалізується підтримка різних форматів вхідних даних та можливість візуалізації результатів прогнозування[10].

Інтеграція навченої моделі в програмний продукт вимагає розробки ефективних механізмів серіалізації та десеріалізації моделі, оптимізації обчислювальних процесів та забезпечення стабільності роботи системи при

різних вхідних даних. Особлива увага приділяється створенню механізмів логування та моніторингу роботи системи.

Процес тестування та валідації системи включає модульне тестування окремих компонентів, інтеграційне тестування та оцінку точності прогнозування на незалежних наборах даних. Проводиться аналіз стабільності роботи системи при різних сценаріях використання та оцінка обчислювальної ефективності.

Етап розгортання системи передбачає створення документації, розробку інструкцій для користувачів та адміністраторів, а також планування процесів технічної підтримки та оновлення системи. Важливим аспектом є забезпечення захисту даних та відповідності системи вимогам інформаційної безпеки.

Особлива увага приділяється етичним аспектам використання системи прогнозування. Розробляються механізми забезпечення прозорості алгоритмів прийняття рішень, впроваджуються засоби захисту персональних даних та створюються протоколи етичного використання системи в освітньому процесі.

1.3 Аналіз існуючих способів вирішення задачі

Дослідження успішності студентів та прогнозування їхньої успішності є комплексною проблемою, яка привертає значну увагу наукової спільноти. У сучасній науковій літературі представлено широкий спектр підходів та аналітичних інструментів для вирішення цієї задачі. Серед основних методів аналізу та прогнозування можна виділити наступні:

- Класичні нейронні мережі:
    - Багатошарові перцептрони, що здатні виявляти складні нелінійні залежності у даних.
    - Глибокі нейронні мережі, які дозволяють автоматично виділяти високорівневі ознаки та шаблони у великих наборах даних.
    - Згорткові нейронні мережі для обробки послідовних даних про успішність.
- Рекурентні нейронні мережі:

- LSTM (Long Short-Term Memory) архітектури для аналізу довготривалих залежностей.
- GRU (Gated Recurrent Units), що використовуються для швидкої роботи з часовими рядами успішності.

- Методи аналізу виживання:
  - Модель пропорційних ризиків Кокса для оцінки ймовірності відсіву.
  - Параметричні моделі виживання для прогнозування тривалості навчання.
  - Непараметричні методи оцінки функції виживання.

Важливо відзначити, що ефективність застосування будь-якого з вищезазначених методів суттєво залежить від якості вхідних даних. Численні дослідження підкреслюють критичну роль попередньої обробки даних, включаючи очищення, нормалізацію та валідацію. Якість та репрезентативність навчальної вибірки, а також повнота та достовірність зібраних даних є ключовими факторами, що визначають точність та надійність прогностичних моделей.

Окрім того, сучасні дослідження все частіше звертаються до гібридних підходів, які комбінують різні методи аналізу для досягнення оптимальних результатів. Такі комбіновані моделі дозволяють компенсувати обмеження окремих методів та підвищити загальну точність прогнозування.

1.3.1. Класичні нейронні мережі

У контексті сучасної освітньої аналітики багатошарові перцептрони MLP демонструють виняткову ефективність у вирішенні комплексних задач прогнозування академічної успішності. Їх застосування охоплює широкий спектр завдань: від прогнозування ймовірності успішного завершення освітньої програми до детальної стратифікації студентських груп за рівнем академічного ризику та формування предиктивних моделей підсумкового оцінювання. Особлива цінність MLP полягає у їхній здатності ефективно опрацьовувати

багатовимірні масиви структурованих даних, включаючи результати проміжного та підсумкового контролю, метрики залученості до навчального процесу, показники активності в системах дистанційного навчання та параметри взаємодії з освітнім контентом[11].

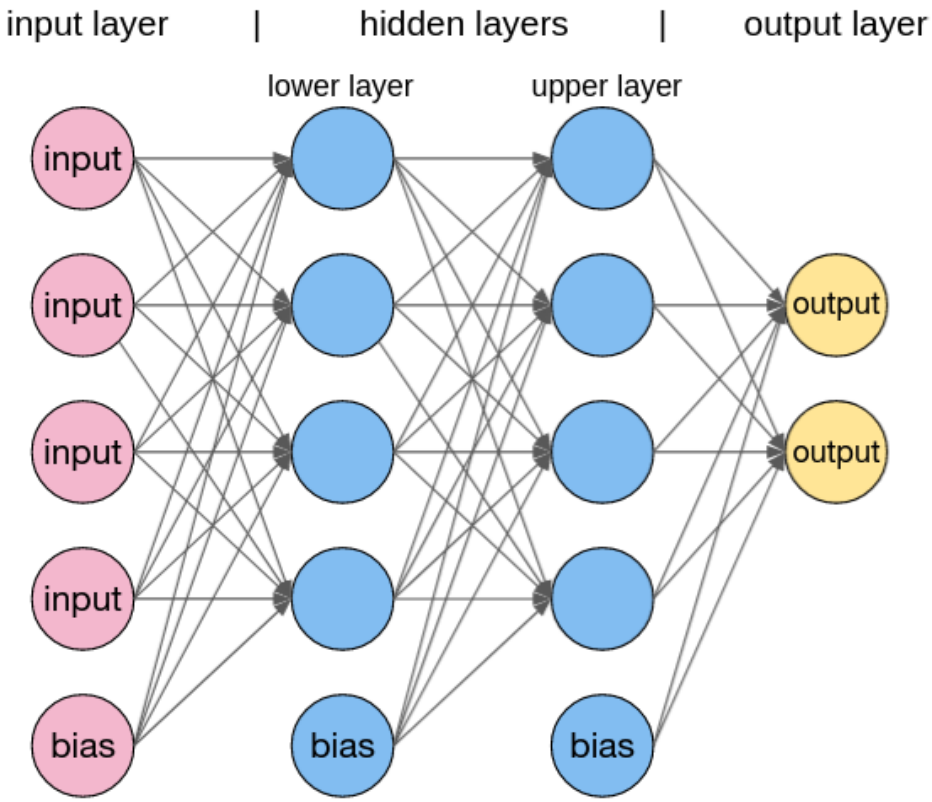


Рисунок 1.2. Приклад архітектури MLP моделі

Унікальною властивістю багатошарових перцептронів є їхня спроможність до виявлення складних взаємозалежностей між множинними параметрами навчального процесу, які залишаються непомітними при застосуванні традиційних методів аналізу. Наприклад, інтегральний аналіз таких показників як результативність проходження окремих завдань, регулярність відвідування занять та своєчасність виконання практичних завдань дозволяє з високою точністю ідентифікувати студентів, що знаходяться у зоні академічного ризику. Більше того, MLP здатні враховувати темпоральні аспекти навчальної активності, що уможливлює раннє виявлення потенційних проблем та своєчасне впровадження коригувальних заходів[11].

Імплементація багатошарових перцептронів також дозволяє здійснювати динамічну адаптацію навчального процесу, базуючись на індивідуальних особливостях засвоєння матеріалу кожним студентом. Це створює передумови для формування персоналізованих освітніх траєкторій та оптимізації розподілу навчального навантаження.

Глибокі нейронні мережі DNN представляють собою передовий інструментарій для комплексного аналізу освітніх даних значного обсягу. Їх архітектура оптимально пристосована для обробки різноманітних типів інформації, як для структурованих логів активності студентів, так і для неструктурованих текстових матеріалів чи мультимедійного контенту. Ключовою перевагою DNN є їхня здатність до автоматичного виділення високорівневих абстракцій та латентних факторів, що визначають успішність навчального процесу. Це суттєво зменшує необхідність у ручній функціональній інженерії, що традиційно вважається одним з найбільш довгих та трудозатратних етапів аналізу даних. В своїй архітектурі DNN візуально мало чим відрізняються від MLP. Основна відмінність полягає в тому, що приховані шари можуть бути різного типу.

У сфері предиктивної аналітики глибокі нейронні мережі демонструють виняткову ефективність при роботі з лонгітюдними освітніми даними. Вони здатні виявляти складні патерни поведінки та їх кореляцію з академічними результатами на основі аналізу часових рядів активності. Більше того, DNN здатні адаптивно корегувати свої прогностичні моделі на основі нових даних, що забезпечує стабільно високу точність передбачень у динамічному освітньому середовищі.

Згорткові нейронні мережі CNN представляють собою потужний інструмент для аналізу часових аспектів освітнього процесу. Їх архітектура оптимально пристосована для виявлення локальних та глобальних шаблонів у послідовностях освітніх даних, таких як щоденна активність студентів, динаміка виконання завдань та показники залученості до навчального процесу. CNN демонструють особливу ефективність при аналізі довготривалих трендів

та циклічних шаблонів у навчальній поведінці, що дозволяє ідентифікувати критичні моменти в траєкторії навчання кожного студента.

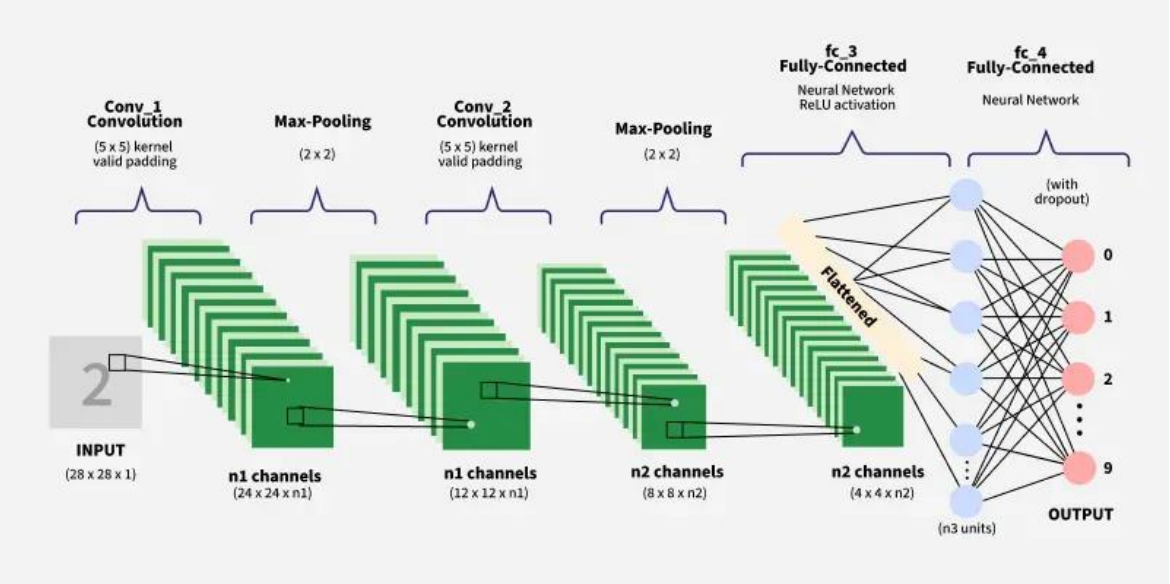


Рисунок 1.3. Приклад архітектури CNN моделі

У контексті прогностичної аналітики CNN забезпечують високоточну ідентифікацію предикторів академічної неуспішності на основі аналізу поведінкових маркерів. Це створює можливість для впровадження проактивних стратегій, включаючи автоматизовані системи раннього попередження, персоналізовані мотиваційні комунікації та адаптивні механізми підтримки навчального процесу. Особлива цінність CNN полягає у їхній здатності до виявлення складних взаємозв'язків між різними аспектами освітньої активності, що дозволяє формувати комплексні стратегії підтримки студентів, спрямовані на оптимізацію результатів навчання та мінімізацію ризику відсіву.

Класичні нейронні мережі, зокрема багатошарові перцептрони MLP, довели свою ефективність у задачах аналізу успішності студентів. Вони здатні моделювати складні нелінійні залежності між різноманітними освітніми та поведінковими факторами, що дозволяє точно прогнозувати результати навчання. Дослідження показують, що MLP можуть перевершувати інші алгоритми машинного навчання, такі як дерева рішень чи метод опорних векторів, у точності передбачення оцінок студентів.

Проте, ефективність MLP може знижуватися при обробці великих або неструктурованих даних, таких як текстові відповіді чи часові ряди активності студентів. У таких випадках глибші або спеціалізовані архітектури, наприклад, згорткові чи рекурентні нейронні мережі, можуть забезпечити кращі результати.

1.3.2. Рекурентні нейронні мережі

Рекурентні нейронні мережі RNN є потужним інструментом для аналізу послідовних даних, що робить їх особливо корисними у сфері освітньої аналітики. Завдяки своїй архітектурі, яка включає зворотні зв'язки, RNN здатні враховувати часові залежності та контекст попередніх подій при обробці поточних даних. Це дозволяє моделювати динаміку навчального процесу студента, враховуючи його попередні дії та результати[12].

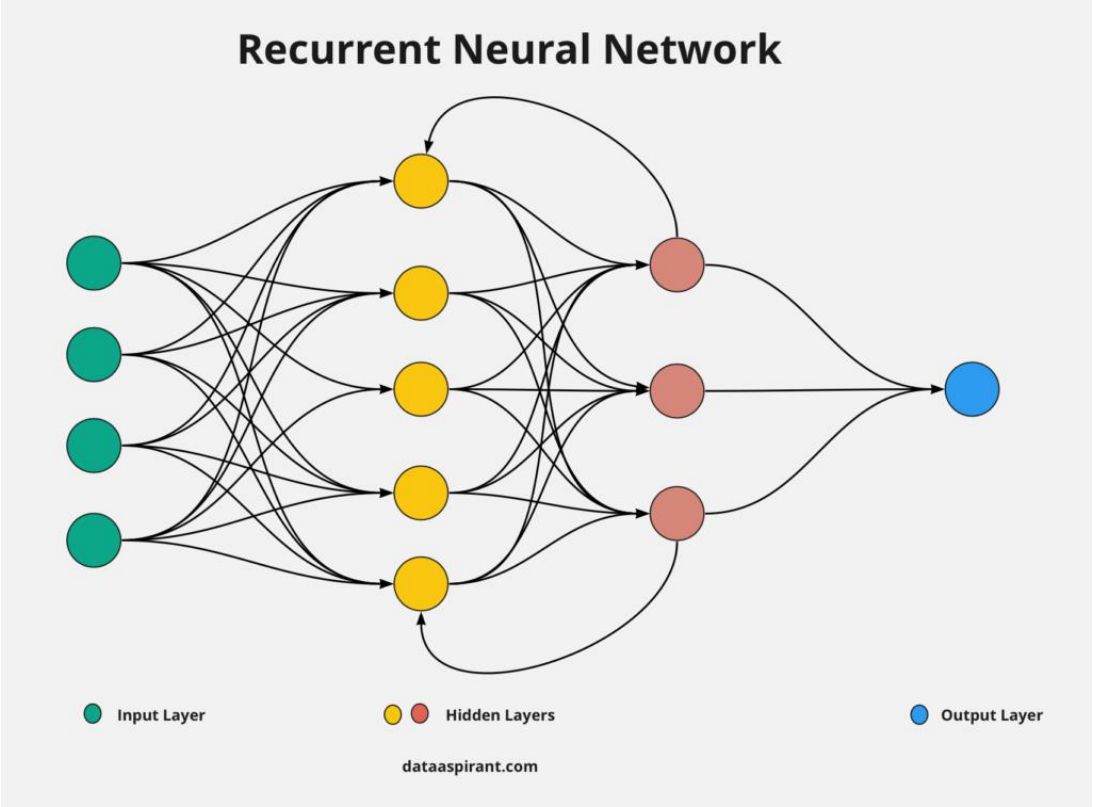


Рисунок 1.4. Приклад архітектури RNN моделі

У практичних застосуваннях RNN використовуються для прогнозування майбутньої успішності студентів на основі їхньої історії навчання. Наприклад, дослідження показали, що моделі на основі RNN можуть ефективно передбачати результати студентів, дозволяючи виявляти тих, хто знаходиться в зоні ризику, ще до завершення курсу. Це дає можливість викладачам своєчасно втручатися та надавати необхідну підтримку.

LSTM (Long Short-Term Memory) – це різновид RNN, який розроблено для подолання проблеми зникнення або вибуху градієнтів при обробці довгих послідовностей. Архітектура LSTM дозволяє зберігати інформацію на тривалих відрізках часу, що особливо важливо при моделюванні навчального шляху студента протягом всього курсу або навіть кількох курсів. У задачах прогнозування LSTM демонструють високу точність та стабільність, особливо коли враховується не лише остання активність студента, а й ранні прояви труднощів у навчанні.

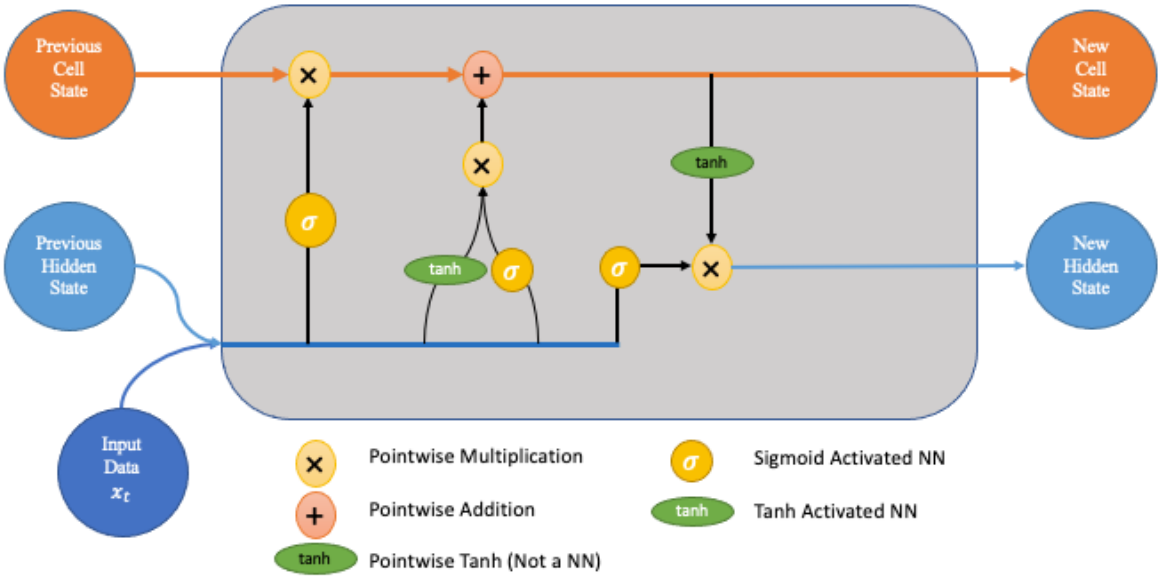


Рисунок 1.5. Приклад архітектури LSTM моделі

GRU (Gated Recurrent Units) є ще одною оптимізованою версією RNN, яка поєднує в собі переваги LSTM, але має спрощену структуру і меншу кількість параметрів. GRU чудово підходять для задач, де потрібно швидко обробляти часові ряди успішності при обмежених обчислювальних ресурсах. У контексті

аналізу освітніх даних GRU забезпечують конкурентну якість прогнозування при знижених витратах на навчання моделі, що робить їх придатними для впровадження у реальні системи моніторингу навчального прогресу.

Крім того, поєднання RNN з механізмами уваги та методами дистиляції знань дозволяє покращити точність прогнозів та зменшити обчислювальні витрати, що є важливим для реального впровадження таких систем у навчальні платформи. Такі моделі не лише забезпечують високі показники точності, але й дозволяють інтерпретувати результати, що є важливим для прийняття обґрунтованих рішень у сфері освіти.

Загалом, RNN, зокрема їхні вдосконалені архітектури LSTM і GRU, демонструють високу ефективність у задачах аналізу успішності студентів, особливо коли йдеться про обробку послідовних та часових даних. Їхнє застосування сприяє більш глибокому розумінню навчального процесу та дозволяє впроваджувати проактивні стратегії підтримки студентів.

1.3.3. Методи аналізу виживання

Методи аналізу виживання представляють собою потужний інструментарій для дослідження часових характеристик певних подій, що особливо актуально у контексті прогнозування відсіву студентів з навчальних програм. Ці методи дозволяють не лише оцінити ймовірність «виживання», тобто, продовження навчання, студента на певному етапі, але й виявити критичні періоди, коли ризик припинення навчання є найвищим[13].

Ключовим методом у цій області є модель пропорційних ризиків Кокса, яка дозволяє оцінити вплив різних факторів на ймовірність відсіву студента, враховуючи часову складову. Особливість даної моделі полягає в її напівпараметричній природі, що звільняє її від обмежень конкретного статистичного розподілу, зберігаючи при цьому фундаментальне припущення про пропорційність ризиків.

Актуальні дослідження демонструють практичне застосування моделі Кокса для аналізу впливу академічної успішності та соціально-економічних

факторів на відрахування студентів. Результати свідчать про значний кумулятивний ефект низької успішності, недостатньої залученості до навчального процесу та відсутності належної педагогічної підтримки на ризик припинення навчання.

Методологічна структура моделі Кокса базується на двох фундаментальних аспектах: часовому вимірі, що визначає момент настання досліджуваної події, та факторах впливу, які кількісно характеризують зміну ризику. При цьому модель передбачає константність впливу досліджуваних змінних протягом усього періоду спостереження.

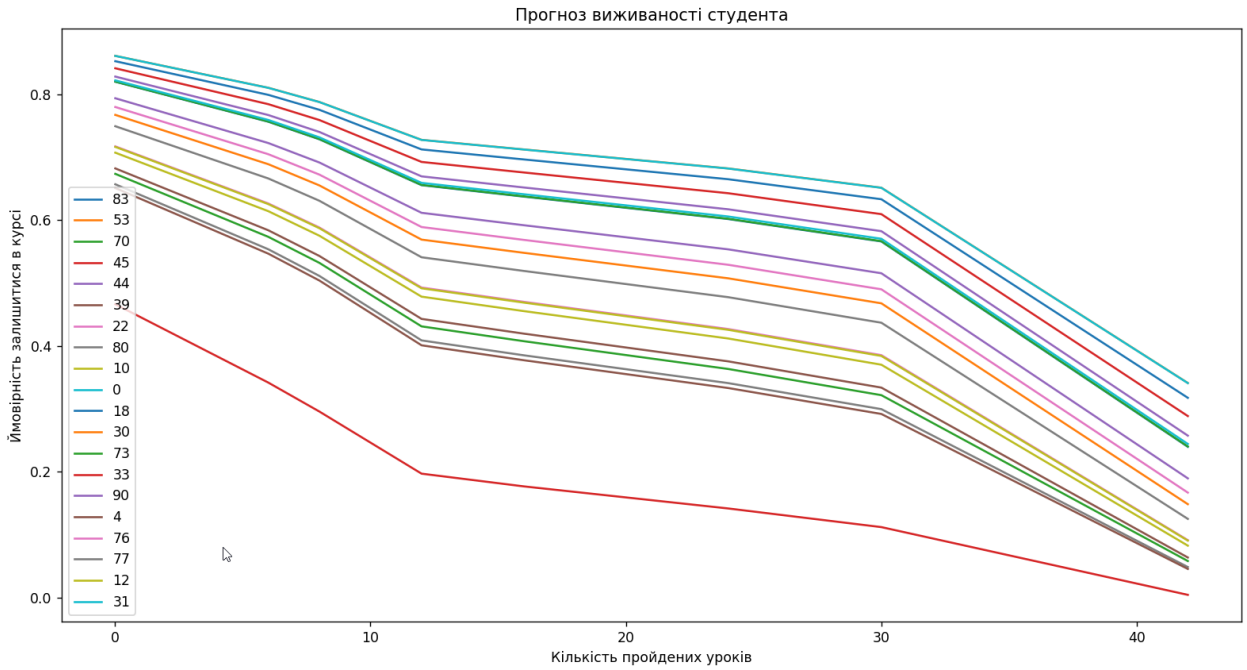


Рисунок 1.6. Приклад результату роботи моделі Кокса

Серед переваг даного методу варто відзначити його універсальність щодо типу розподілу даних, інтерпретаційну прозорість коефіцієнтів впливу та здатність працювати з неповними наборами даних. Водночас, необхідно враховувати обмеження методу, зокрема, залежність від гіпотези пропорційності ризиків та потенційні труднощі інтерпретації при надмірній кількості змінних.

Альтернативним підходом є метод Каплана-Мейєр, який забезпечує візуалізацію функції виживання та дозволяє оцінити ймовірність продовження

навчання у часовій перспективі. У контексті освітньої аналітики цей метод дає можливість ідентифікувати критичні періоди потенційного відрахування студентів.

Методологія Каплана-Мейєр концентрується на аналізі двох ключових аспектів: часу настання досліджуваної події та випадків, коли подія ще не відбулася. Особлива цінність методу полягає у можливості порівняльного аналізу різних груп студентів, наприклад, за рівнем академічної успішності, що дозволяє формувати детальні аналітичні профілі без необхідності специфікації статистичного розподілу.

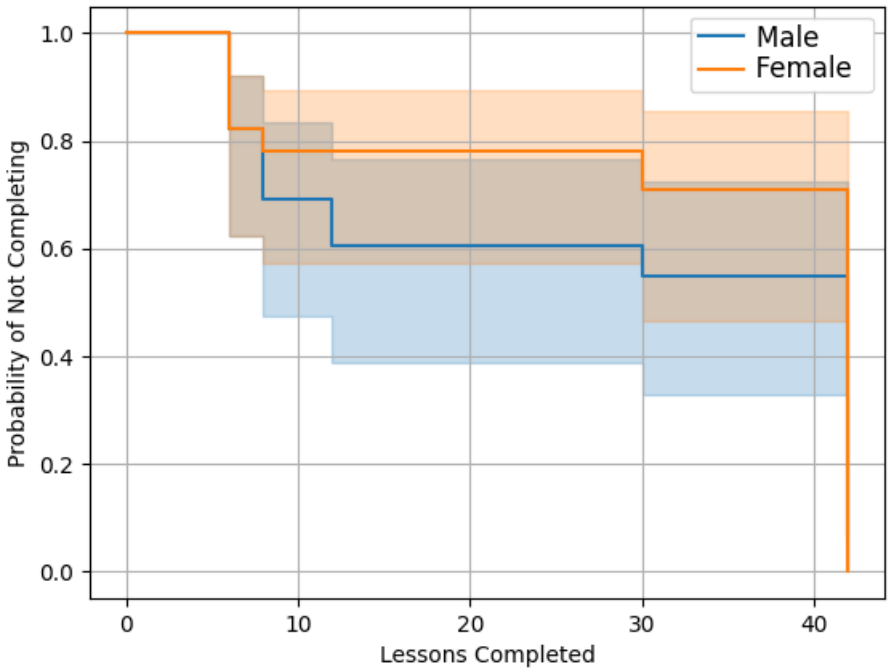


Рисунок 1.7. Приклад результату роботи кривих Каплана-Мейєр

Емпіричні дослідження підтверджують ефективність методу Каплана-Мейєр як діагностичного інструменту в освітній аналітиці. Дослідники успішно застосували цей метод для аналізу траєкторій навчання та ідентифікації критичних моментів, що вимагають педагогічного втручання. Ключовою перевагою методу є його здатність інтегрувати неповні дані та незалежність від припущень щодо розподілу, що робить його надійним інструментом аналізу освітніх процесів.

Багаторівневі моделі виживання дозволяють враховувати ієрархічну структуру освітніх даних, де студенти можуть бути згруповані за різними ознаками, наприклад, за навчальними групами, викладачами або освітніми програмами. Такий підхід забезпечує більш точну оцінку впливу факторів різного рівня на ймовірність завершення навчального курсу та дозволяє виявити контекстуальні ефекти, що впливають на успішність студентів.

1.4 Визначення вхідних даних для вирішення поставленої задачі

Як було зазначено в даному розділі вище, якість та повнота зібраних даних є критичним фактором, який безпосередньо впливає на точність прогнозування та ефективність освітньої аналітики. Для забезпечення комплексного та систематичного збору релевантних даних, планується здійснити суттєву модифікацію існуючої навчальної платформи. Ця модифікація передбачає впровадження розширеної системи моніторингу наступних ключових метрик:

- Часові показники виконання завдань (включаючи деталізацію за типами завдань та рівнями складності).
- Кількісні параметри ітерацій при вирішенні завдань, що дозволяють оцінити рівень наполегливості та здатність до самокорекції.
- Статус завершення завдань в рамках синхронного навчання (під час уроку), з урахуванням контекстуальних факторів.
- Комплексні показники взаємодії з домашніми завданнями, включаючи:
    - Частоту спроб виконання.
    - Успішність завершення.

Поточна система збору даних вже включає наступні важливі категорії інформації:

- Комплексні демографічні та контекстуальні дані студента:
    - Вікові характеристики.
    - Канали залучення до освітньої програми.
    - Обрана модальність навчання.

    - Соціологічні та демографічні показники.
- Якісні показники залученості:
    - Структурований зворотний зв'язок від студентів.
    - Метрики мотивації та зацікавленості.
    - Показники активності участі в освітньому процесі.
- Діагностичні метрики початкового рівня:
    - Результати виконання пробних завдань.
    - Оцінка базових компетенцій.
    - Показники готовності до навчання.

Також, до даних студента буде внесено регулярні педагогічні оцінки на щотижневій основі, що забезпечить можливість відстеження динаміки прогресу студента та ефективності освітнього процесу з точки зору вчителя.

Першим етапом в створенні програмного продукту буде пробний додаток, що дозволить проводити базову перевірку студента після пробного заняття. Тобто, надасть можливість внести інформацію про студента, щоб виконати аналіз та переглянути, чи є він ризиковим після першого пробного заняття.

Другим етапом роботи над програмою буде розширення роботи зі студентами до поняття груп. Основною ціллю другого етапу буде введення поняття «група студентів» і, власне, аналізу студента в групі. Також, буде додано аналіз того, чи пройде студент курс до кінця, якщо його буде додано в ту чи іншу групу.

Третім, і останнім етапом роботи буде введення останніх, необхідних, функціональних можливостей та робота над покращенням користувацького досвіду менеджера школи, що буде взаємодіяти з програмою.

Основною платформою, для якої буде розроблено додаток є операційна система Windows.

1.5 Висновки до першого розділу

У результаті проведеного аналізу було визначено, що проблема передчасного припинення навчання студентами програмування є актуальною як

для приватних, так і для державних освітніх закладів. Сучасні підходи до підготовки ІТ-фахівців потребують впровадження інноваційних методів прогнозування освітньої успішності, які дозволяють ефективно ідентифікувати як студентів із високим потенціалом, так і тих, хто знаходиться в групі ризику. Розглянуто дві основні бізнес-моделі взаємодії з абітурієнтами – модель утримання та модель селективного відбору, кожна з яких має специфічні завдання, інструменти реалізації та критерії ефективності.

Проблема відбору та утримання студентів в освітніх закладах, що навчають програмуванню є активно досліджуваною по всьому світу. Наразі, в рішенні цієї проблеми активно використовуються та досліджуються інформаційні системи з використанням як класичних так і більш просунутих нейронних мереж та методів аналізу виживання.

У межах першого розділу було чітко сформульовано мету та задачі дослідження, обґрунтовано доцільність створення програмного застосунку як основного інструменту реалізації прогнозної технології, а також окреслено ключові функціональні вимоги до майбутньої системи. Окрему увагу приділено аналізу вже існуючих підходів до прогнозування академічної успішності, що дозволило виявити їх недоліки та визначити напрями для вдосконалення. Було сформовано основне бачення продукту та його окреслено завдання для його створення.

РОЗДІЛ 2. АНАЛІЗ МЕТОДІВ ВИРІШЕННЯ ЗАДАЧІ

2.1 Методи вирішення задачі

Кроки реалізації програмного продукту можуть бути розділені на два типи: ті, що розвивають функціональність продукту і ті, що покращують використання продукту користувачем.

Спочатку, розглянемо функціональність продукту. В процесі визначення можна виділити три основних етапи:

1. Виявлення студентів, що запишуться на курс після пробного заняття.
2. Виявлення студентів, що можуть завершити навчання на курсі.
3. Виявлення найкращих студентів для проходження курсу програмування на основі попередніх досягнень учня.

Кожен етап містить під собою різні задачі та може використовувати різні інструменти. Через це, було прийнято рішення розділити дані функціональності на три окремих розділи та зробити їх незалежними між собою.

2.1.1 Вирішення задачі виявлення студентів, що запишуться на курс після пробного заняття

Дана задача є найбільш простою і класичною з усіх трьох. В даному контексті таку задачу можна розглядати як класичну задачу класифікації, оскільки бажаним результатом є значення того, запишеться студент на курс, чи ні, а не питання того, скільки занять даний студент відвідає.

Для вирішення такої задачі буде реалізовано два класичних методи: k найближчих середніх та випадкові ліси.

Метод K-найближчих сусідів (KNN) – це класичний алгоритм машинного навчання, що часто використовується в випадках, коли необхідно вирішити задачу класифікації чи регресії. Алгоритм працює на основі принципу схожості. Для класифікації нового об’єкта, в нашому випадку – студента, алгоритм шукає *K* найближчих до нього об’єктів у навчальній вибірці та визначає його категорію на основі їхніх значень. У контексті дослідження успішності

студентів KNN може бути використаний для розділення студентів на групи з подібними характеристиками, наприклад, на тих, хто має високий, середній або низький ризик відсіву[14].

Вихідні дані методу K-найближчих сусідів:

1. $X = X_1, X_2, \ldots, X_p$ – вектор ознак для кожного індивіда.
2. Y – мітка класу, тобто, результат роботи моделі.

Для знаходження найближчих сусідів використовується метрика відстані, наприклад, евклідова відстань:

$$d(x, y) = \sqrt{\sum_{i=1}^{p} (x_i - y_i)^2}$$

де x та y – вектори ознак для двох індивідів.

Параметр K визначає кількість найближчих сусідів, які враховуються для класифікації. Вибір K впливає на точність моделі. Мале значення K може призводити до перетренування, але велике значення – до недостатнього навчання.

Для нового об'єкта x_new визначаються K найближчих сусідів з навчальної вибірки. Клас x_new визначається як найпоширеніший клас серед цих сусідів або як середнє значення їхніх міток.

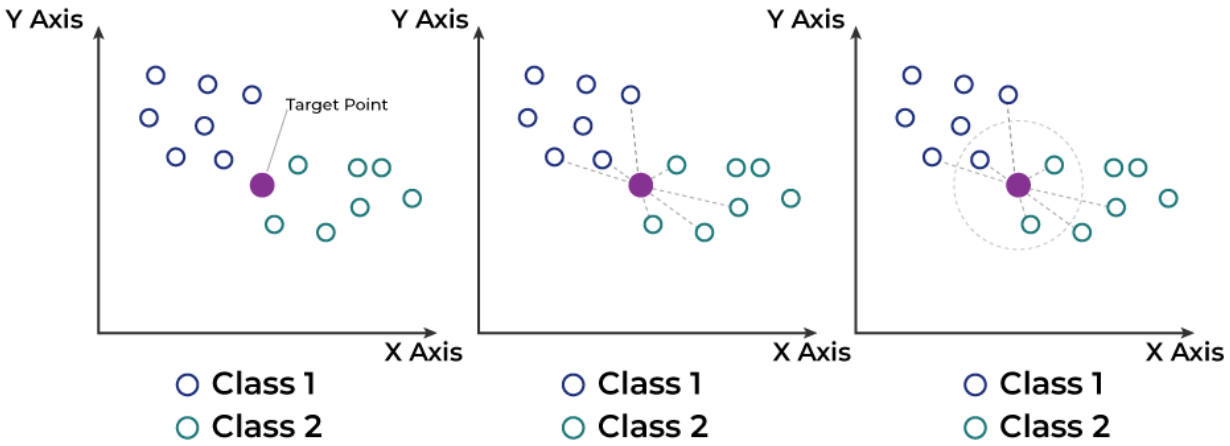


Рисунок 2.1. Приклад класифікації за KNN

Сучасні алгоритми аналізу даних часто базуються на концепції колективного інтелекту, де множина простих моделей об’єднується для досягнення вищої точності. Серед них вирізняються випадкові ліси, ансамблевий метод, що перетворює звичайні дерева рішень на потужний прогностичний механізм.

На відміну від єдиного дерева, яке може схилятися до перенавчання, випадкові ліси будують сотні чи тисячі дерев, кожне з яких тренується на унікальній підвибірці даних. Ця техніка, відома як багінг, запобігає надмірній залежності від окремих спостережень[15]. Але справжня суть методу криється у подвійній випадковості:

1. Бутстрепна вибірка – кожне дерево отримує свій набір точок даних, відібраних із заміною.
2. Обмежений набір ознак – при розщепленні вузлів алгоритм обирає лише випадкову підмножину характеристик, уникаючи домінування однієї чи кількох впливових змінних.

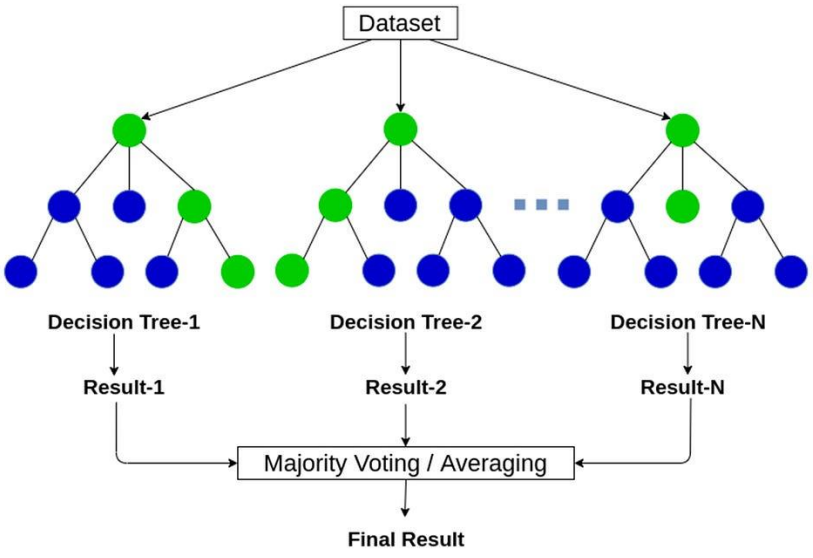


Рисунок 2.2. Відображення роботи методу випадкових лісів

Ключова перевага алгоритму випадкових лісів полягає в його здатності до одночасної мінімізації зсуву та дисперсії помилки за рахунок ансамблевої природи. Механізм бутстреп агрегування ефективно знижує кореляцію між

окремими деревами, при цьому випадковий вибір ознак на кожному кроці розщеплення забезпечує додаткову диверсифікацію моделей. Такий підхід демонструє особливу стійкість до ефекту перенавчання навіть у випадках високої розмірності ознакового простору. Важливо відзначити, що метод забезпечує вбудовану оцінку важливості предикторів через аналіз середнього зменшення дисперсії, що робить його корисним інструментом для відбору ознак.

Незважаючи на переваги, метод має суттєві обчислювальні витрати, які зростають лінійно з кількістю дерев. У випадках великих датасетів це може призводити до необхідності використання розподілених обчислень.

Іншим критичним аспектом є втрата інтерпретованості – якщо окремі дерева рішень допускають візуалізацію та аналіз логічних правил, то рішення, отримане з тисячі дерев стає практично чорною скринькою.

2.1.2 Вирішення задачі утримання студента на курсі програмування

Дана задача є однією з найскладніших з усіх трьох. Основна її задача – дізнатись, чи завершить студент навчання на відповідному курсі у відповідній групі і якщо не завершить, через скільки уроків він достроково завершить навчання на курсі.

Для реалізації цього завдання будуть використані два методи. Метод Каплан-Мейєр для базового дослідження набору даних та метод Кокса для кінцевого використання в програмному застосунку.

Метод Каплан-Мейєр – це непараметричний метод, який використовується для оцінки функції виживання $S(t)$, тобто ймовірності того, що подія (наприклад, відсів студентів) не відбудеться до моменту часу t. Цей метод є одним із найпоширеніших у аналізі виживання через свою простоту та здатність працювати з цензурованими даними (коли подія ще не відбулася до кінця спостереження)[16].

Функція виживання $S(t)$ оцінюється як добуток ймовірностей того, що подія не відбудеться в кожен момент часу t_i, коли відбувається подія. Математично це виражається як:

$$S(t) = \prod_{i:t_i \leq t} (1 - \frac{d_i}{n_i})$$

де:

- t_i – моменти часу, коли відбуваються події;
- d_i – кількість подій у момент часу t_i;
- n_i – кількість індивідів, які знаходяться під ризиком на момент часу t_i.

Для порівняння кривих виживання для різних груп використовується лог-ранговий тест. Нульова гіпотеза полягає в тому, що функції виживання для двох груп однакові. Статистика тесту розраховується як:

$$X^2 = \sum_{i=1}^{m} \frac{(O_{1i} - E_{1i})^2}{E_{1i}}$$

де:

- O_{1i} – кількість подій у першій групі в момент часу t_i;
- E_{1i} – очікувана кількість подій у першій групі в момент часу t_i, яка розраховується за формулою:

$$E_{1i} = \frac{n_{1i} - d_i}{n_i}$$

де n_{1i} – кількість індивідів у першій групі, які знаходяться під ризиком на момент часу t_i.

Модель Кокса описує функцію ризику, яка визначає ймовірність настання події в момент часу t за умови, що подія ще не відбулася до цього моменту[17]. Математично функція ризику виражається як:

$$h(t, X) = h_0(t) \cdot exp(\beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p)$$

де:

- $h(t, X)$ – функція ризику в момент часу t для індивіда з коваріатами X;
- $h_0(t)$ – базова функція ризику, що залежить виключно від часу;
- $X_1, X_2, \ldots, X_p$ – коваріати (фактори, що впливають на ризик);

- $\beta_1, \beta_2, \dots, \beta_p$ – коефіцієнти, які оцінюють вплив коваріат на ризик.

Ключовим припущенням моделі Кокса є пропорційність ризиків: вплив коваріат на ризик є постійним у часі. Це означає, що співвідношення ризиків для двох індивідів залишається незмінним протягом усього часу спостереження. Формально це можна записати як:

$$\frac{h(t, X_1)}{h(t, X_2)} = \exp\left(\sum_{i=1}^{p} \beta_i (X_{1i} - X_{2i})\right)$$

Також, варто зазначити, що вплив коваріат на ризик є мультиплікативним, тобто кожен коефіцієнт β_i визначає, наскільки змінюється ризик при зміні коваріати X_i на одиницю.

Параметри моделі Кокса $\beta_1, \beta_2, \dots, \beta_p$ оцінюються за допомогою методу часткової правдоподібності. Функція часткової правдоподібності визначається як:

$$L(\beta) = \prod_{j=1}^{m} \frac{exp(\sum_{i=1}^{p} \beta_i X_{ij})}{\sum_{k \in R_j} exp(\sum_{i=1}^{p} \beta_i X_{ik})}$$

де:

- m – кількість унікальних моментів часу, коли відбуваються події;
- R_j – множина індивідів, які знаходяться під ризиком на момент часу t_j;
- X_{ij} – значення коваріати i для індивіда j.

Максимізація функції часткової правдоподібності дозволяє отримати оцінки коефіцієнтів β.

2.1.3 Вирішення задачі пошуку найкращих студентів для проходження курсу програмування

Дана задача є однією доволі типовою. Для її виконання необхідно використати метод ранжування RankNet. Ця технологія поєднує нейронні мережі та попарну оптимізацію для точного визначення релевантності об’єктів через систему оцінок[18].

RankNet аналізує взаємозв'язки між характеристиками об'єктів. В основі алгоритму лежить принцип зворотного поширення градієнта для мінімізації функції втрат на основі кросс-ентропії. Це дозволяє обчислювати ймовірнісні показники позиціонування документів в ієрархії.

Серед переваг можна виділити високу здатність адаптуватися до закономірностей у даних та стабільність процесу навчання, а серед недоліків слід зазначити значну складність обчислень та не пряме покращення метрик пошуку.

RankNet залишається важливим методом машинного ранжування, використання якого є досить простим і ефективним порівняно з іншими методами для вирішення подібних завдань.

Продемонструємо його суть за допомогою простого прикладу. Візьмемо пару об’єктів o_i та o_j, з наступними параметрами:

- $s_i = f(o_i; \theta)$ – оцінка, передбачена моделлю для o_i.
- $s_j = f(o_j; \theta)$ – оцінка, передбачена моделлю для o_j.

RankNet визначає ймовірність того, що o_i релевантніший за o_j за допомогою сигмоїдальної функції:

$$P_{ij} = \sigma(s_i - s_j) = \frac{1}{1 + e^{-(s_i - s_j)}}$$

де:

- $P_{ij} \in (0,1)$ - ймовірність, що o_i релевантніший за o_j.
- Якщо $s_i \gg s_j$, то $P_{ij} \approx 1$.
- Якщо $s_i \ll s_j$, то $P_{ij} \approx 0$.
- Якщо $s_i = s_j$, то $P_{ij} = 0.5$ (невизначеність).

2.2 Розробка проекту проведення аналітичної діяльності

Для реалізації проєкту необхідно обрати інструментарій для вирішення поставленої проблеми. Для повноцінної вирішення задачі необхідно обрати:

- Програмне середовище.
- Базу даних та систему управління нею.
- Основні бібліотеки для роботи з базою даних, реалізації вищеописаних методів роботи з даними.
- Бібліотеку для розробки інтерфейсу програмного застосунку.

2.2.1 Вибір програмного середовища

Для виконання даної роботи необхідно обрати таку мову програмування, що дозволить виконувати наступні задачі на комфортному рівні:

- Робота з нейронними мережами та іншими моделями. Даний пункт є найважливішим, оскільки якісна робота та налаштування моделей є доволі важливим фактором, що впливає на якість.
- Створення інтерфейсів. Найменш пріоритетний пункт з усіх, адже додаток розробляється для внутрішнього користування в компанії і, в майбутньому, планується бути заміненим через інтеграцію API з внутрішньою платформою. А в такому випадку, бачити інтерфейс додатку буде тільки системний адміністратор.
- Робота з базами даних. Має середню вагу, оскільки інтеграція з базами даних це хоч і важливий процес, проте він є вторичним і не впливає на кінцеву якість продукту.

З вимог, що були описані раніше, можна підібрати наступні мови програмування:

1. Python. Є найбільш популярною та широко використованою мовою програмування. І хоча Python, можливо, не є найкращим інструментом для роботи з нейронними мережами та моделями, проте він точно є одним з найкращих. Для роботи, в області Data Science, дана мова програмування має велику кількість бібліотек і фреймворків, в яких вже реалізовано більшість моделей на достатньо якісному рівні. Також, Python має непогані можливості в побудові інтерфейсів та

роботою з базами даних, що забезпечують високий рівень швидкодії та надійності.

2. R. Це мова програмування, яка спеціалізується на статистичному аналізі та обробці даних. Вона, безперечно, є одним із найкращих інструментів для аналітичних задач, де необхідна робота з моделями машинного навчання, зокрема зі статистичними методами, класифікацією, регресією, візуалізацією та іншими елементами Data Science. R має широку базу вже готових рішень у вигляді пакетів, які дозволяють працювати з різноманітними моделями без потреби у їх реалізації «з нуля». Проте, R має обмежені можливості у створенні десктопних інтерфейсів, а її взаємодія з базами даних хоч і можлива, але не настільки зручна й гнучка, як в інших мовах програмування.
3. Java. Це мова програмування, що характеризується високою продуктивністю, стабільністю та хорошою підтримкою об’єктно-орієнтованого підходу. Вона часто використовується для створення великих комерційних додатків та систем із розвиненим інтерфейсом. Java підтримує побудову десктопних інтерфейсів на високому рівні (через Swing або JavaFX), а також має широкий набір інструментів для взаємодії з базами даних. Що ж стосується роботи з моделями машинного навчання – Java тут дещо поступається Python і R, як за простотою, так і за кількістю відповідних бібліотек. І хоча існують фреймворки типу Weka або Deeplearning4j, вони не мають настільки широкої підтримки.
4. C#. Це потужна мова програмування, яка особливо добре підходить для створення десктопних додатків у середовищі Windows. Вона дозволяє реалізовувати зручні та функціональні графічні інтерфейси за допомогою Windows Forms або WPF, що робить її привабливою для побудови робочих інструментів для внутрішнього використання. Мова C# також оснащена потужними інструментами для підключення до баз даних, зокрема через ADO.NET або Entity Framework. Однак, при роботі з моделями машинного навчання можливості цієї мови

обмежені. Хоча існує бібліотека ML.NET, вона поки що не може конкурувати за гнучкістю та кількістю готових рішень із Python чи R.

Отже, з огляду на пріоритетність поставлених для мови програмування задач доцільно обрати Python або R.

І хоча R є лідером в роботі з даними, Python не має значного відставання. В тому числі, Python може запропонувати зручну роботу з інтерфейсами, що, в цілому, недоступно в R. Розглянемо дану мову програмування детальніше.

Python є універсальною мовою програмування з динамічною типізацією, яка відрізняється своїм елегантним та зрозумілим синтаксисом. Ця мова стала справжнім проривом у світі програмування, особливо в галузях машинного навчання та обробки великих даних, завдяки своїй гнучкості та потужному функціоналу[19].

Архітектура Python базується на декількох ключових концепціях, включаючи об'єктно-орієнтоване, узагальнене та функціональне програмування. Особливу увагу варто приділити модульній системі, яка дозволяє структурувати код у логічні блоки та пакети, що значно полегшує розробку та підтримку великих проектів. Мова також пропонує потужні інструменти для інтроспекції, що дає можливість аналізувати структуру об'єктів під час виконання програми.

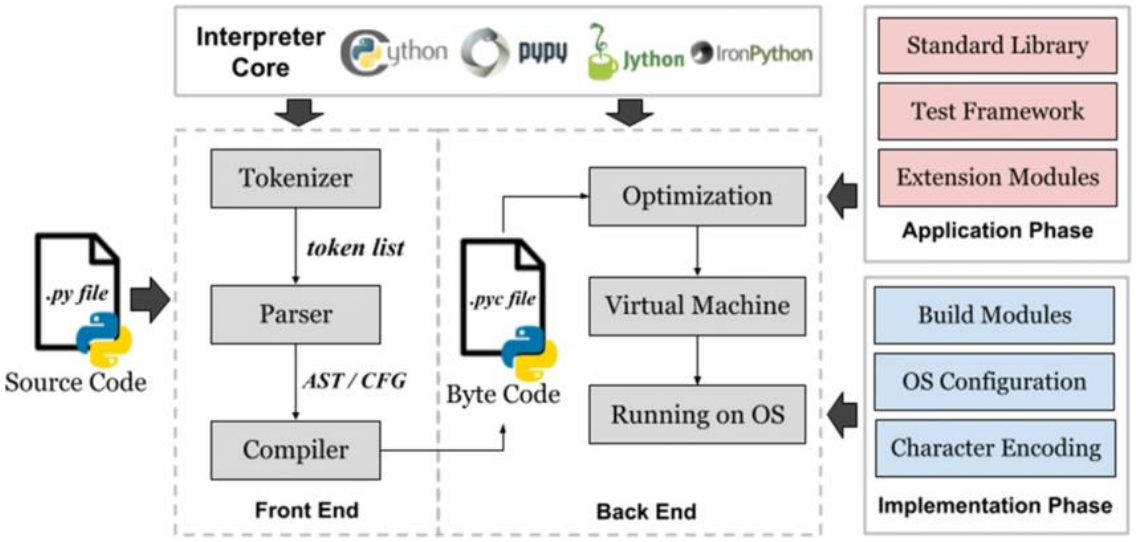


Рисунок 2.3. Архітектура мови Python

Мова програмування Python є універсальним інструментом, що знаходить широке застосування у різноманітних сферах. Однією з головних галузей використання Python є Data Science, де ця мова стала одним з стандартів завдяки великій кількості спеціалізованих бібліотек (таких як «NumPy», «Pandas», «Scikit-learn», «TensorFlow», «Keras», тощо) і активній спільноті, що постійно розширює екосистему. Python також є популярним у веб-розробці завдяки фреймворкам Django і Flask, в автоматизації бізнес-процесів, розробці скриптів, створенні програмних застосунків, тестуванні програмного забезпечення, роботі з API та розробці штучного інтелекту. Завдяки своїй читабельності та простоті синтаксису, Python активно використовується також у сфері освіти для навчання основам програмування та алгоритмів.

Серед переваг Python слід відзначити його низький поріг входу, що робить його зручним для початківців, а також гнучкість, завдяки якій можна швидко реалізовувати ідеї, експериментувати з різними моделями та методами. Важливою перевагою є також кросплатформеність. Python можна запускати на різних операційних системах без суттєвих змін у коді. Крім того, для більшості задач уже існують готові модулі або розширення, що значно прискорює розробку.

Проте, незважаючи на всі переваги, Python має і свої недоліки. Найбільш суттєвим серед них є низька швидкодія порівняно з мовами нижчого рівня, такими як C++ чи Java, що може стати проблемою в задачах, де потрібна обробка великої кількості даних у реальному часі або висока продуктивність. Також, певні труднощі можуть виникнути при створенні складних інтерфейсів або при розгортанні програм у середовищах, де потрібен повний контроль над ресурсами системи. Тим не менш, завдяки своїй гнучкості та потужній екосистемі, Python часто стає першим вибором у задачах, пов’язаних із аналізом даних, моделюванням та автоматизацією.

2.2.2 Система управління базами даних

Система управління базами даних (СУБД) – це програмне забезпечення, що забезпечує зручний, ефективний та безпечний спосіб взаємодії з базами даних. Вона дозволяє створювати, зберігати, редагувати, видаляти та шукати інформацію у структурованому вигляді, при цьому контролюючи доступ, забезпечуючи цілісність даних та підтримуючи одночасну роботу кількох користувачів. Іншими словами, СУБД – це посередник між користувачем або додатком і самими даними, що забезпечує правильну обробку запитів, керує структурою бази даних і гарантує, що вся інформація зберігається коректно та не втрачається у разі збоїв[20].

СУБД відіграють ключову роль у будь-якому проєкті, де необхідно працювати з великими обсягами даних. Особливо це актуально у випадку автоматизованих систем аналітики, звітності або прогнозування, де дані надходять регулярно і мають бути оброблені у чітко визначеному форматі. В сучасних умовах більшість СУБД підтримують мову SQL, яка є стандартом для взаємодії з реляційними базами даних.

Типи СУБД розрізняються залежно від способу організації, зберігання та обробки даних. Найпоширенішими сьогодні є реляційні СУБД, які будують структуру даних у вигляді таблиць з чітко визначеними зв’язками між ними. Такий підхід зручний для роботи з чітко структурованою інформацією, де важливо дотримуватись логічних зв’язків між сутностями. Через що, реляційні СУБД, такі як, PostgreSQL або MySQL активно застосовуються в корпоративних системах, банківській сфері, CRM-системах, бухгалтерських програмах та аналітичних інструментах.

Для зв’язку таблиць використовуються ключі. В цілому, варто виділити два типи ключів, первинний і зовнішній. Первинний ключ – це поле, що являється унікальним ідентифікатором кожного запису. Первинний ключ повинен завжди бути унікальним і ніколи не повторюватись, наприклад, це може бути номер паспорту або звичайне число, що буде збільшено на 1 для кожного наступного рядку таблиці.

Рисунок 2.4. Реляційна СУБД

Разом із тим, існують й інші типи СУБД, що краще адаптовані до сучасних вимог. Зокрема, нереляційні (NoSQL) системи відмовляються від жорсткої табличної структури на користь гнучких моделей даних, що дозволяє ефективно працювати з неструктурованими або динамічними даними. Вони класифікуються за чотирма основними типами:

1. Документоорієнтовані (MongoDB, CouchDB).

   Зберігають дані у форматі, аналогічному до JSON. Ідеальні для каталогів продуктів, контенту CMS або профілів користувачів, де кожен запис може мати різну структуру.

2. Колоночні (Cassandra, HBase).

   Оптимізовані для обробки великих масивів даних із високою швидкістю запису. Використовуються в аналітиці реального часу, IoT системах або фінансових даних, де потрібно швидко агрегувати інформацію по стовпцях.

3. Ключ-значення (Redis, DynamoDB).

   Працюють за принципом словника, де кожен елемент має унікальний ключ. Застосовуються для кешування, сесій користувачів або динамічних налаштувань, де критична швидкість доступу.

4. Графові (Neo4j, ArangoDB).

   Моделюють дані як вузли та зв'язки між ними. Необхідні для соцмереж, рекомендаційних систем або маршрутизації, де важливі складні взаємозв'язки (наприклад, визначення «друзів друзів»).

Такі СУБД, як MongoDB чи Cassandra, отримали популярність у проєктах, пов'язаних із Big Data, стрімінговими сервісами, такими як Netflix чи AmazonPrime, мобільними застосунками та соціальними мережами, наприклад Facebook чи LinkedIn, де класичні реляційні моделі вже не забезпечують належної гнучкості, горизонтальної масштабованості або обробки даних у реальному часі.

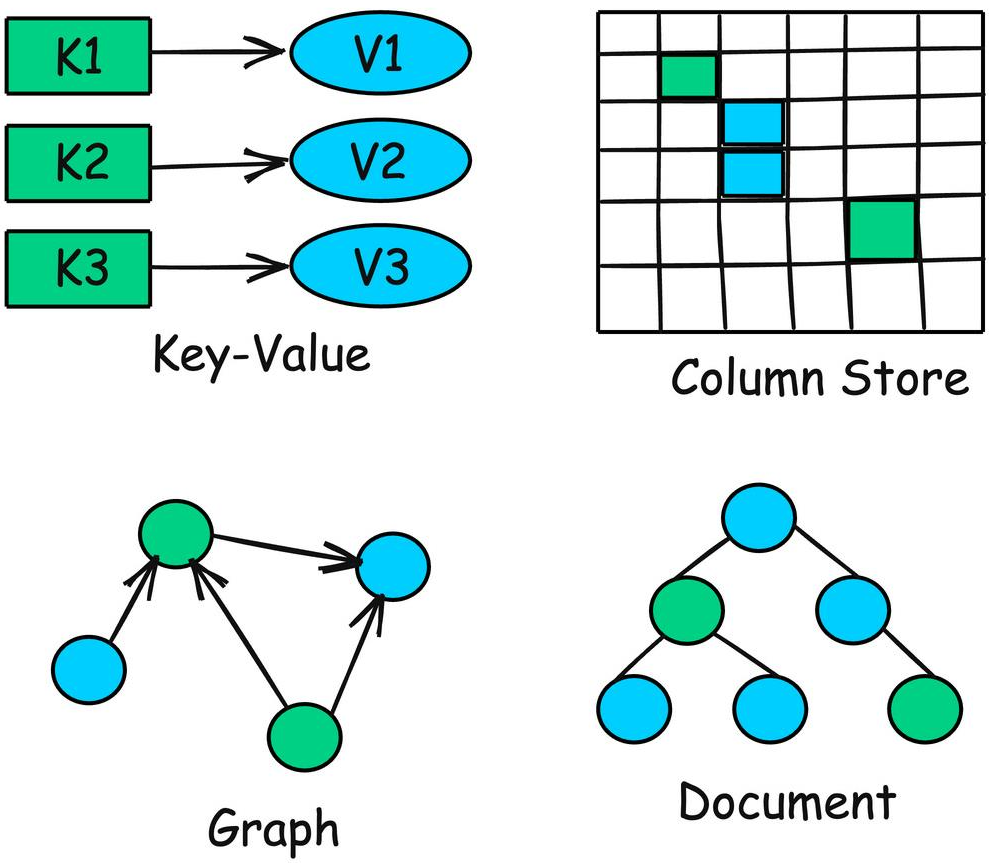


Рисунок 2.5. Нереляційні СУБД

Ще одним цікавим напрямом є об’єктно-орієнтовані СУБД, які поєднують концепції баз даних та об’єктно-орієнтованого програмування. Вони зберігають дані у вигляді об’єктів, що є зручним у випадках, коли модель даних складна та має багаторівневу структуру – наприклад, у системах проектування, наукових дослідженнях або технічних симуляціях.

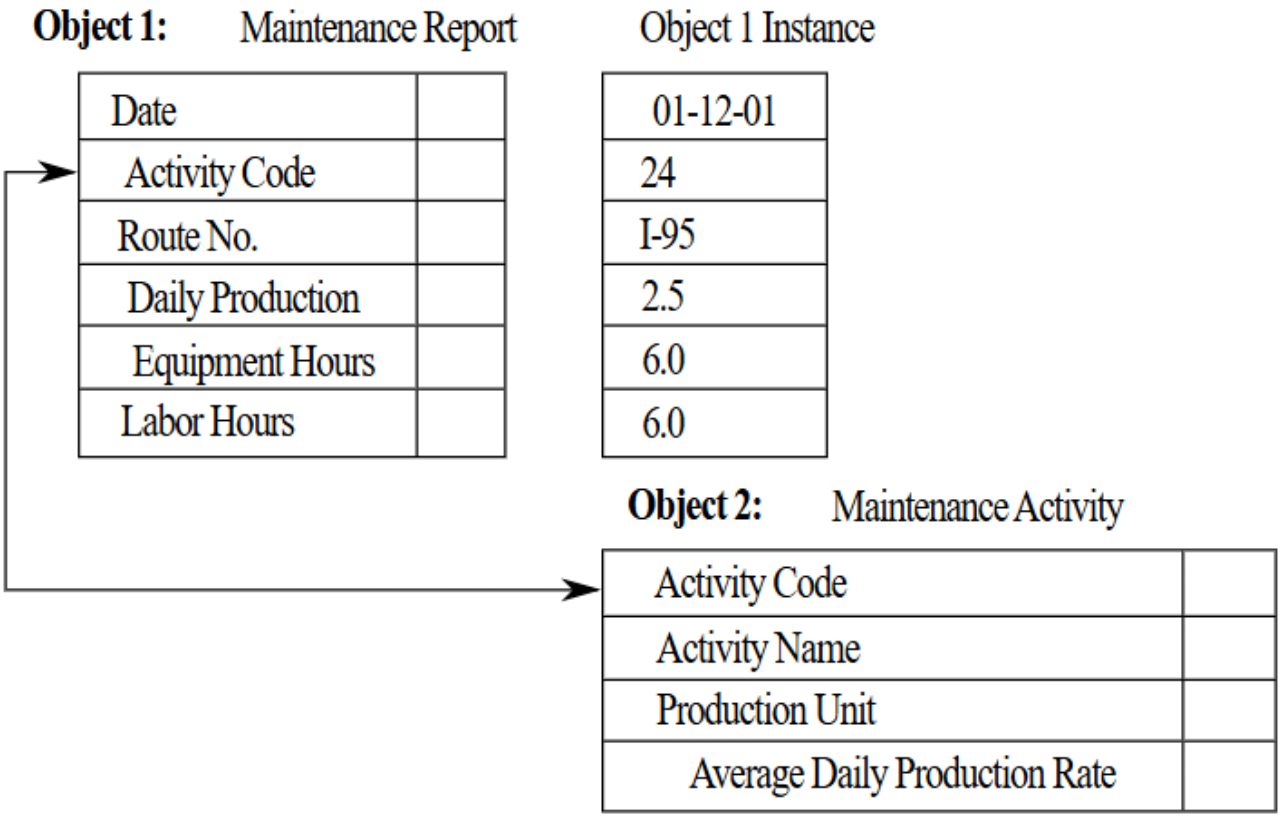


Рисунок 2.6. Об’єктно-орієнтовна СУБД

У межах даної роботи було обрано саме реляційну СУБД, оскільки вона найкраще відповідає структурі та природі тих даних, з якими працює розроблений додаток. У контексті прогнозування завершення курсу студентом школи програмування критично важливо зберігати чіткі зв’язки між сутностями — зокрема, між студентами, їх відвідуваннями, виконаними завданнями, оцінками, участю в активностях та іншими навчальними показниками. Така структура чудово піддається формалізації у вигляді таблиць, а реляційна модель якраз і забезпечує найбільш логічну та стабільну основу для цього.

Водночас, з технічного погляду, реляційна СУБД є стабільною та перевіреною технологією, яку легко інтегрувати в середовище Python. Такі системи як PostgreSQL або SQLite мають безліч бібліотек, що забезпечують прямий доступ до даних, що значно полегшує процес обробки та дозволяє зосередитись саме на побудові якісної моделі, а не на інфраструктурних проблемах.

2.2.3 Вибір бібліотек для виконання робіт

Для виконання даної роботи необхідно використати бібліотеки для роботи з багатьма аспектами системи. В першу чергу, необхідно розглянути роботу з даними.

До базової, найпершої, дії з даними є завантаження та вивантаження даних в MySQL. Хоча і існує багато бібліотек, що дозволяють працювати з MySQL, вибір кінцевого варіанту буде не складним, оскільки існує бібліотека «mysql-connector-python», яка є офіційним клієнтом від розробників MySQL. Вибір саме цієї бібліотеки зумовлений її стабільністю, активною підтримкою та повною сумісністю з можливостями самої СУБД, що, у свою чергу, забезпечує надійний і прозорий канал обміну даними між додатком та базою.

Бібліотека «mysql-connector-python» дозволяє зручно підключатися до бази, виконувати запити будь-якої складності, обробляти результати та, що важливо, працювати з параметризованими запитами, що допомагає уникати проблем з SQL-ін’єкціями. Крім того, бібліотека підтримує транзакційність, що дає змогу контролювати зміни в базі на всіх етапах виконання додатку, і це особливо важливо при збереженні аналітичних результатів чи створенні логів системи.

Для подальшої роботи з даними необхідно використовувати бібліотеку, що водночас легко обробить інформацію від «mysql-connector-python» і буде підтримувати роботу з бібліотеками для роботи з машинним навчанням, такими як «scikit-learn».

Бібліотекою, що ідеально підходить під вищеописані вимоги є «pandas» , яка на сьогодні є одним із ключових інструментів у роботі з табличними даними в середовищі Python. Її популярність зумовлена, передусім, високою гнучкістю у поводженні з даними, простотою синтаксису та широким спектром функціональних можливостей.

Основними структурами, які пропонує «pandas», є Series, одновимірний масив з підписами, що використовується для збереження стовпців або окремих значень, і DataFrame, тобто, двовимірна таблиця, схожа на звичні таблиці Excel або SQL-таблиці. Саме DataFrame є серцем більшості операцій, які виконує розробник при аналізі даних.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3 entries, 0 to 2
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  3 non-null      int64
 1   Survived     3 non-null      int64
 2   Pclass       3 non-null      int64
 3   Name         3 non-null      object
 4   Sex          3 non-null      object
 5   Age          3 non-null      int64
 6   SibSp        3 non-null      int64
 7   Parch        3 non-null      int64
 8   Ticket       3 non-null      object
 9   Fare         3 non-null      float64
 10  Cabin        1 non-null      object
 11  Embarked     3 non-null      object
dtypes: float64(1), int64(6), object(5)
memory usage: 420.0+ bytes
None
```

Рисунок 2.7. Приклад перегляду DataFrame

Функціональність бібліотеки охоплює повний життєвий цикл роботи з даними. З самого початку можна зчитати дані з файлів різних форматів: CSV, Excel, JSON, SQL або навіть безпосередньо з API-запитів. Після цього доступні гнучкі можливості для очищення даних. Заповнення або видалення пропущених значень, перейменування стовпців, приведення типів даних до потрібного формату, а також фільтрація та сортування за заданими критеріями. Важливим є також те, що «pandas» дозволяє об'єднувати кілька таблиць в одну, операції join, merge, concatenate, що особливо зручно при роботі з розрізненими джерелами інформації[21].

Для аналізу доступні методи обчислення середніх значень, медіани, дисперсії, стандартного відхилення, побудови частотних розподілів та інших статистичних характеристик. Крім того, можна швидко згрупувати дані за певним критерієм, що дає змогу отримати агреговані показники по групах студентів, курсів, викладачів тощо.

```
count    12.000000
mean     35.833333
std      12.755272
min      19.000000
25%      27.250000
50%      33.000000
75%      42.750000
max      60.000000
Name: Age, dtype: float64
```

Рисунок 2.8. Приклад статистичного обрахунку колонки DataFrame

Ще однією важливою перевагою є інтеграція з іншими бібліотеками. Результати, підготовлені за допомогою «pandas», легко передаються до методів бібліотек «scikit-learn» для побудови моделей машинного навчання або «matplotlib» чи «seaborn» для візуалізації.

Для ефективного аналізу даних, особливо в контексті машинного навчання та прогнозування, важливо не лише працювати з числами, а й

візуалізувати результати. Саме для цього в екосистемі Python існує кілька потужних бібліотек, серед яких найбільш відомими є «matplotlib» та «seaborn». Обидві бібліотеки дозволяють створювати графіки, діаграми та інші типи візуального представлення даних, проте підходи та рівень контролю над побудовою у них суттєво різняться.

Бібліотека «Seaborn» – це високорівнева бібліотека для статистичної візуалізації, яка побудована поверх matplotlib і значно спрощує створення складних графіків. Вона чудово підходить для швидкої побудови теплових карт, boxplot, графіків розсіювання з групуванням за категоріями, регресійних ліній, тощо. Бібліотека «Seaborn» автоматично оформлює графіки в єдиному стилі, що дозволяє отримати візуально приємні результати з мінімальним втручанням розробника. Проте ця простота іноді обертається обмеженнями, зокрема, складно реалізувати кастомні підписи, нетипову анімацію або дуже специфічні формати графіків.

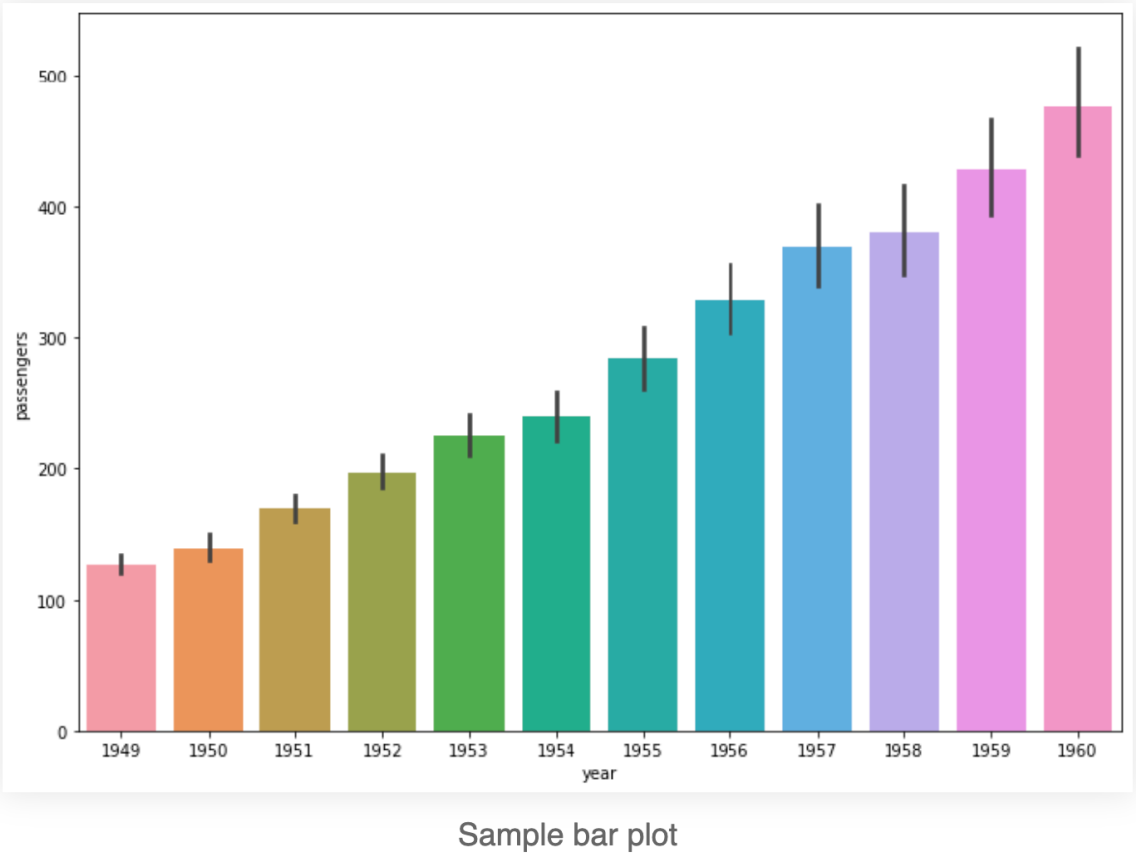


Рисунок 2.9. Приклад графіку бібліотеки «Seaborn»

На противагу цьому, бібліотека «matplotlib» виступає як низькорівнева бібліотека, яка надає повний контроль над кожним елементом графіка. Хоча синтаксис може здаватися громіздким, особливо на початкових етапах, саме «matplotlib» дозволяє будувати візуалізації будь-якої складності, включаючи анімовані графіки, поєднання кількох підграфіків, багаторівневі підписи, зміну масштабу, кольорової палітри та інші параметри. Крім того, ця бібліотека є універсальним стандартом у науковій спільноті, що робить її зручною для інтеграції з іншими інструментами й форматами звітності.

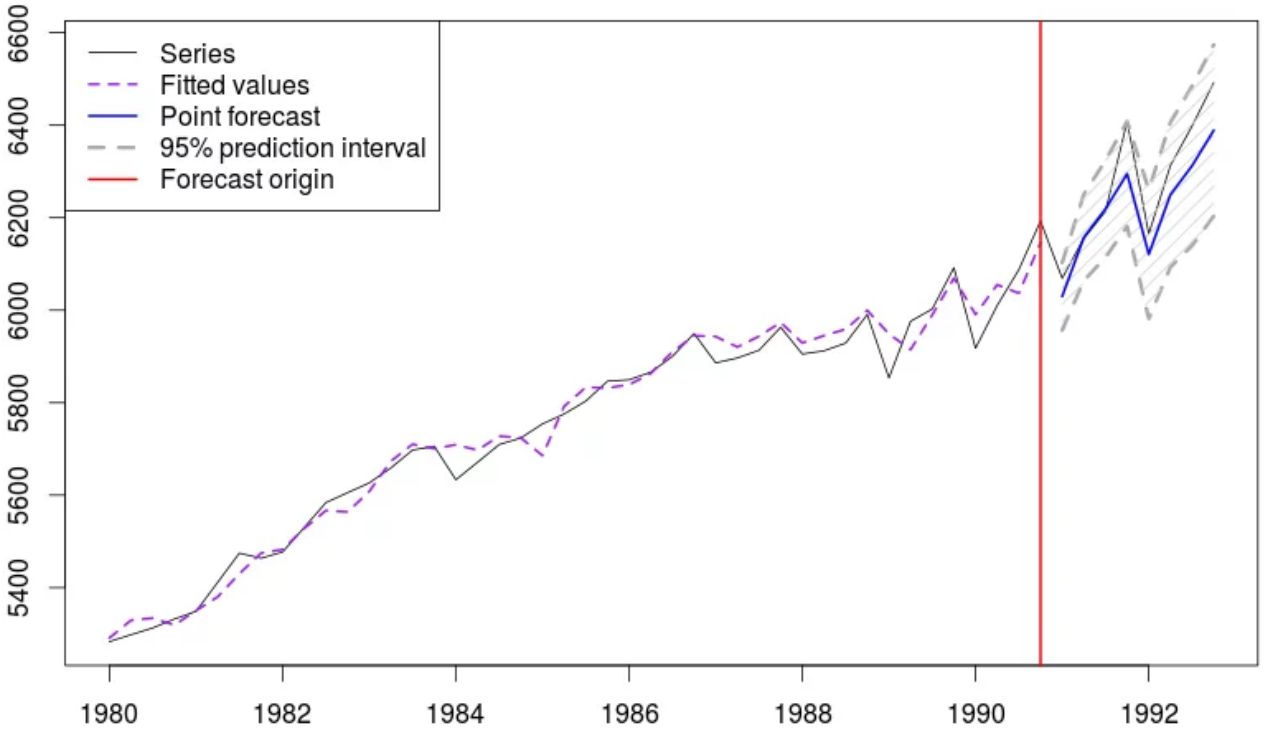


Рисунок 2.10. Приклад графіку бібліотеки «matplotlib»

У межах розробки додатку вибір був зроблений на користь бібліотеки «matplotlib». Основним аргументом є її гнучкість і контроль над деталями, які можуть бути критичними для точного відображення динаміки навчання студентів, побудови звітних графіків для внутрішнього користування компанії та глибшого аналізу результатів моделей. Попри те, що на побудову графіків може знадобитися трохи більше коду, результат виходить повністю адаптованим під завдання.

Для реалізації прогнозної частини додатку, основною потребою є побудова та навчання трьох моделей: K-Nearest Neighbors (KNN), моделі Кокса для аналізу виживання, а також RankNet, нейронної моделі, орієнтованої на ранжування. Для виконання такого завдання необхідна бібліотека, яка не тільки підтримує роботу з кожною із зазначених моделей, а й дозволяє легко масштабувати проєкт, забезпечує добру документацію та активну спільноту користувачів. З урахуванням усіх вимог, найбільш доцільним вибором стає бібліотека «scikit-learn» у поєднанні з «lifelines» і «torch».

Бібліотека «scikit-learn» вважається класичним інструментом для реалізації методів машинного навчання в середовищі Python. Вона підтримує велику кількість алгоритмів класифікації, регресії, кластеризації, зокрема і KNN. У цій бібліотеці реалізація K-Nearest Neighbors є стабільною, добре протестованою і має можливість налаштування гіперпараметрів, таких як кількість сусідів, тип метрики, спосіб зважування тощо. Також «scikit-learn» пропонує вбудовані інструменти для попередньої обробки даних, крос-валідації, підбору параметрів і побудови пайплайнів, що дозволяє значно спростити процес розробки та експериментування.

Щодо моделі Кокса, то тут більш вузькоспеціалізованим і адаптованим рішенням є бібліотека «lifelines», яка спеціально призначена для роботи з аналізом виживання. У ній реалізовано різні модифікації моделі Кокса, включаючи регуляризовані варіанти. Бібліотека «lifelines» зручно інтегрується з «pandas», що забезпечує комфортну роботу з табличними даними. І хоча ця бібліотека не є частиною «scikit-learn», вона логічно доповнює її, дозволяючи покрити специфічні задачі, пов’язані з тривалістю навчання або ймовірністю завершення курсу у часовій перспективі.

Щодо реалізації RankNet, яка базується на нейронних мережах і часто використовується для задач навчання з ранжуванням. Тут найкращим вибором буде використання бібліотеки «PyTorch». Ця бібліотека надає повний контроль над архітектурою моделі, тренувальним циклом та функцією втрат, що є важливим для кастомних моделей, як RankNet. До того ж, завдяки активному

розвитку спільноти, доступно багато прикладів реалізації подібних нейронних структур, що може значно пришвидшити розробку.

Таким чином, для покриття всіх трьох моделей буде використано комбінацію бібліотек, з опорою на «scikit-learn» для KNN, «lifelines» для моделі Кокса та «PyTorch» для RankNet. Такий підхід дозволить забезпечити якісну реалізацію кожного з алгоритмів без втрати продуктивності або точності.

2.2.4 Вибір бібліотеки для побудови інтерфейсу

В якості фінального кроку розробки додатку необхідно обрати бібліотеку для побудови користувацьких інтерфейсів. Для рішення таких задач існує декілька основних бібліотек.

Першим, найпростішим вибором, є бібліотека «Tkinter». Це вбудована бібліотека для створення графічного інтерфейсу користувача у мові програмування Python. Вона постачається разом із стандартною дистрибуцією Python, тому не потребує окремого встановлення, що значно спрощує початок роботи з нею. Завдяки простоті використання, хорошій документації та легкості в освоєнні «Tkinter» часто обирають розробники, яким потрібно оперативно створити базовий інтерфейс для настільного застосунку.

Використовуючи бібліотеку «Tkinter», можна створювати стандартні графічні компоненти, такі як кнопки, поля вводу, таблиці, списки, меню, вкладки та інші базові елементи управління. Її функціонал дозволяє будувати як прості, так і відносно складні форми, інтерактивні панелі та діалогові вікна. Особливо зручно те, що ця бібліотека добре інтегрується з іншими частинами Python-додатку, включаючи обробку даних, взаємодію з файлами, базами даних чи іншими модулями.

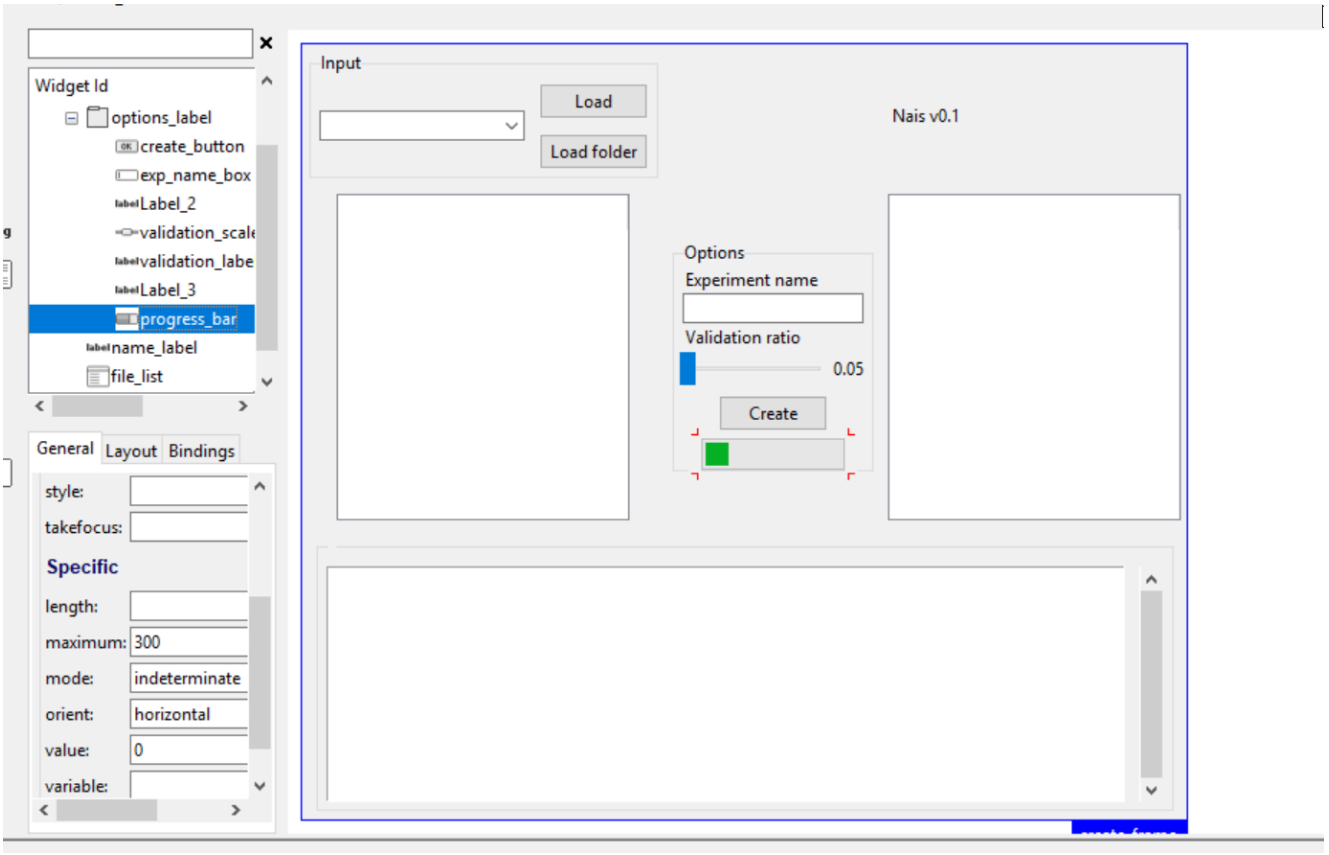


Рисунок 2.11. Приклад інтерфейсу, побудованого на бібліотеці «Tkinter»

У плані стилю та візуальної привабливості бібліотека «Tkinter» нерідко поступається сучаснішим рішенням, оскільки її інтерфейс виглядає досить просто і навіть дещо застаріло, особливо у стандартному відображенні на різних операційних системах. Проте, враховуючи завдання поточного проєкту створення внутрішнього інструменту для обмеженого кола користувачів цей недолік не є критичним. Навпаки, простота і стабільність «Tkinter» дозволяють зосередитися на основній логіці додатку без витрат на дизайн інтерфейсу чи складну верстку. Саме тому бібліотека «Tkinter» може стати цілком виправданим і практичним вибором у даному контексті.

Наступним варіантом є бібліотека «PyQt6». Це потужна бібліотека для створення сучасних графічних інтерфейсів у Python, яка є обгорткою над фреймворком Qt шостої версії. Цей інструмент надає доступ до великого набору функцій для побудови складних, гнучких і візуально привабливих інтерфейсів. Бібліотека підтримує модульну архітектуру, має розвинуту систему сигналів і слотів, дозволяє реалізовувати багатовіконні додатки, а

також пропонує широкі можливості кастомізації елементів управління. Все це робить її особливо привабливою для створення повноцінних десктопних програм з інтуїтивно зрозумілим та професійним виглядом.

Ще однією перевагою бібліотеки «PyQt6» є підтримка стилізації інтерфейсу за допомогою CSS-подібного синтаксису (Qt Style Sheets). Це дозволяє точно контролювати зовнішній вигляд елементів інтерфейсу від кольорів і шрифтів до розмірів полів, відступів і навіть поведінки при наведенні чи натисканні. Така гнучкість відкриває широкі можливості для кастомізації й надає розробнику інструменти для створення професійного, візуально завершеного продукту. Однак у контексті внутрішнього сервісу, де естетика не є ключовим пріоритетом, глибока стилізація може бути зайвою і лише ускладнити процес розробки без суттєвої користі.

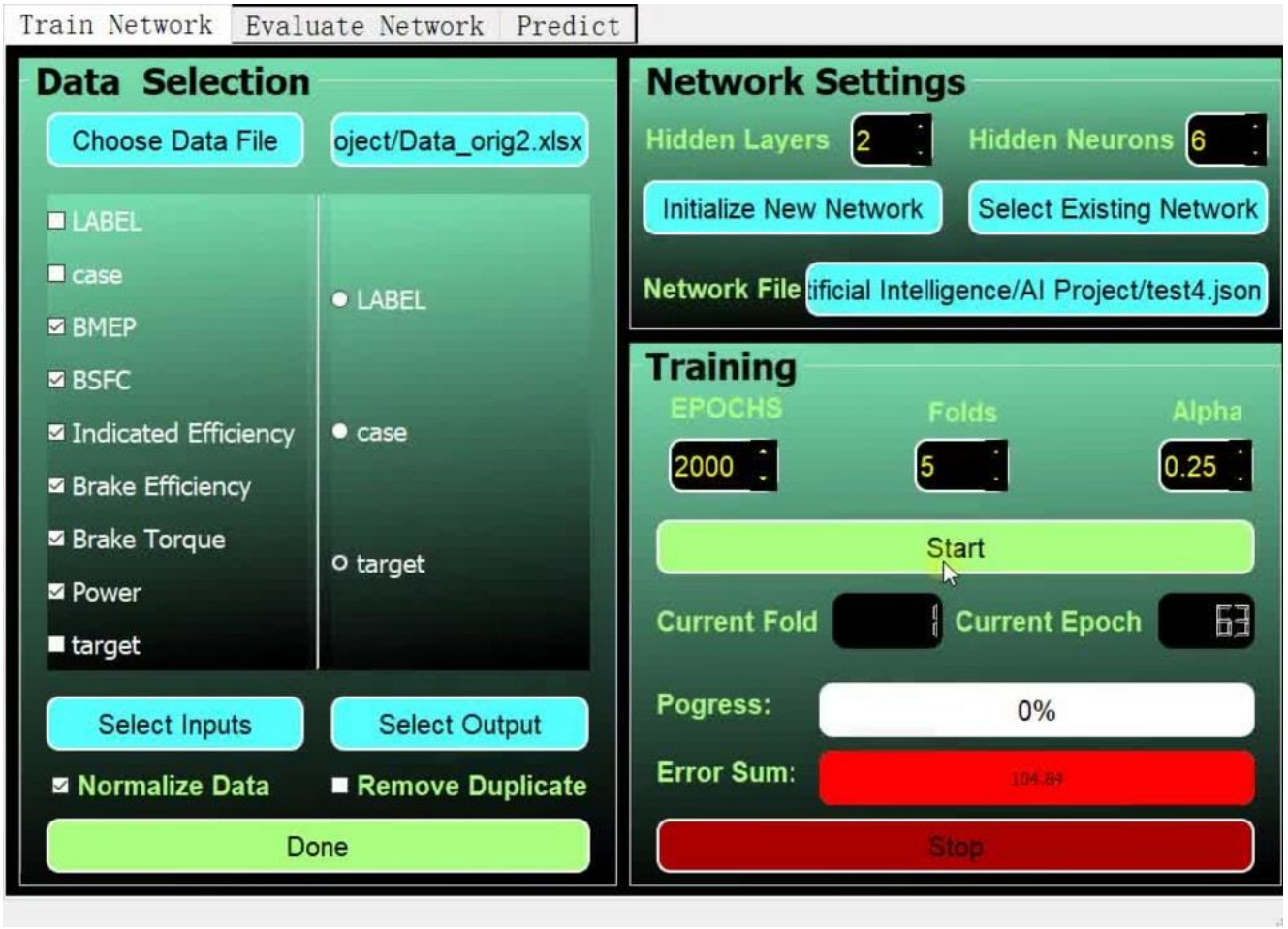


Рисунок 2.12. Приклад інтерфейсу, побудованого на бібліотеці «PyQt6»

Важливою перевагою бібліотеки «PyQt6» є її універсальність та підтримка широкого спектра віджетів і компонентів: базові кнопки і поля вводу до вбудованих браузерів, графіків і таблиць із підтримкою фільтрації, сортування чи багаторівневої навігації. Водночас ця бібліотека забезпечує кросплатформність, дозволяючи інтерфейсу коректно адаптуватися до Windows, Linux і macOS, що сприяє стабільній роботі застосунків.

Окремої уваги заслуговує додаток Qt Designer. Це графічний інструмент для створення інтерфейсів, який постачається разом з фреймворком Qt. За його допомогою можна швидко і зручно розміщувати віджети, налаштовувати їхні властивості та будувати ієрархію елементів без необхідності писати код вручну. Qt Designer зберігає створені графічні макети у форматі «.ui», який легко інтегрується у Python-застосунок через спеціальний конвертер або шляхом прямого імпорту. Це значно пришвидшує розробку, особливо коли потрібно часто змінювати компонування або протестувати різні варіанти візуального оформлення.

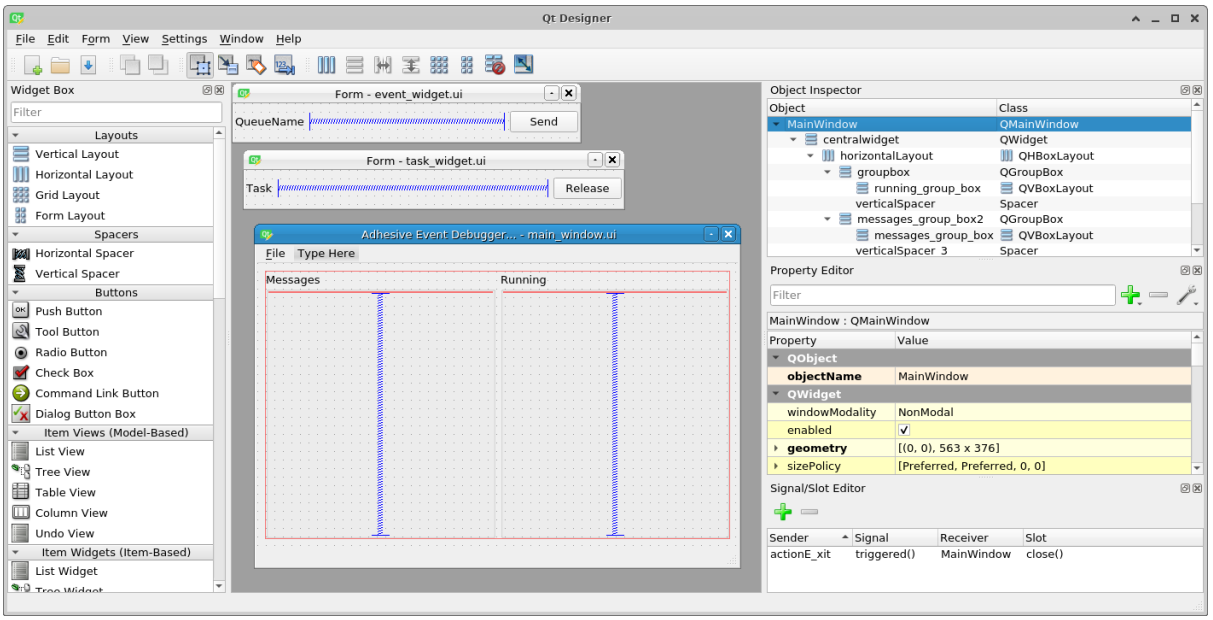


Рисунок 2.13. Додаток QtDesigner

Разом з тим, бібліотека «PyQt6» це досить важкий інструмент як у плані навчання, так і в технічному сенсі. Вона вимагає встановлення окремих пакетів, є складнішою в налаштуванні, має специфічний синтаксис і вимагає більше

часу для освоєння. У випадку внутрішнього інструменту, який не передбачає широку аудиторію чи часту взаємодію з кінцевим користувачем, така складність може виявитися надмірною. Особливо, якщо інтерфейс додатку не є ключовим елементом функціоналу. Отже, попри очевидну технічну перевагу «PyQt6», її доцільність у поточному проєкті викликає сумніви через надлишковість можливостей і складність реалізації, які, ймовірно, не будуть повністю використані.

Наостанок, розглянемо бібліотеку «Kivy». Це потужний інструмент для створення кросплатформених інтерфейсів, який орієнтований переважно на розробку додатків із сучасним, динамічним дизайном. Цей інструмент дає змогу створювати застосунки, що стабільно працюють на різних операційних системах – Windows, Linux, macOS, а також на мобільних платформах Android і iOS. Архітектура бібліотеки «Kivy» побудована так, щоб забезпечити максимальну інтерактивність, що робить її зручною для реалізації жестів, анімацій, мультитач-управління та роботи з сенсорними екранами. На відміну від традиційних фреймворків, таких як «Tkinter» або «PyQt6», бібліотека «Kivy» реалізує зовсім інший підхід до побудови та оформлення інтерфейсу, орієнтуючись переважно на концепції, властиві мобільній розробці.

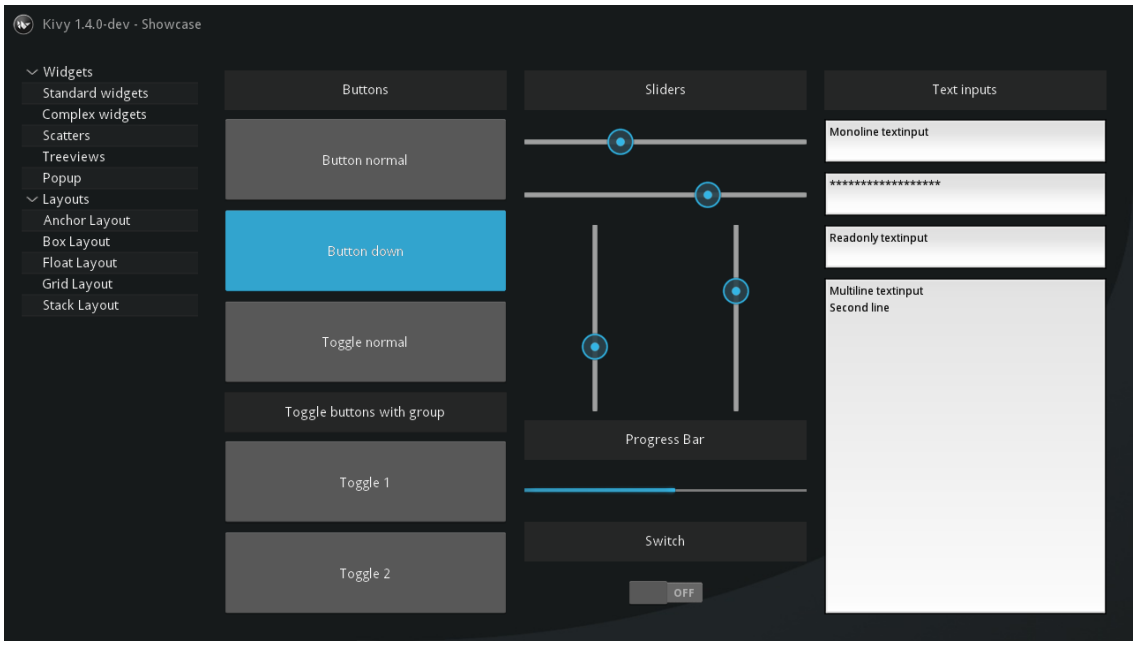


Рисунок 2.14. Приклад інтерфейсу, побудованого на бібліотеці «Kivy»

Створення інтерфейсів у «Kivy» відбувається програмно або за допомогою декларативного модуля «.kv». Стиль написання подібний до CSS, однак має власний синтаксис, що вимагає певного часу на вивчення. Бібліотека «Kivy» надає гнучке управління графічними елементами, дозволяючи створювати унікальні компоненти та анімації, що значно вирізняє її серед інших бібліотек. Проте через це початковий поріг входу є вищим, а інтерфейси іноді виглядають менш нативно на настільних ОС. Тому, хоч «Kivy» і є перспективною платформою для розробки багатоплатформених інтерфейсів, у випадку створення інструменту для внутрішнього користування в компанії, де функціональність важливіша за візуальну привабливість, її можливості є надмірними.

Після розгляду кількох бібліотек для створення графічного інтерфейсу користувача можна зробити обґрунтований вибір на користь бібліотеки «PyQt6». Хоча «Tkinter» є найпростішим інструментом для створення графічних інтерфейсів у середовищі Python, його функціональність доволі обмежена для реалізації сучасного дизайну та інтерактивності. Натомість бібліотека «Kivy», попри свою гнучкість і підтримку мультимедійних компонентів, більше підходить для мобільних застосунків або кросплатформених проєктів, що мають підвищені вимоги до візуального оформлення та графіки.

На тлі цього «PyQt6» виглядає як оптимальне рішення для реалізації внутрішнього робочого додатку. Його переваги – це зріла екосистема, підтримка створення складних елементів інтерфейсу, інтеграція з Qt Designer, що значно спрощує процес розробки форм, а також підтримка стилізації через CSS, що дозволяє швидко адаптувати вигляд програми під потреби користувача.

Враховуючи також мій наявний досвід роботи з цією бібліотекою, її використання забезпечить не лише ефективність у реалізації поставлених задач, а й зменшить час на розробку та налагодження інтерфейсу. Саме тому вибір було зроблено на користь «PyQt6» як найбільш гнучкої та зручної у даному контексті технології.

2.3 Висновки до другого розділу

У результаті проведеного аналізу було здійснено вибір технологій, необхідних для розробки настільного додатку, що дозволяє прогнозувати успішність завершення курсу студентом школи програмування. Основною мовою програмування обрано Python універсальний інструмент, який поєднує в собі зручність синтаксису, широку підтримку наукових бібліотек, а також розвинуту екосистему для створення інтерфейсів, роботи з базами даних та реалізації моделей машинного навчання.

Серед баз даних було вирішено використовувати реляційну СУБД, зокрема MySQL, що забезпечує надійне зберігання структурованих даних, зручний доступ до них та можливість ефективного масштабування в межах майбутньої інтеграції з внутрішніми системами компанії. Для взаємодії з цією базою обрано бібліотеку «mysql-connector-python», яка забезпечує просте та ефективне підключення до MySQL без зайвої складності у налаштуванні.

Для обробки та підготовки даних було використано бібліотеку «pandas», яка надає зручні інструменти для роботи з табличними даними: групування, фільтрація, обчислення статистичних показників і формування ознак для моделей.

Для візуалізації результатів обрали бібліотеку «matplotlib», яка, хоча й поступається «seaborn» у плані візуальної привабливості, натомість вирізняється універсальністю, гнучкістю та широкими можливостями для створення кастомізованих графіків, що дозволяє максимально адаптувати візуалізацію під специфіку конкретного дослідження.

Для побудови моделей машинного навчання, серед яких KNN, модель Кокса та RankNet, було обрано бібліотеку «scikit-learn» у поєднанні з «lifelines» та «tensorflow», що дозволяє реалізувати необхідні підходи як у класичних методах, так і в нейромережевих. Це забезпечує гнучкість та масштабованість у підходах до прогнозування.

Для реалізації інтерфейсу розглянуто декілька бібліотек, з яких остаточний вибір було зроблено на користь «PyQt6». Саме ця бібліотека

поєднує гнучкість, підтримку складних UI-рішень, наявність зручного графічного конструктора Qt Designer, а також підтримку CSS-стилізації, що дозволяє створити естетичний, логічний і функціональний інтерфейс. До того ж, мій досвід роботи з цією бібліотекою значно пришвидшує процес розробки та знижує поріг входження при підтримці проєкту.

У підсумку можна зазначити, що обрані інструменти є взаємодоповнюючими та утворюють надійну технічну основу для реалізації функціонального, гнучкого та масштабованого програмного продукту для прогнозування успішності навчання студентів.

РОЗДІЛ 3. РЕАЛІЗАЦІЯ НАСТІЛЬНОГО ДОДАТКУ НА ОСНОВІ РЕАЛЬНИХ ДАНИХ.

3.1 Розробка інтерфейсу користувача

Проектований додаток можна охарактеризувати як настільний додаток. Настільний додаток розроблено під платформу Windows, проте також повинен працювати на інших операційних системах.

Для початку роботу з інтерфейсом було розроблено інтерфейс користувача за допомогою програми Qt Designer.

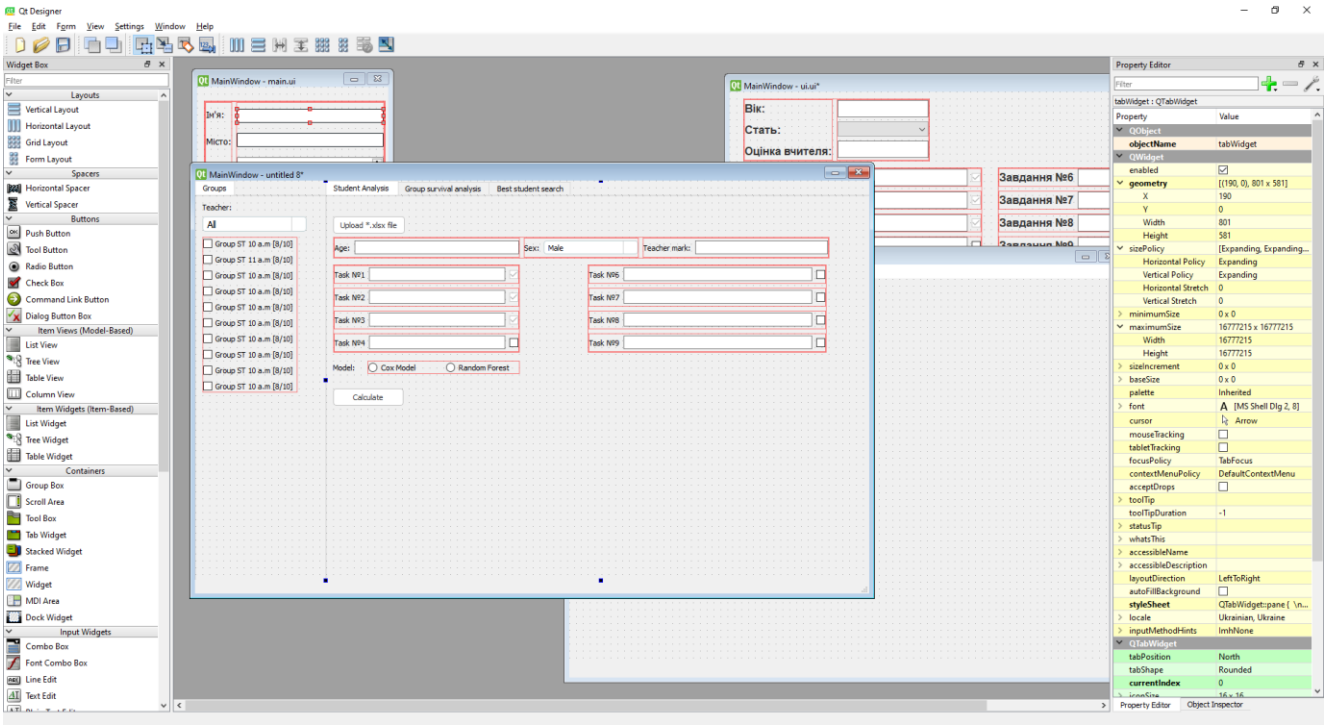


Рисунок 3.1. Процес розробки інтерфейсу користувача

Вікно буде розділено на 4 основні частини. Кожна з них відповідає за свою функціональність та винесена в окрему вкладку.

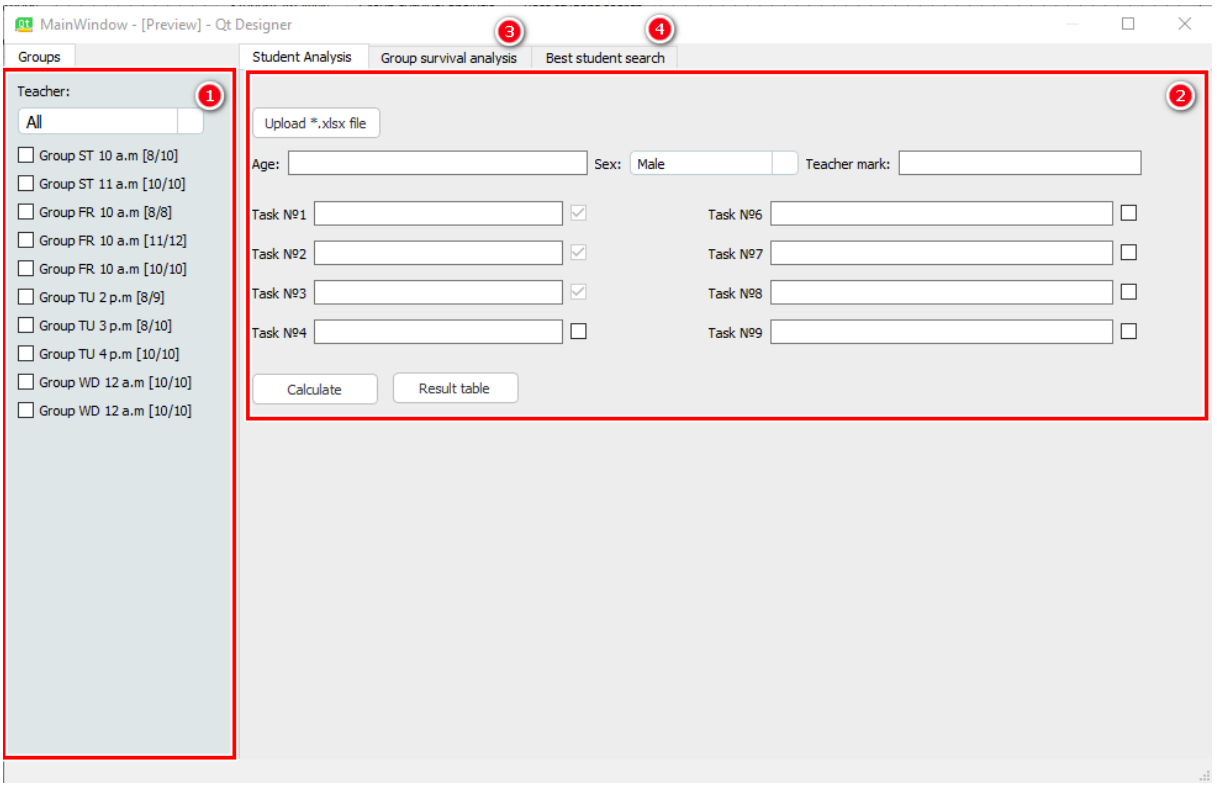


Рисунок 3.2. Вкладки в інтерфейсі користувача

Розглянемо кожен елемент детальніше. Перша частина (1 на рисунку 3.2), містить список груп, що належать конкретно обраним викладачам. Дана вкладка необхідна для роботи вкладок «Аналіз виживання груп» та «Пошук кращого студента». Коли користувач знаходиться на вкладці «Аналіз студента» прапорці груп недоступні до натискання.

Рисунок 3.3. Відображення активних та недоступних прапорців

Також, групи можна фільтрувати за викладачами. Для цього, необхідно скористатись випадним списком «Вчителі» та обрати потрібного викладача. За потреби переглянути всі групи потрібно обрати пункт випадного списку «Всі».

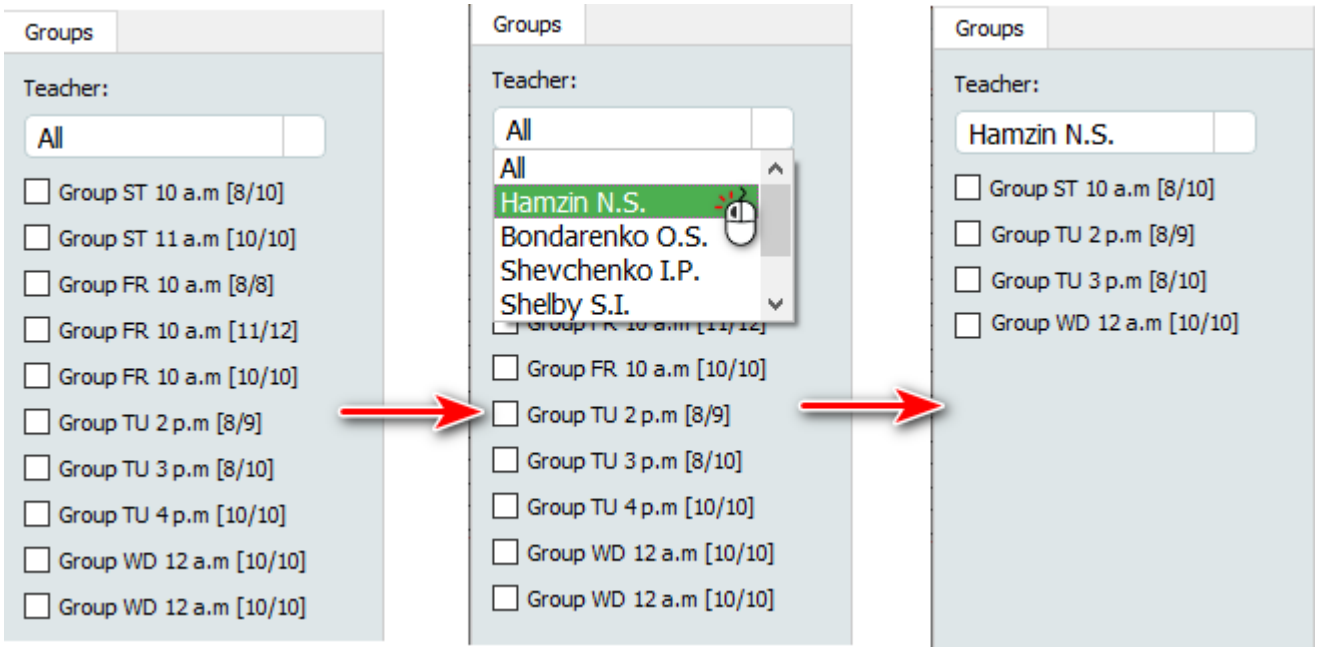


Рисунок 3.4. Зміна відображення груп за викладачем

Друга частина (2 на рисунку 3.2) вирішує проблему, описану в розділі 2.1.1 Вирішення задачі виявлення студентів, що запишуться на курс після пробного заняття. Для отримання прогнозу необхідно або заповнити інформацію по окремому студенту, заповнивши поля, виділені червоним на рисунку 3.5, або завантаживши *.xlsx файл до програми, за допомогою відповідної кнопки, що виділена синім на рисунку 3.5.

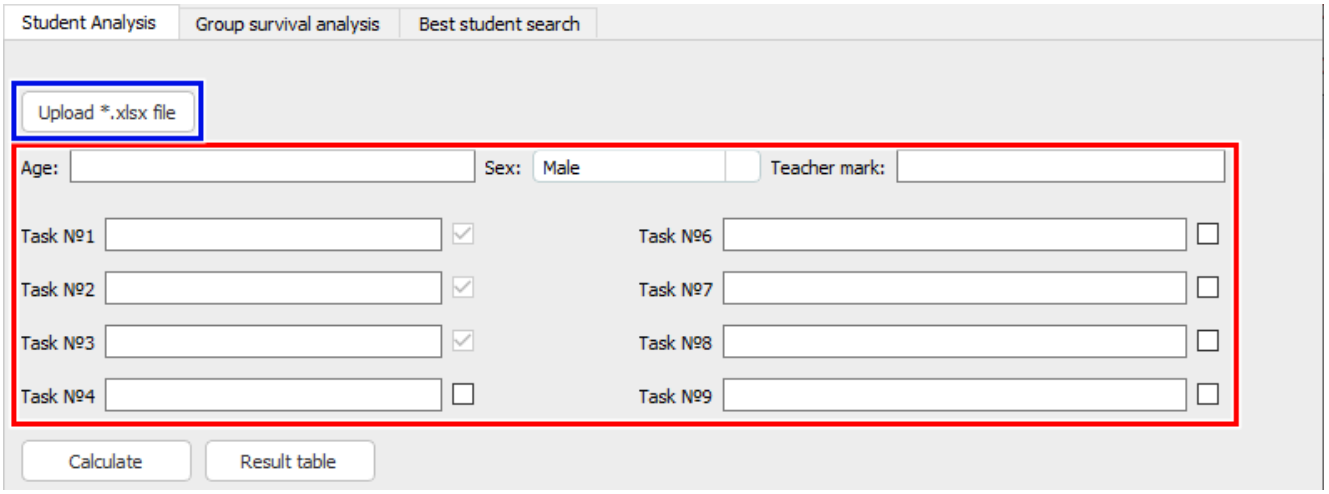


Рисунок 3.5. Вкладка «Аналіз студента»

Після завантаження *.xlsx файлу текст на кнопці буде змінено на «Очистити», а поля для введення одного студента (виділено червоним на рис. 3.5) буде заблоковано для введення.

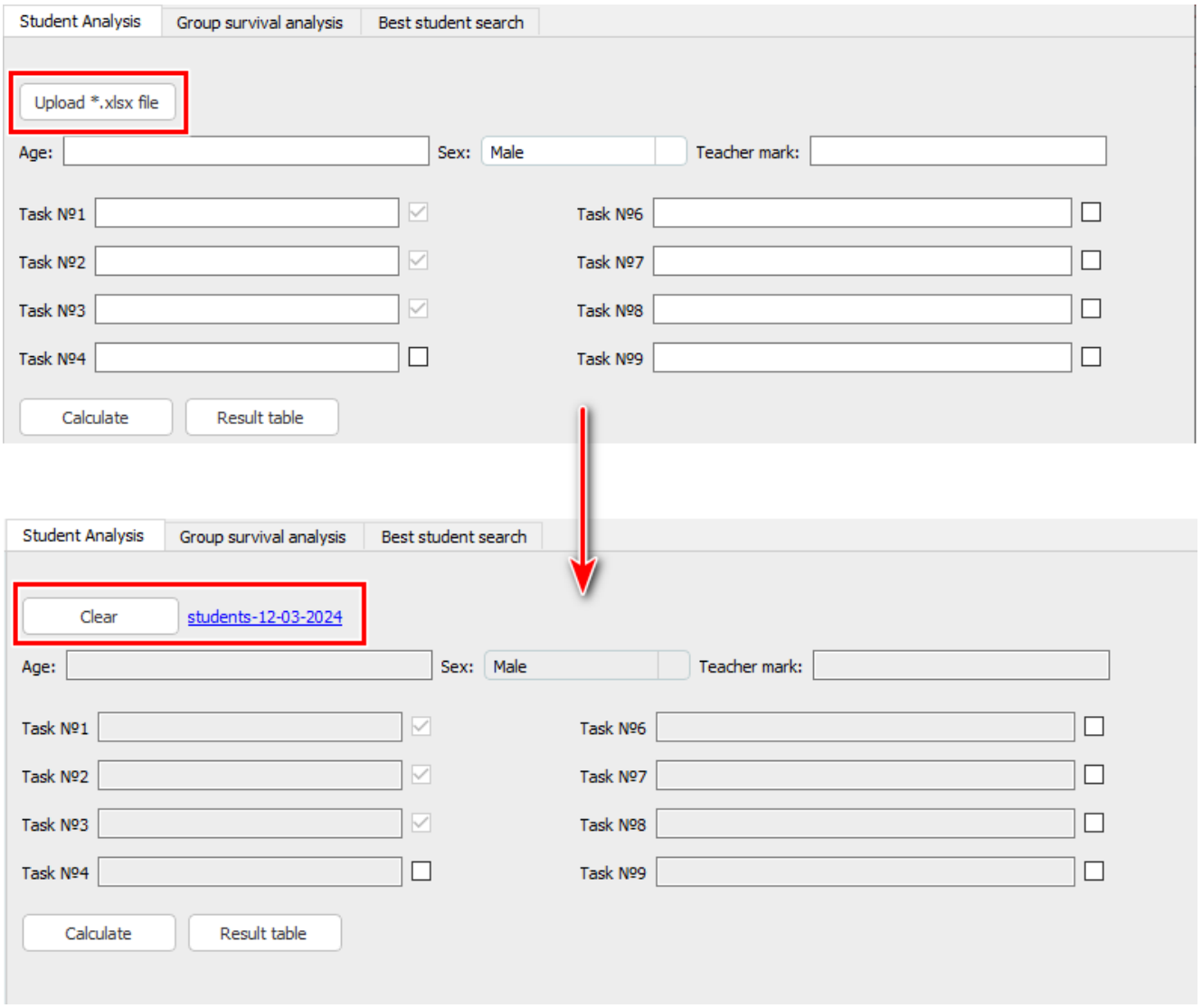


Рисунок 3.6. Блокування полів при завантаженні файлу

При натисканні на кнопку «Очистити» файл буде прибрано, а поля для введення інформації про одного студента буде розблоковано (виділено червоним на рис. 3.5).

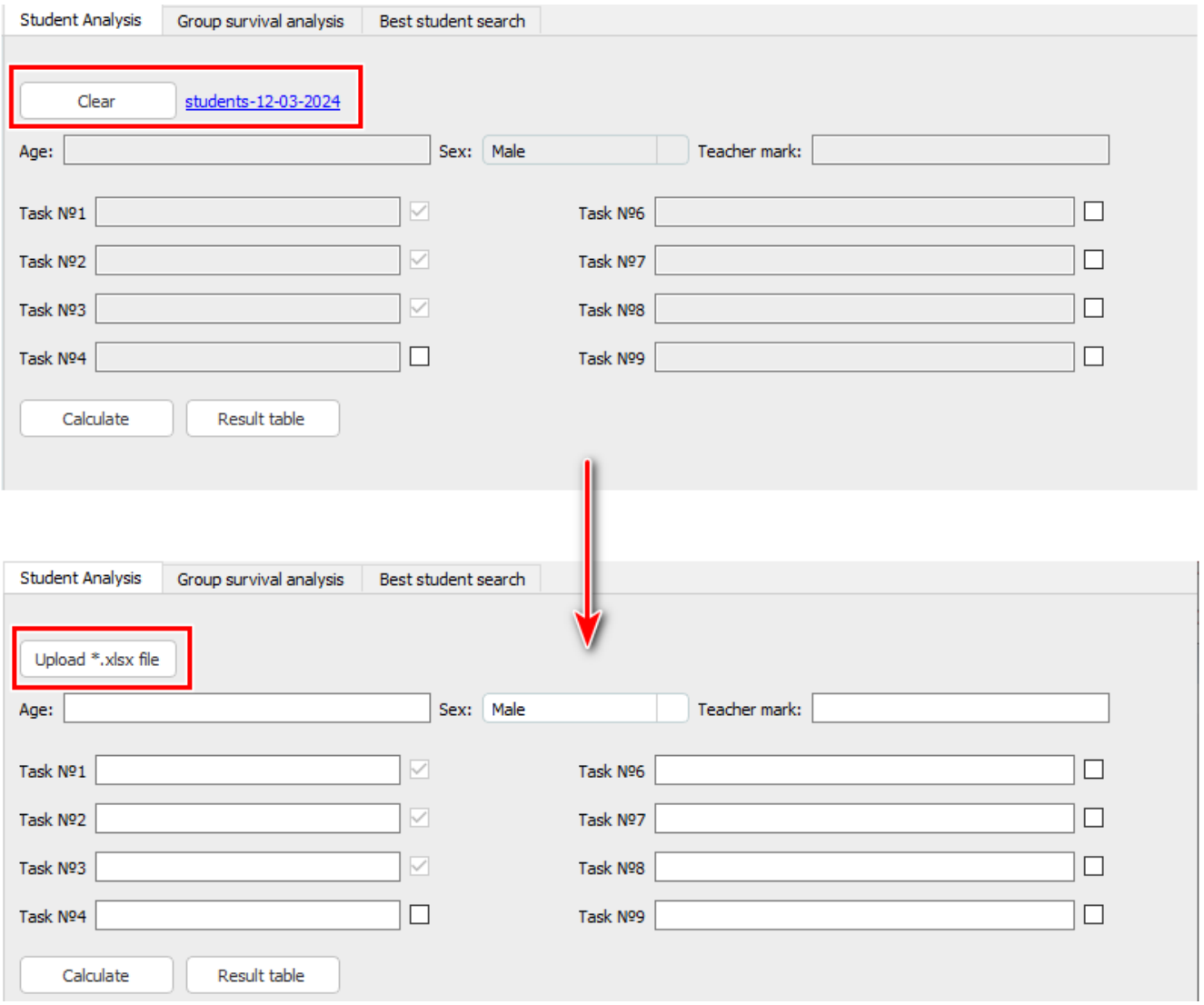


Рисунок 3.7. Розблокування полів при завантаженні файлу

Після заповнення всіх полів, виділених червоним на рисунку 3.5 або завантаженні файлу буде доступна до натискання кнопка «Розрахувати».

При натисканні буде виконано розрахунок відповідно до заповнених параметрів. Після завершення розрахунку буде виведено невелике інформаційне вікно про успішне завершення розрахунку.

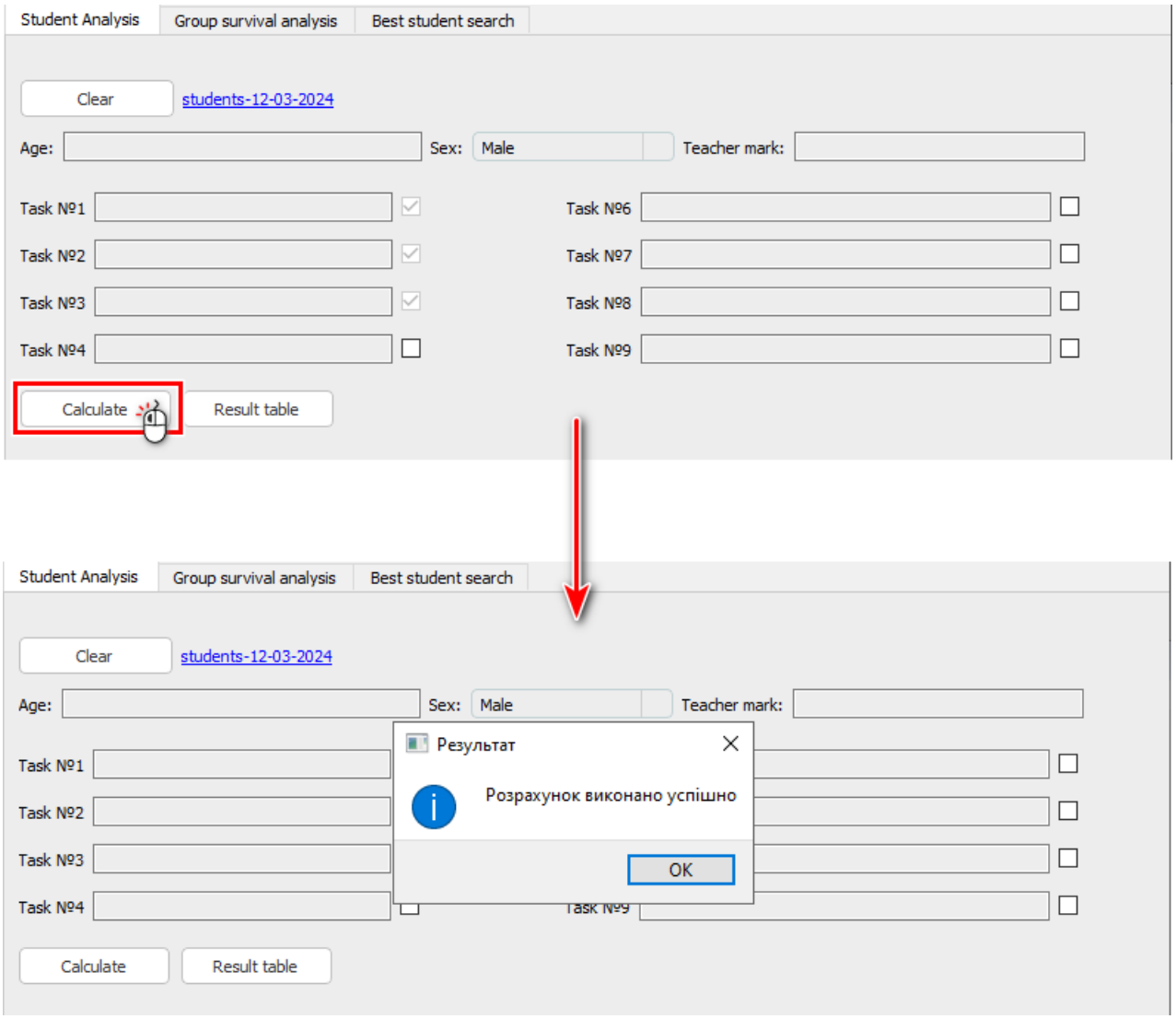


Рисунок 3.8. Інформативне вікно про завершення розрахунку

Для того, щоб переглянути результат прогнозування необхідно натиснути на кнопку «Таблиця результатів». Після натискання, буде відкрито вікно з результатами, що зображене на рис. 3.9.

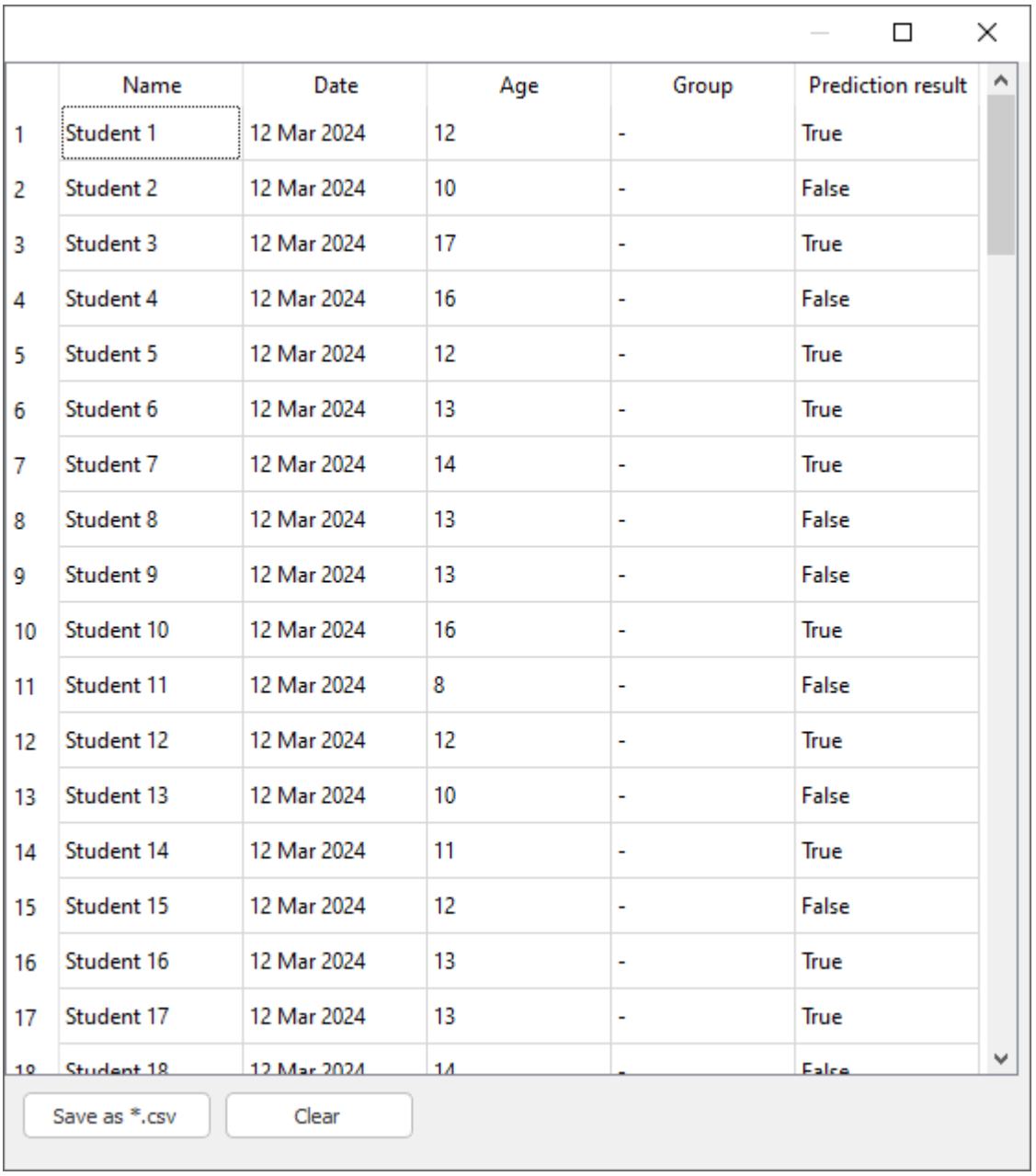


Рисунок 3.9. Таблиця результатів

Після відкриття таблиці результатів її можна зберегти в якості *.csv файлу або очистити, натиснувши на відповідні кнопки.

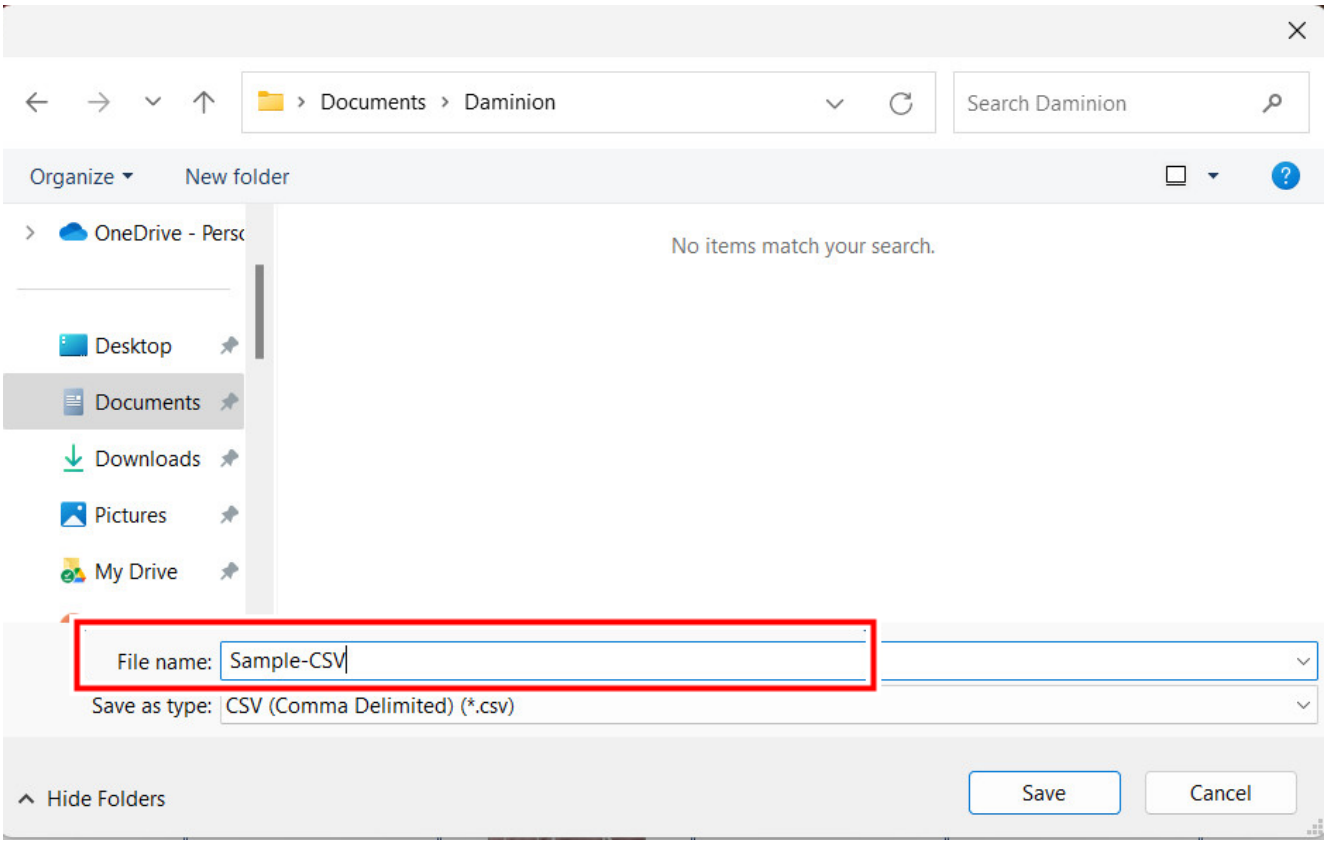


Рисунок 3.10. Вікно збереження таблиці в *.csv форматі

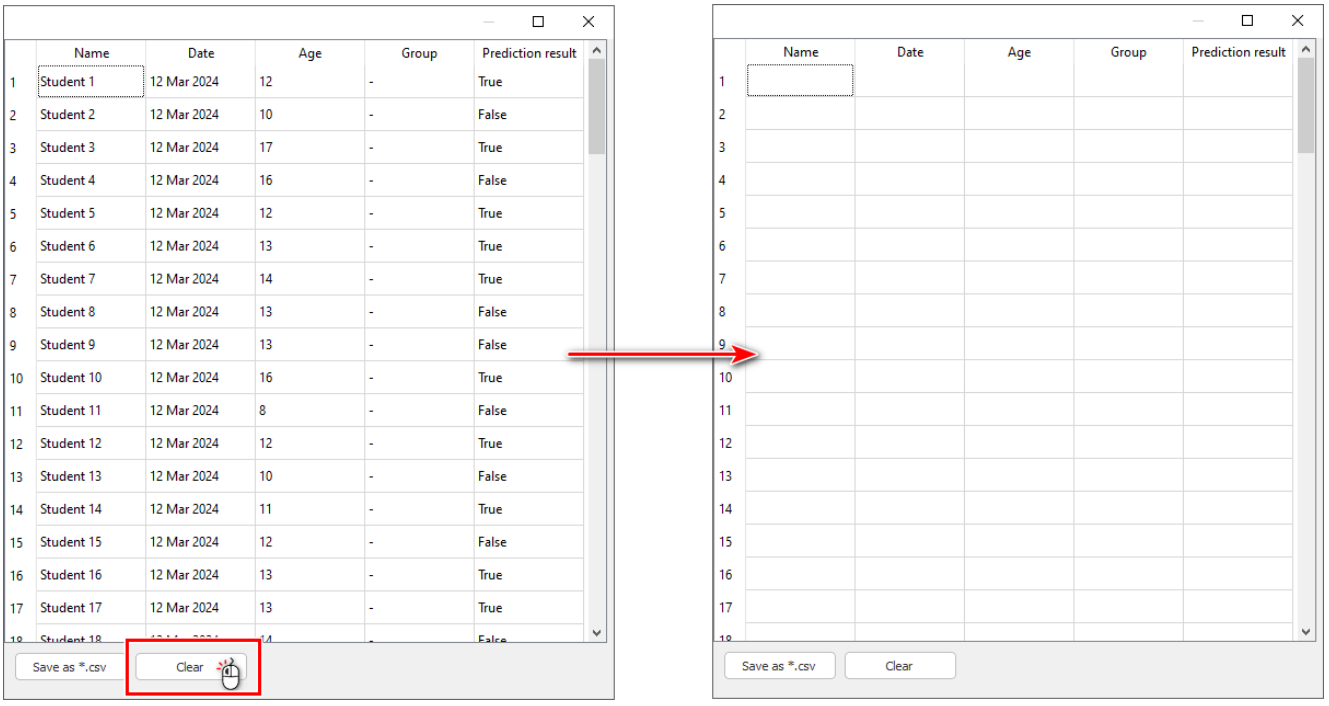


Рисунок 3.11. Очищення таблиці

Третя частина (3 на рисунку 3.2) вирішує проблему, описану в розділі 2.1.2 Вирішення задачі утримання студента на курсі програмування.

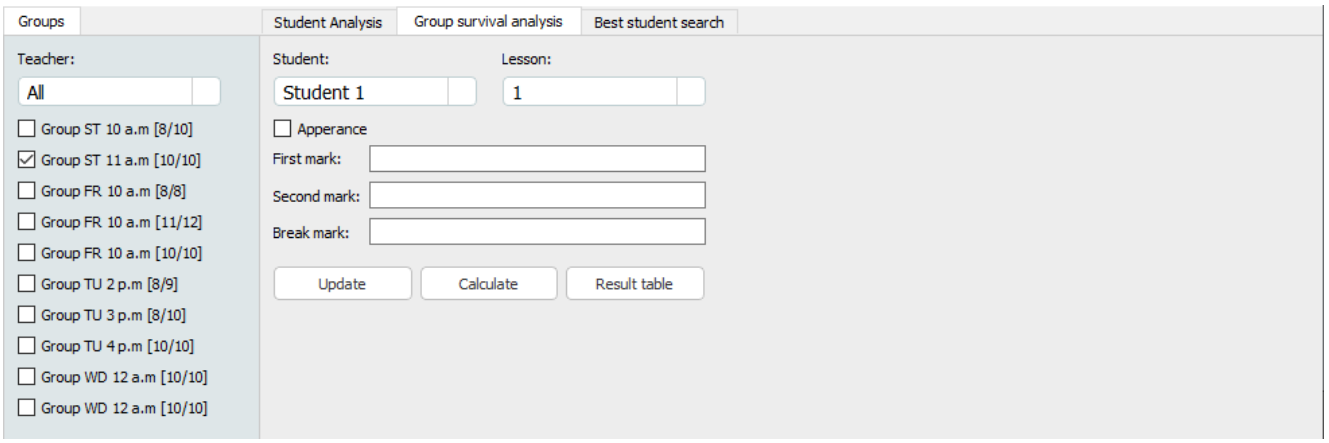


Рисунок 3.12. Вікно «Аналіз виживання груп»

Дане вікно відрізняється від попереднього, оскільки більша його частина призначена для заповнення даних про пройдені студентом уроки, а менша – для створення прогнозу та його перегляду.

Для початку роботи необхідно обрати групу з списку груп справа та почати заповнювати інформацію про студента. Якщо групу не буде обрано – параметри вікна «Аналіз виживання груп» будуть недоступними до натискання.

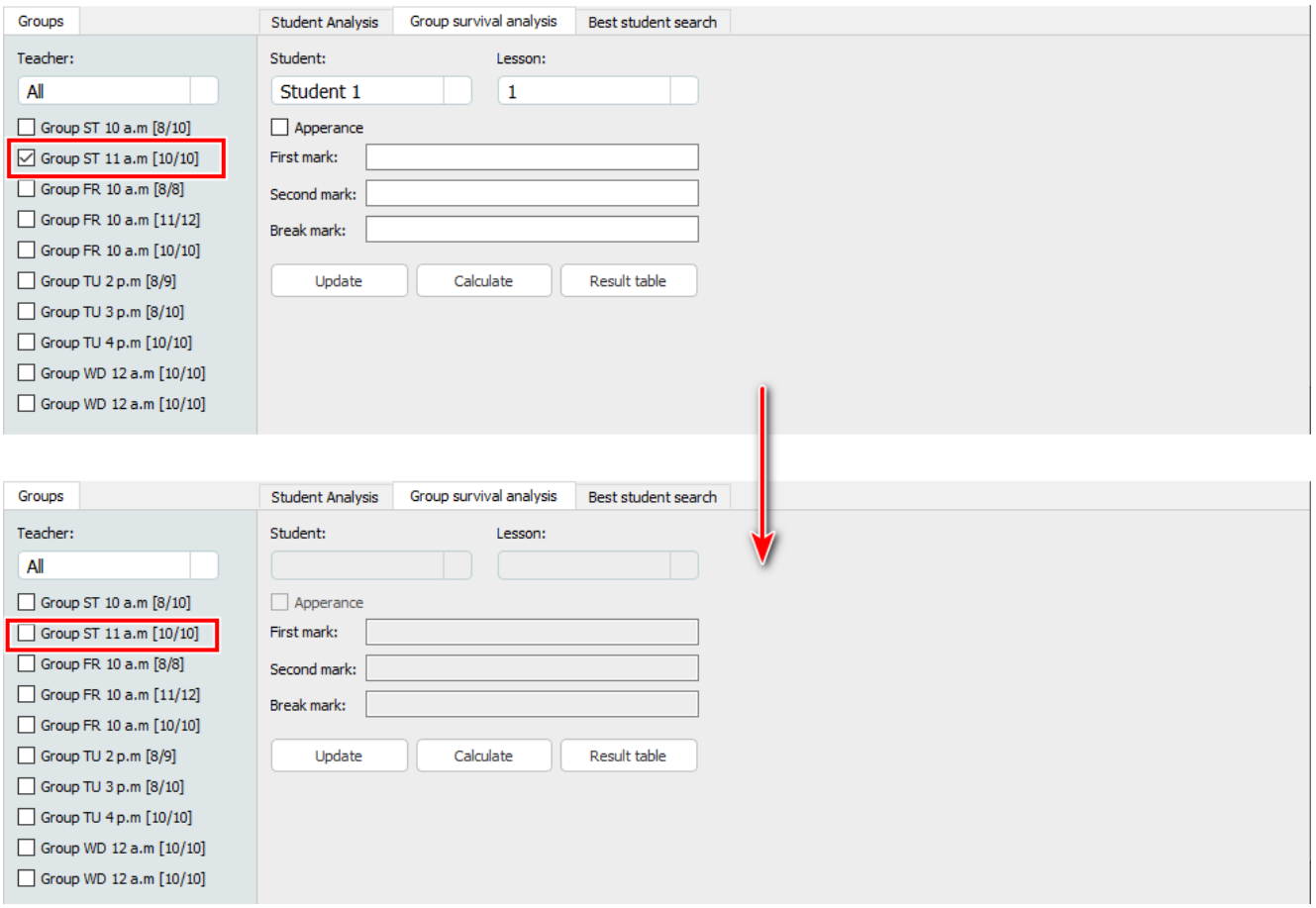


Рисунок 3.13. Блокування параметрів при деактивації прапорців груп

Після вибору групи необхідно внести оцінки для кожного студента відповідно до завершеного уроку:

- Перша оцінка. Відповідає за оцінку роботи студента на першій, теоретичній, половині уроку.
- Друга оцінка. Відповідає за оцінку роботи студента на другій, практичній, половині уроку.
- Оцінка за перерву. Відповідає за оцінку залученості студента на перерві.

Після заповнення необхідно натиснути на кнопку «Оновити» (виділено червоним на рисунку 3.14). Після чого, інформація буде передано до відповідної таблиці, а параметри будуть очищені.

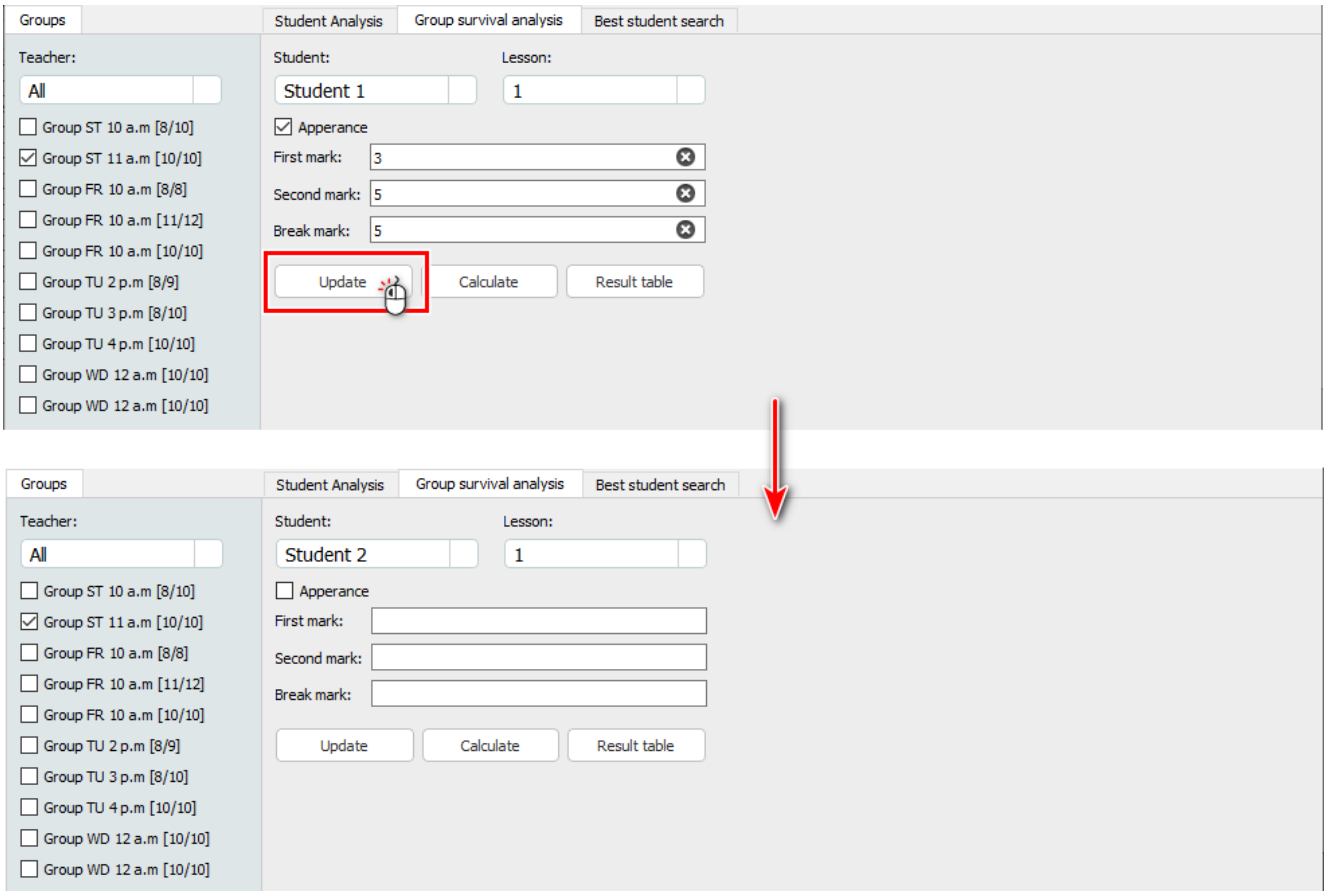


Рисунок 3.14. Очищення параметрів після заповнення

Якщо користувач заповнить одного і того ж студента двічі буде збережено тільки ті дані, що були введені останніми.

Після введення інформації про всіх студентів можна перейти до виконання прогнозу. Для цього, необхідно натиснути на кнопку «Розрахувати». Після завершення розрахунку буде виведено інформаційне повідомлення.

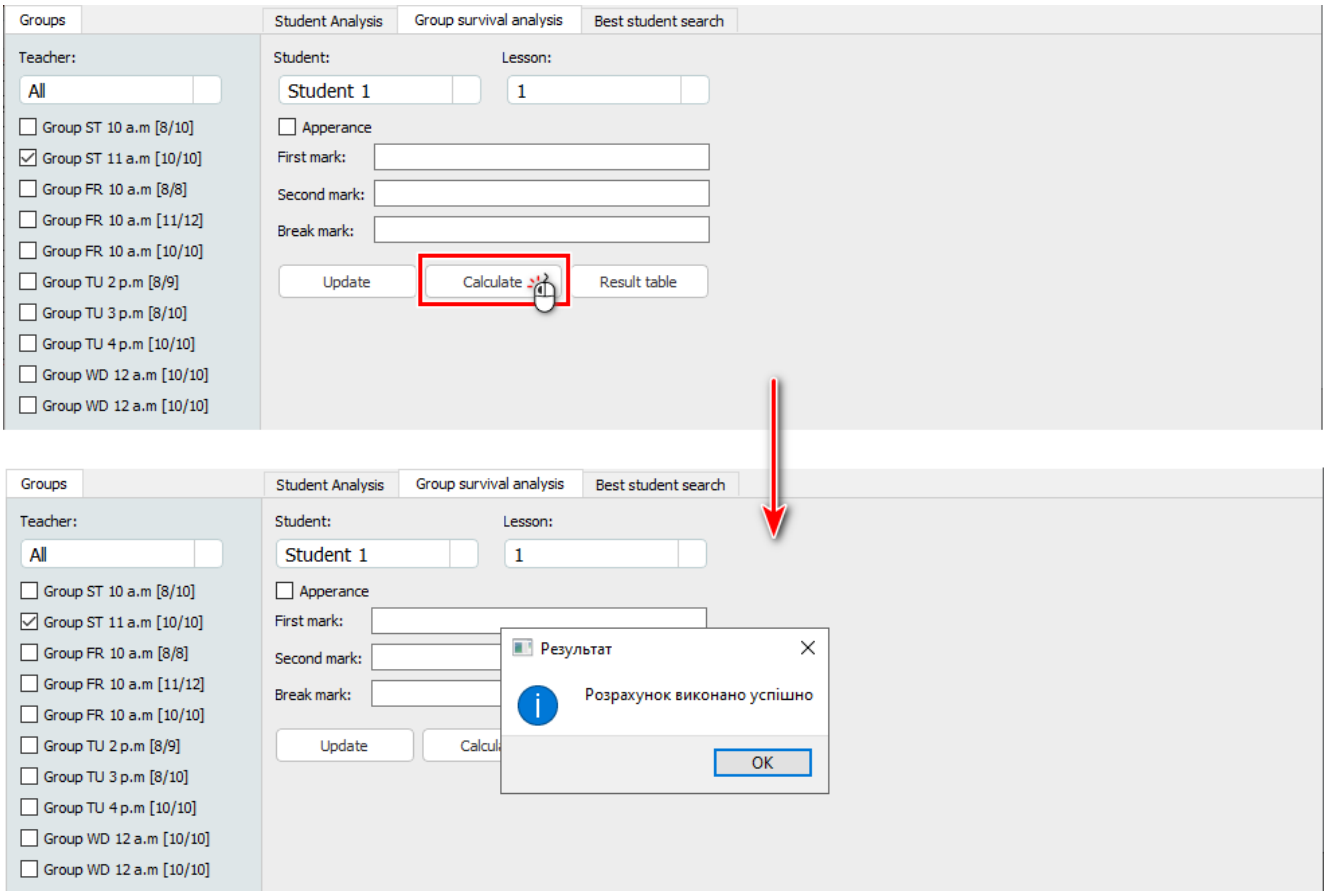


Рисунок 3.15. Виконання розрахунку вікні «Аналіз виживання груп»

Для перегляду результатів прогнозування необхідно натиснути на кнопку «Таблиця результатів» , в результаті чого буде відкрите окреме вікно з результатами, зображене на рисунку 3.16.

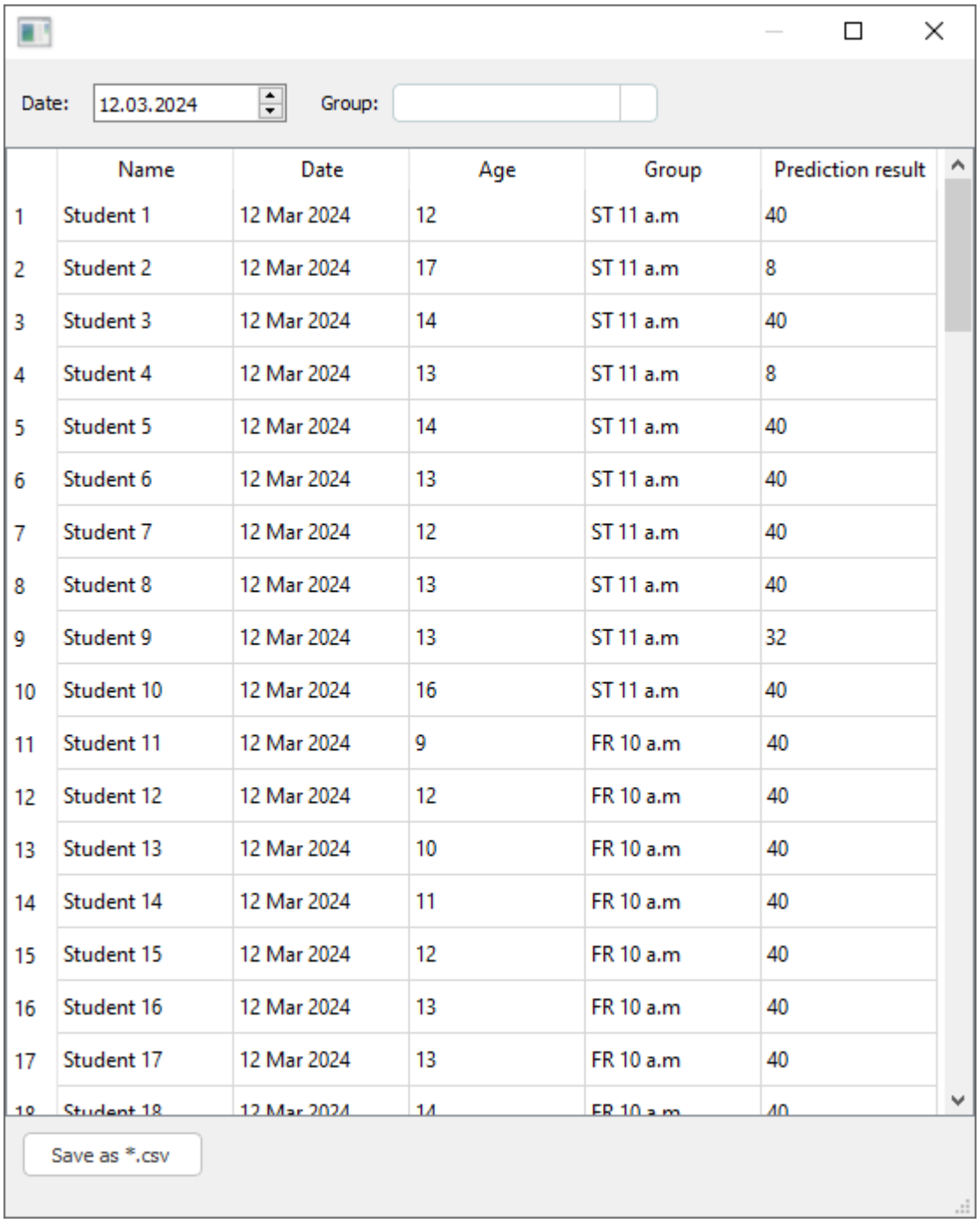


Рисунок 3.16. Таблиця перегляду результатів

В цілому, вікно таблиці результатів за логікою роботи аналогічне до таблиці результатів вкладки «Аналіз студента». Основна різниця криється в наповненні таблиці та присутності фільтрів.

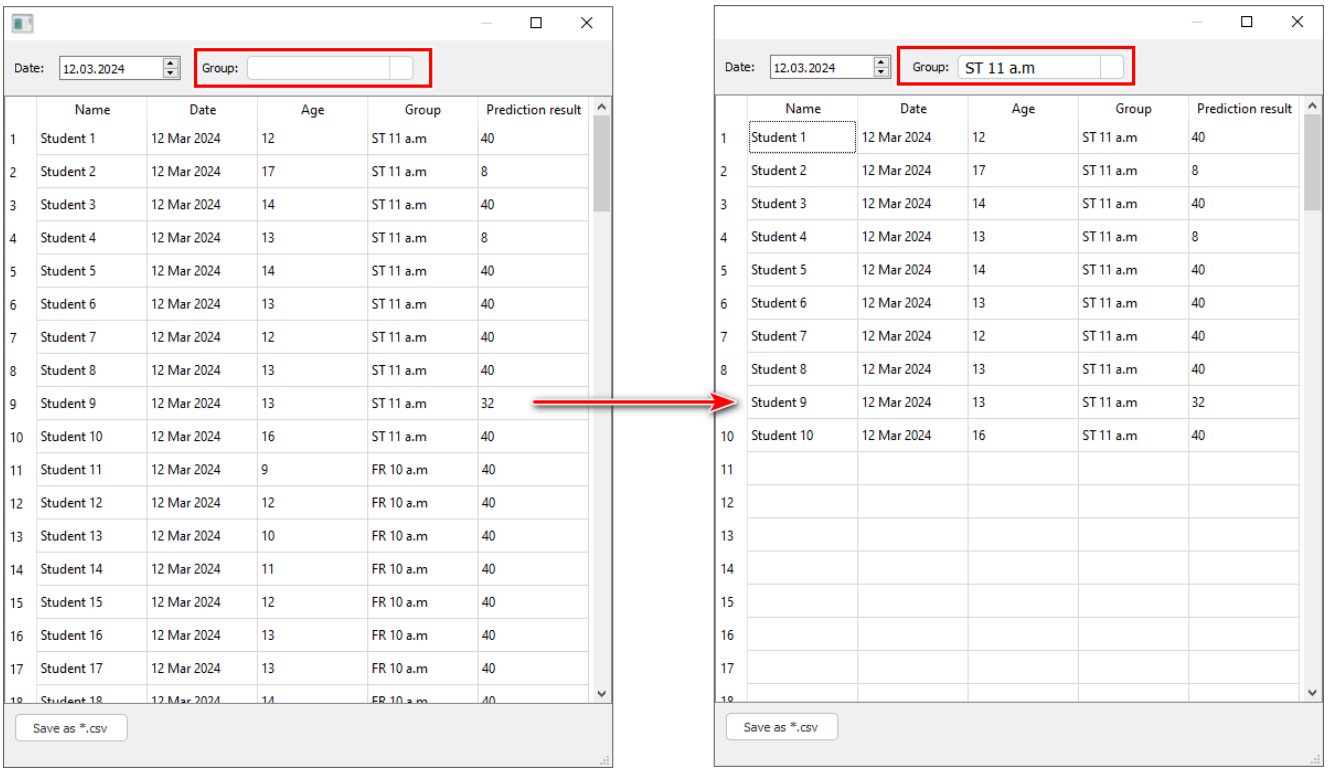


Рисунок 3.17. Приклад роботи фільтрації вікна результатів

Збереження таблиці результатів в *.csv файл аналогічне до попередньо описаного.

Четверта і остання частина (4 на рисунку 3.2) вирішує проблему, описану в розділі 2.1.3 Вирішення задачі пошуку найкращих студентів для проходження курсу програмування.

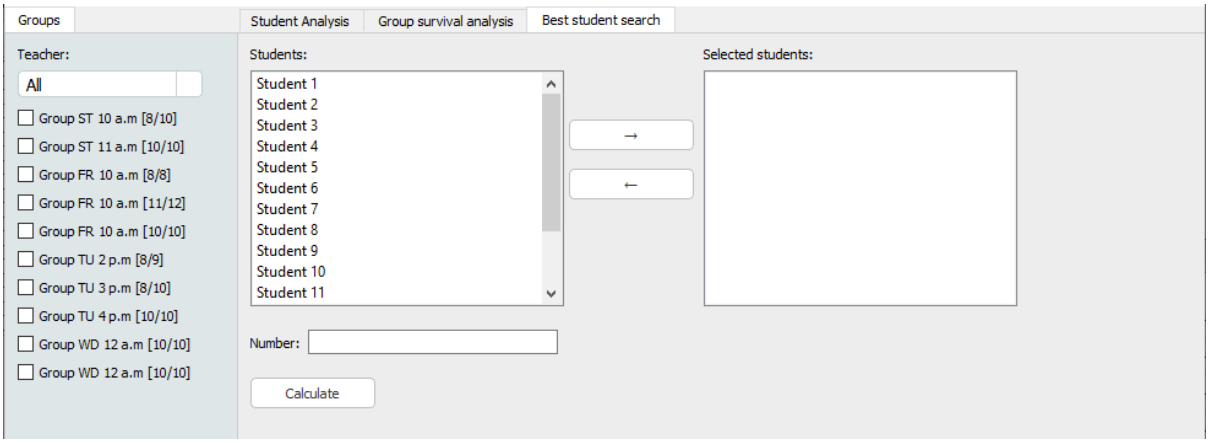


Рисунок 3.18. Вікно «Аналіз виживання груп»

Для початку роботи, необхідно обрати студентів, яких потрібно порівняти. Для цього, їх потрібно перетягнути з одного списку «Студенти» до списку «Обрані студенти» за допомогою кнопок «→» і «←».

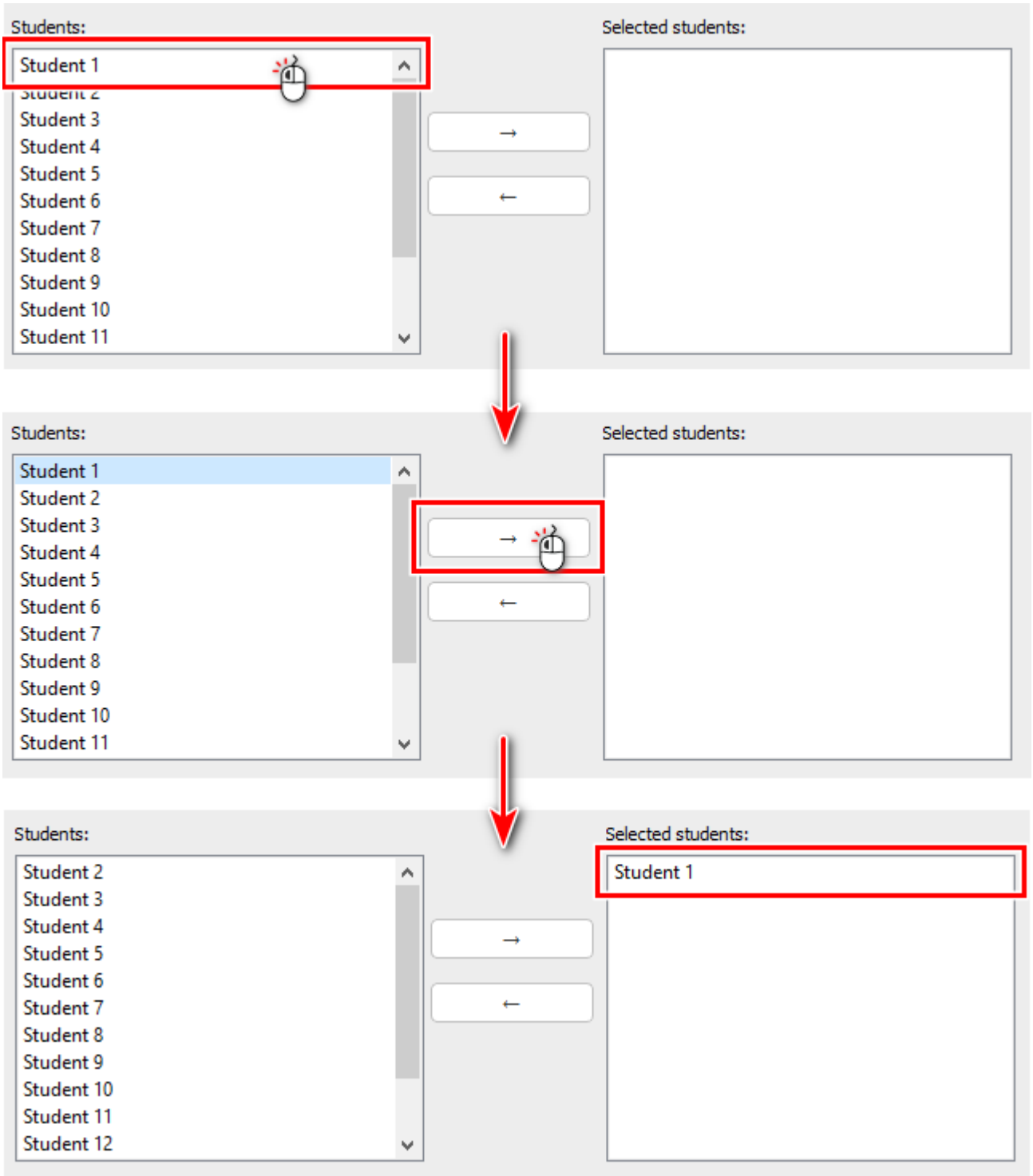


Рисунок 3.19. Додавання студенту до списку обраних студентів

Список «Студенти» наповнюється відповідно до обраних груп в боковому меню.

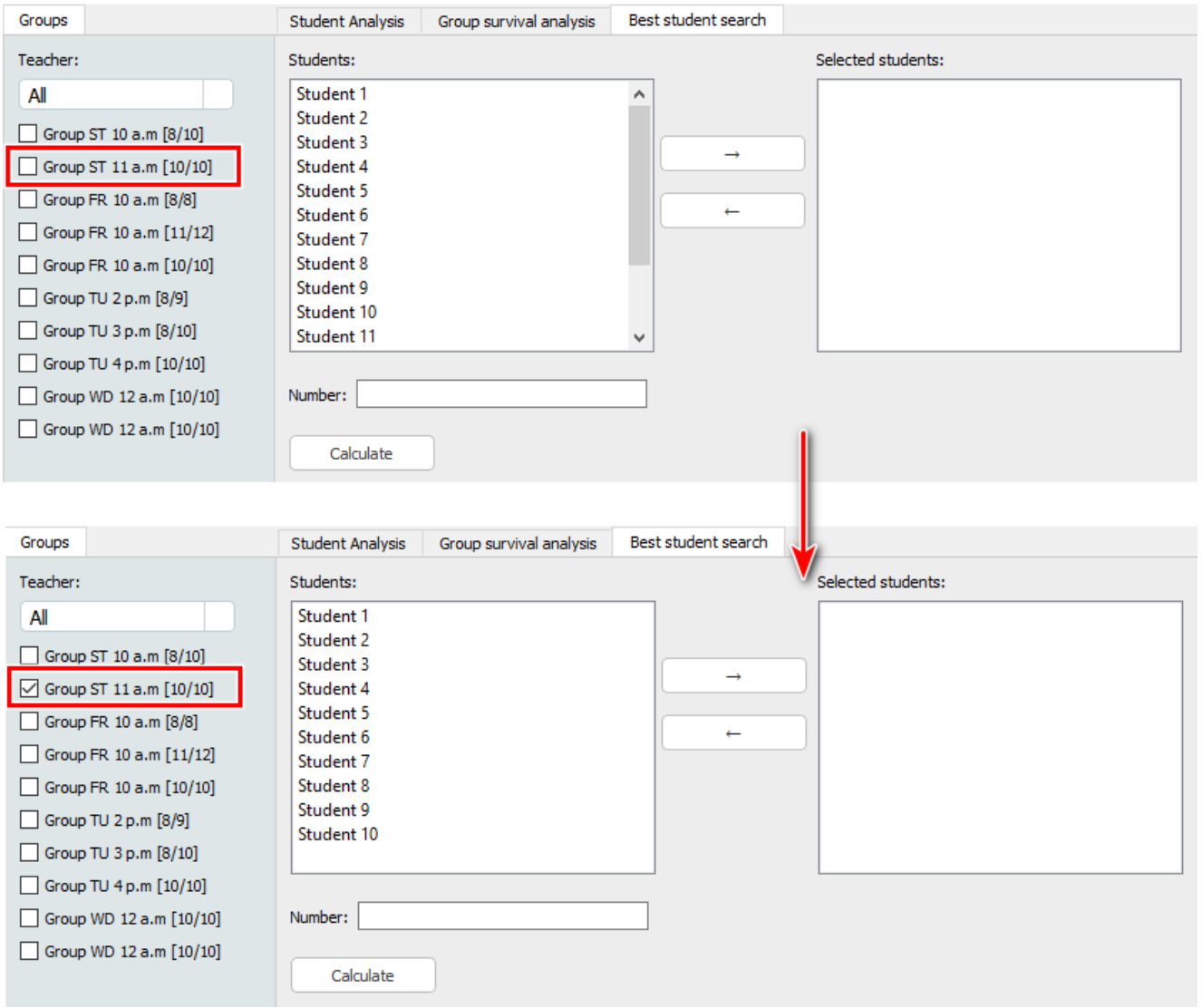


Рисунок 3.20. Зміна списку студентів відповідно обраних груп

Якщо не обрано жодної групи – в списку «Студенти» відображаються всі студенти.

Для отримання результату порівняння, необхідно обрати студентів, визначити кількість студентів, яку потрібно отримати та натиснути на кнопку «Обрахувати». Після завершення обрахунку буде виведено інформаційне вікно з результатами обрахунків.

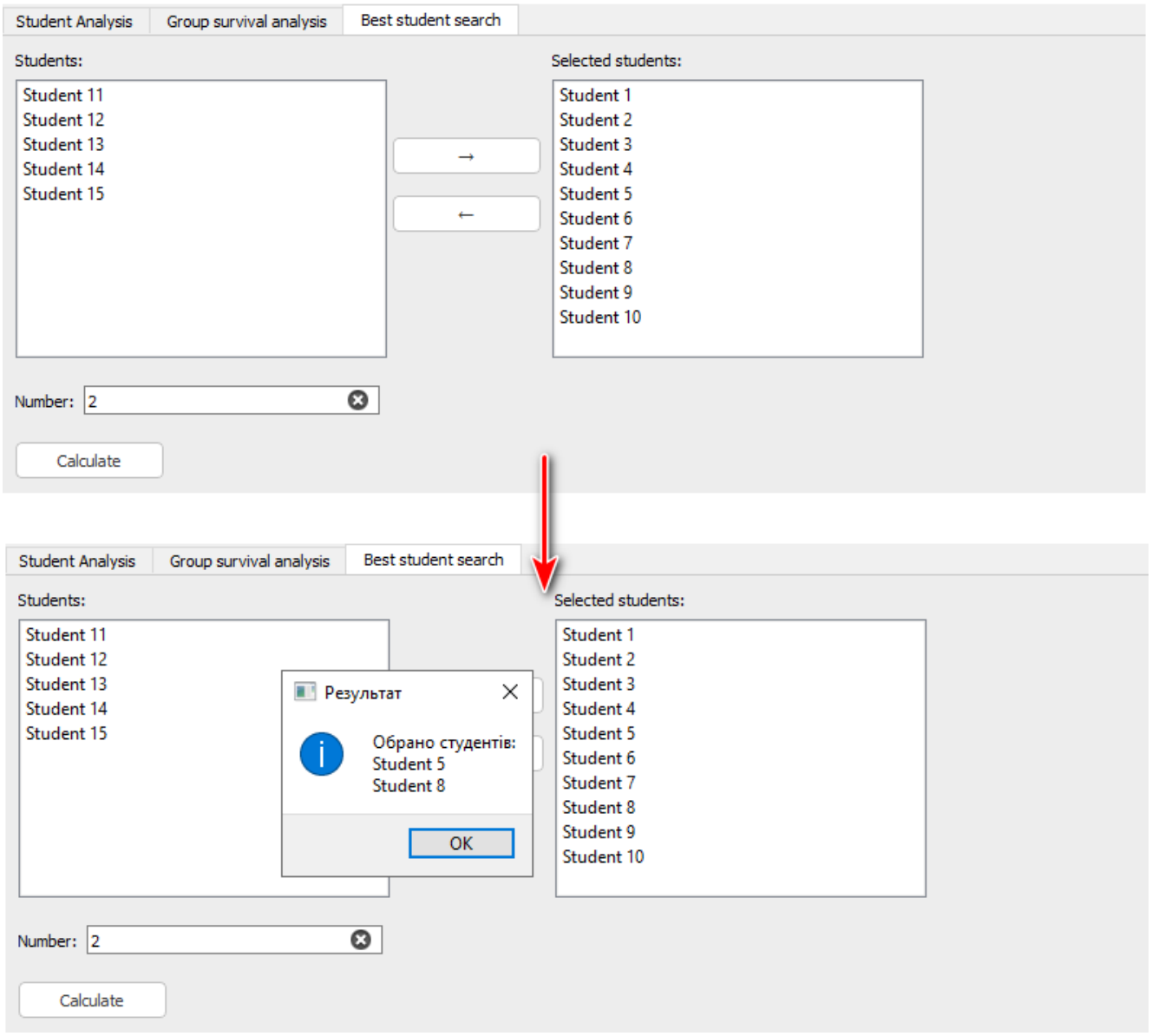


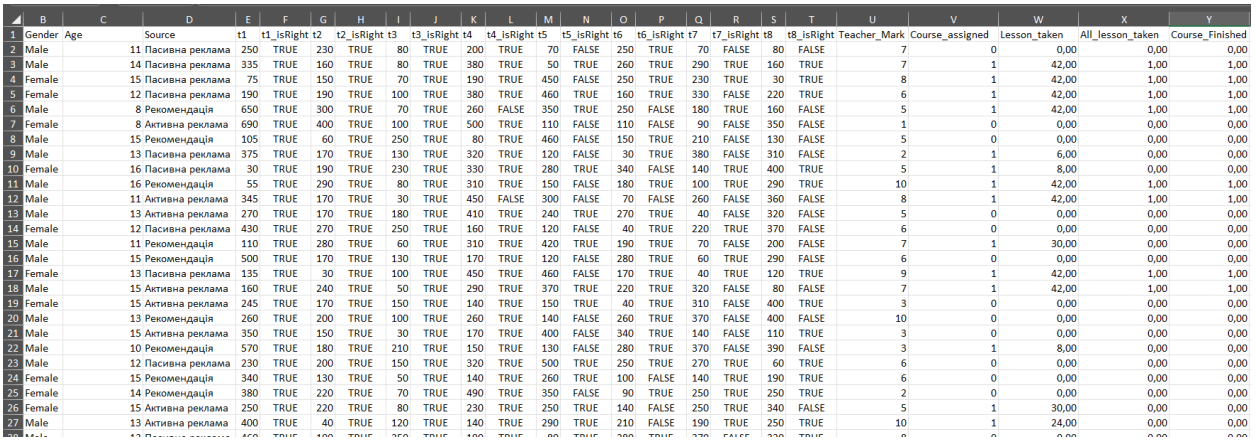
Рисунок 3.21. Результат пошуку найкращих студентів

3.2 Програмна реалізація моделей

В процесі роботи над моделями дані будуть розділені на тренувальний та тестовий набори. Тренувальний набір використовується для навчання та налаштування моделей класифікації та аналізу виживання. Тестовий набір призначений для оцінки якості моделей та перевірки їхньої здатності до узагальнення на нових даних.

Спершу, реалізуємо метод KNN. Він є найпростішим та найпершим з точки зору бізнесу. Спершу, проглянемо дані, що надані для вирішення цієї задачі.

Набір даних представляє собою набір статистики з першого, пробного уроку та дані про те, чи записався та завершив студент курс. Кожен запис у наборі даних також містить інформацію про результат навчання одного студента.



Gender	Age	Source	t1	t1_isRight	t2	t2_isRight	t3	t3_isRight	t4	t4_isRight	t5	t5_isRight	t6	t6_isRight	t7	t7_isRight	t8	t8_isRight	Teacher_Mark	Course_assigned	Lesson_taken	All_lesson_taken	Course_Finished
Male	11	Пасивна реклама	250	TRUE	230	TRUE	80	TRUE	200	TRUE	70	FALSE	250	TRUE	70	FALSE	80	FALSE	7	0	0,00	0,00	0,00
Male	14	Пасивна реклама	335	TRUE	160	TRUE	80	TRUE	380	TRUE	50	TRUE	260	TRUE	290	TRUE	160	TRUE	7	1	42,00	1,00	1,00
Female	15	Пасивна реклама	75	TRUE	150	TRUE	70	TRUE	190	TRUE	450	FALSE	250	TRUE	230	TRUE	30	TRUE	8	1	42,00	1,00	1,00
Female	12	Пасивна реклама	190	TRUE	190	TRUE	100	TRUE	380	TRUE	460	TRUE	160	TRUE	330	FALSE	220	TRUE	6	1	42,00	1,00	1,00
Male	8	Рекомендація	650	TRUE	300	TRUE	70	TRUE	260	FALSE	350	TRUE	250	FALSE	180	TRUE	160	FALSE	5	1	42,00	1,00	1,00
Female	8	Активна реклама	690	TRUE	400	TRUE	100	TRUE	500	TRUE	110	FALSE	110	FALSE	90	FALSE	350	FALSE	1	0	0,00	0,00	0,00
Male	15	Рекомендація	105	TRUE	60	TRUE	250	TRUE	80	TRUE	460	FALSE	150	TRUE	210	FALSE	130	FALSE	5	0	0,00	0,00	0,00
Male	13	Пасивна реклама	375	TRUE	170	TRUE	130	TRUE	320	TRUE	120	FALSE	30	TRUE	380	FALSE	310	FALSE	2	1	6,00	0,00	0,00
Female	16	Пасивна реклама	30	TRUE	190	TRUE	230	TRUE	330	TRUE	280	TRUE	340	FALSE	140	TRUE	400	TRUE	5	1	8,00	0,00	0,00
Male	16	Рекомендація	55	TRUE	290	TRUE	80	TRUE	310	TRUE	150	FALSE	180	TRUE	100	TRUE	290	TRUE	10	1	42,00	1,00	1,00
Male	11	Активна реклама	345	TRUE	170	TRUE	30	TRUE	450	FALSE	300	FALSE	70	FALSE	260	FALSE	360	FALSE	8	1	42,00	1,00	1,00
Male	13	Активна реклама	270	TRUE	170	TRUE	180	TRUE	410	TRUE	240	TRUE	270	TRUE	40	FALSE	320	FALSE	5	0	0,00	0,00	0,00
Female	12	Пасивна реклама	430	TRUE	270	TRUE	250	TRUE	160	TRUE	120	FALSE	40	TRUE	220	TRUE	370	FALSE	6	0	0,00	0,00	0,00
Male	11	Рекомендація	110	TRUE	280	TRUE	60	TRUE	310	TRUE	420	TRUE	190	TRUE	70	FALSE	200	FALSE	7	1	30,00	0,00	0,00
Male	15	Рекомендація	500	TRUE	170	TRUE	130	TRUE	170	TRUE	120	FALSE	280	TRUE	60	TRUE	290	FALSE	6	0	0,00	0,00	0,00
Female	13	Пасивна реклама	135	TRUE	30	TRUE	100	TRUE	450	TRUE	460	FALSE	170	TRUE	40	TRUE	120	TRUE	9	1	42,00	1,00	1,00
Male	15	Активна реклама	160	TRUE	240	TRUE	50	TRUE	290	TRUE	370	TRUE	220	TRUE	320	FALSE	80	FALSE	7	1	42,00	1,00	1,00
Female	15	Активна реклама	245	TRUE	170	TRUE	150	TRUE	140	TRUE	150	TRUE	40	TRUE	310	FALSE	400	TRUE	3	0	0,00	0,00	0,00
Male	13	Рекомендація	260	TRUE	200	TRUE	100	TRUE	260	TRUE	140	FALSE	260	TRUE	370	FALSE	400	FALSE	10	0	0,00	0,00	0,00
Male	15	Активна реклама	350	TRUE	150	TRUE	30	TRUE	170	TRUE	400	FALSE	340	TRUE	140	FALSE	110	TRUE	3	0	0,00	0,00	0,00
Male	10	Рекомендація	570	TRUE	180	TRUE	210	TRUE	150	TRUE	130	FALSE	280	TRUE	370	FALSE	390	FALSE	3	1	8,00	0,00	0,00
Male	12	Пасивна реклама	230	TRUE	200	TRUE	150	TRUE	320	TRUE	500	TRUE	250	TRUE	270	TRUE	60	TRUE	6	0	0,00	0,00	0,00
Female	15	Рекомендація	340	TRUE	130	TRUE	50	TRUE	140	TRUE	260	TRUE	100	FALSE	140	TRUE	190	TRUE	6	0	0,00	0,00	0,00
Female	14	Рекомендація	380	TRUE	220	TRUE	70	TRUE	490	TRUE	350	FALSE	90	TRUE	250	TRUE	250	TRUE	2	0	0,00	0,00	0,00
Female	15	Активна реклама	250	TRUE	220	TRUE	80	TRUE	230	TRUE	250	TRUE	140	FALSE	250	TRUE	340	FALSE	5	1	30,00	0,00	0,00
Male	13	Активна реклама	400	TRUE	40	TRUE	120	TRUE	140	TRUE	290	TRUE	210	FALSE	190	TRUE	250	TRUE	10	1	24,00	0,00	0,00

Рисунок 3.22. Дані в Excel

Набір даних містить наступні поля:

1. Student_name. Ім’я студента. Для потреб компанії ім’я було замінено. Наразі, в інтерфейсі генеруються заглушки за форматом «Student 1», «Student 2», тощо.
2. Gender. Стать студента.
3. Age. Вік студента.
4. Source. Джерело реклами, з якого студент записався до школи. Може мати чотири значення:

a. Пасивна реклама. Студент записався з пасивної реклами, наприклад з реклами на білбордах чи соціальних мережах.

b. Активна реклама. Студент записався після рекламного дзвінка оператора школи.

c. Рекомендація. Студент записався за рекомендацією учня школи.

d. Співробітник. Студента записав співробітник школи. Тобто, це дитина, особисто знайома з співробітником.

5. t1 – t8. Час проходження завданнь.

6. t1_isRight - t1_isRight. Вірність виконання завданнь.

7. Teacher_Mark. Оцінка вчителя після пробного заняття.

8. Course_assigned. Результат, чи записався студент на курс.

9. Lesson_taken. Визначає скільки уроків пройшов студент. Максимальна кількість уроків - 40.

10. All_lesson_taken. Булева зміна, що визначає, чи всі уроки були пройдені.

11. Course_Finished. Булева зміна, що визначає, чи завершив студент курс.

Фактично, поля «All_lesson_taken» і «Course_Finished» дублюють інформацію про завершення курсу, оскільки завершити курс не проходячи весь курс неможливо.

Варто зазначити, що час виконання першого завдання може бути необ’єктивним, оскільки студенти часто заходять на завдання, ще до пояснення вчителя, через що час часто може бути завищеним.

Завантажимо дані до Python та підготуємо для подальшої роботи.

	Student_name	Gender	Age	Source	t1	t1_isRight	t2	t2_isRight	t3	t3_isRight	t4	t4_isRight	t5	t5_isRight	t6	t6_isRight	t7	t7_isRight	t8	t8_isRight	Teacher_Mark	Result
0	St1	Male	11	Пасивна реклама	250	True	230	True	80	True	200	True	70	False	250	True	70	False	80	False	7	0
1	St2	Male	14	Пасивна реклама	335	True	160	True	80	True	380	True	50	True	260	True	290	True	160	True	7	1
2	St3	Female	15	Пасивна реклама	75	True	150	True	70	True	190	True	450	False	250	True	230	True	30	True	8	1
3	St4	Female	12	Пасивна реклама	190	True	190	True	100	True	380	True	460	True	160	True	330	False	220	True	6	1
4	St5	Male	8	Рекомендація	650	True	300	True	70	True	260	False	350	True	250	False	180	True	160	False	5	1
5	St6	Female	8	Активна реклама	690	True	400	True	100	True	500	True	110	False	110	False	90	False	350	False	1	0
6	St7	Male	15	Рекомендація	105	True	60	True	250	True	80	True	460	False	150	True	210	False	130	False	5	0
7	St8	Male	13	Пасивна реклама	375	True	170	True	130	True	320	True	120	False	30	True	380	False	310	False	2	1
8	St9	Female	16	Пасивна реклама	30	True	190	True	230	True	330	True	280	True	340	False	140	True	400	True	5	1
9	St10	Male	16	Рекомендація	55	True	290	True	80	True	310	True	150	False	180	True	100	True	290	True	10	1
10	St11	Male	11	Активна реклама	345	True	170	True	30	True	450	False	300	False	70	False	260	False	360	False	8	1
11	St12	Male	13	Активна реклама	270	True	170	True	180	True	410	True	240	True	270	True	40	False	320	False	5	0
12	St13	Female	12	Пасивна реклама	430	True	270	True	250	True	160	True	120	False	40	True	220	True	370	False	6	0
13	St14	Male	11	Рекомендація	110	True	280	True	60	True	310	True	420	True	190	True	70	False	200	False	7	1
14	St15	Male	15	Рекомендація	500	True	170	True	130	True	170	True	120	False	280	True	60	True	290	False	6	0
15	St16	Female	13	Пасивна реклама	135	True	30	True	100	True	450	True	460	False	170	True	40	True	120	True	9	1
16	St17	Male	15	Активна реклама	160	True	240	True	50	True	290	True	370	True	220	True	320	False	80	False	7	1

Рисунок 3.23. Загальний вигляд набору даних

Проаналізувавши дані можна прийти до висновку, що необхідно виконати наступні дії:

1. Перевірити DataFrame на порожні та некоректно заповнені дані.

2. Розбити категоріальні змінні, такі як «Gender» та «Source».
3. Видалити непотрібні стовпці.

Для перевірки на порожні та некоректно заповнені дані використаємо функцію «df.info()», що поверне основну статистику по кожному стовпцю.

```
Data columns (total 25 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Student_name      100 non-null    object
 1   Gender            100 non-null    object
 2   Age               100 non-null    int64
 3   Source            100 non-null    object
 4   t1                100 non-null    int64
 5   t1_isRight        100 non-null    bool
 6   t2                100 non-null    int64
 7   t2_isRight        100 non-null    bool
 8   t3                100 non-null    int64
 9   t3_isRight        100 non-null    bool
 10  t4                100 non-null    int64
 11  t4_isRight        100 non-null    bool
 12  t5                100 non-null    int64
 13  t5_isRight        100 non-null    bool
 14  t6                100 non-null    int64
 15  t6_isRight        100 non-null    bool
 16  t7                100 non-null    int64
 17  t7_isRight        100 non-null    bool
 18  t8                100 non-null    int64
 19  t8_isRight        100 non-null    bool
 20  Teacher_Mark      100 non-null    int64
 21  Course_assigned   100 non-null    int64
 22  Lesson_taken      100 non-null    int64
 23  All_lesson_taken  100 non-null    int64
 24  Course_Finished   100 non-null    int64
dtypes: bool(8), int64(14), object(3)
```

Рисунок 3.24. Основна статистика стовпців

Як видно з рисунку 3.24, порожні дані відсутні, а тип кожної змінної логічно визначений. Тепер, можна перейти до видалення непотрібних змінних та розбиття категоріальних змінних. Спершу, розіб'ємо категоріальні змінні:

1. Змінну «Gender» розіб'ємо на «Male» та «Female».
2. Змінну «Source» розіб'ємо на «Act_add», «Pass_add», «Recomend» та «Worker».

Також, одразу переведемо булеві змінні в числовий формат.

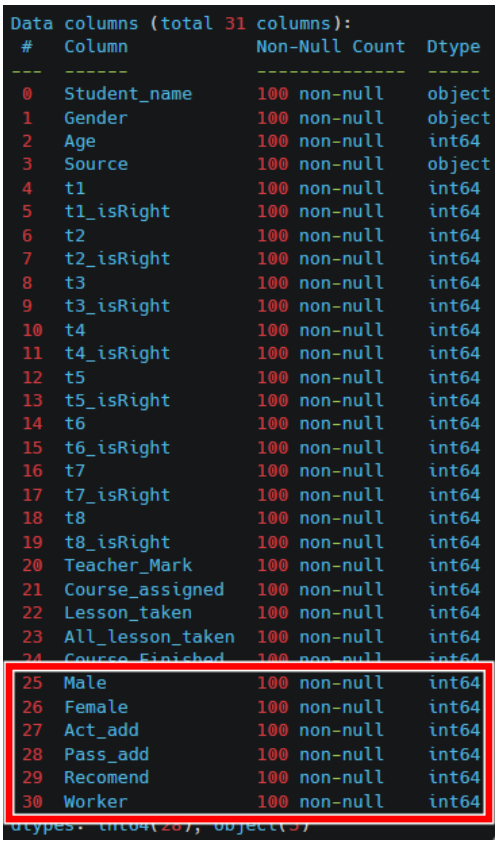


Рисунок 3.25. Розбиття категоріальних змінних

Перейдемо до видалення змінних. Змінні «Gender» та «Source» необхідно видалити, оскільки вони були розбиті, і більше не потрібні в загальному вигляді. Зміну «Student_name» необхідно видалити, оскільки вона ніяк не впливає на модель, а зміну «All_lesson_taken» - оскільки вона дублює інформацію іншого параметру.

```
Data columns (total 27 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Age              100 non-null    int64
 1   t1               100 non-null    int64
 2   t1_isRight       100 non-null    int64
 3   t2               100 non-null    int64
 4   t2_isRight       100 non-null    int64
 5   t3               100 non-null    int64
 6   t3_isRight       100 non-null    int64
 7   t4               100 non-null    int64
 8   t4_isRight       100 non-null    int64
 9   t5               100 non-null    int64
 10  t5_isRight       100 non-null    int64
 11  t6               100 non-null    int64
 12  t6_isRight       100 non-null    int64
 13  t7               100 non-null    int64
 14  t7_isRight       100 non-null    int64
 15  t8               100 non-null    int64
 16  t8_isRight       100 non-null    int64
 17  Teacher_Mark     100 non-null    int64
 18  Course_assigned  100 non-null    int64
 19  Lesson_taken     100 non-null    int64
 20  Course_Finished  100 non-null    int64
 21  Male             100 non-null    int64
 22  Female           100 non-null    int64
 23  Act_add          100 non-null    int64
 24  Pass_add         100 non-null    int64
 25  Recomend         100 non-null    int64
 26  Worker           100 non-null    int64
dtypes: int64(27)
```

Рисунок 3.26. Видалення непотрібних змінних з набору даних

Також, нормалізуємо дані для побудови коректної моделі, оскільки KNN є чутливою до нормалізації.

	Age	t1	t1_isRight	t2	t2_isRight	t3	t3_isRight	t4	t4_isRight	t5	t5_isRight	t6	t6_isRight	t7
0	0.3	0.333333	1	0.540541	1	0.172414	1	0.361702	1	0.044444	0	0.68750	1	0.108108
1	0.6	0.462121	1	0.351351	1	0.172414	1	0.744681	1	0.000000	1	0.71875	1	0.702703
2	0.7	0.068182	1	0.324324	1	0.137931	1	0.340426	1	0.888889	0	0.68750	1	0.540541
3	0.4	0.242424	1	0.432432	1	0.241379	1	0.744681	1	0.911111	1	0.40625	1	0.810811
4	0.0	0.939394	1	0.729730	1	0.137931	1	0.489362	0	0.666667	1	0.68750	0	0.405405

Рисунок 3.27. Нормалізація даних

Для побудови якісної KNN моделі необхідно правильно визначити не тільки набір параметрів, а й кількість сусідів k. Визначимо точність моделі для різних значень k.

Таблиця 2. Точність KNN моделі в залежності від параметру k

Значення параметру k	**Точність**
2	0.6
3	0.7
4	0.6
5	0.85
6 - 7	0.75
8	0.8
9	0.9

Хоча й найбільша точність досягається при значенні параметру k = 9, великі значення k можуть призвести до недонавчання. Через це, я б зупинив вибір значення k на 5, що визначає 85% вірності передбачення.

Також, для методів кластеризації можна побудувати матрицю помилок.

Таблиця 3. Матриця помилок

	Прогнозоване позитивне	Прогнозоване негативне
Дійсно позитивне	10	1
Дійсно негативне	4	5

З цієї таблиці чітко видно, що найбільша кількість помилок відбувається при прогнозуванні позитивного як негативного.

Перед тим, як переходити до моделі Кокса можна перевірити дані за допомогою кривих Каплан-Мейєр. Також, варто зазначити, що модель даних була розширена відповідно до нових параметрів наступними полями:

1. First_mark_mean. Відповідає за середнє значення оцінки роботи студента на першій, теоретичній, половині уроку.
2. Second_mark_mean Відповідає за середнє значення оцінки роботи студента на другій, практичній, половині уроку.
3. Break_mark_mean Відповідає за середнє значення оцінки залученості студента на перерві.

Спершу, реалізуємо один з найпростіших методів – метод Каплан-Мейєр. Він є непараметричним, тому реалізація та інтерпретація результатів буде не складним. В загальному, результат відсотку проходження курсу зображений на рисунку нижче.

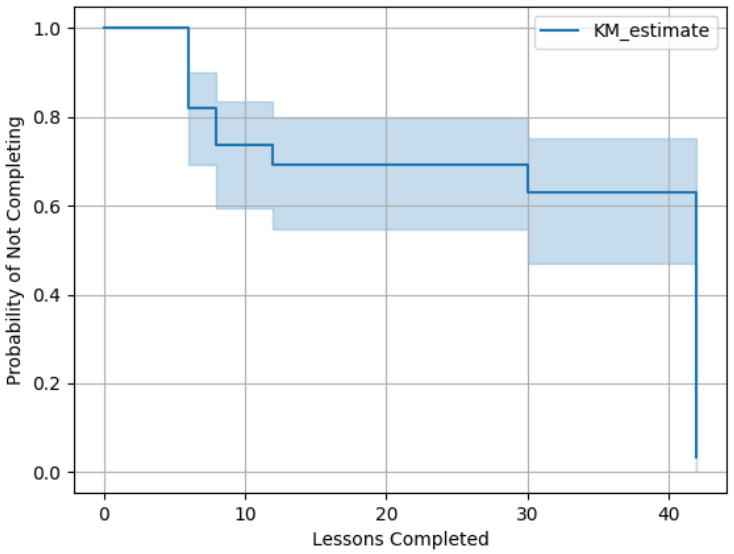


Рисунок 3.27. Крива Каплан-Мейєр

Також, побудуємо криві Каплан-Мейєр для груп студентів. Криві будуть побудовані груп, що розділені за:

- Статтю.
- Віком. Проте, буде виведено тільки дві групи: перша група – до 13 років, а друга група – від 13 років.
- Оцінкою вчителя. Суть розподілу аналогічна до розподілу за віком: перша група до 6 балів, а інша – від 7.

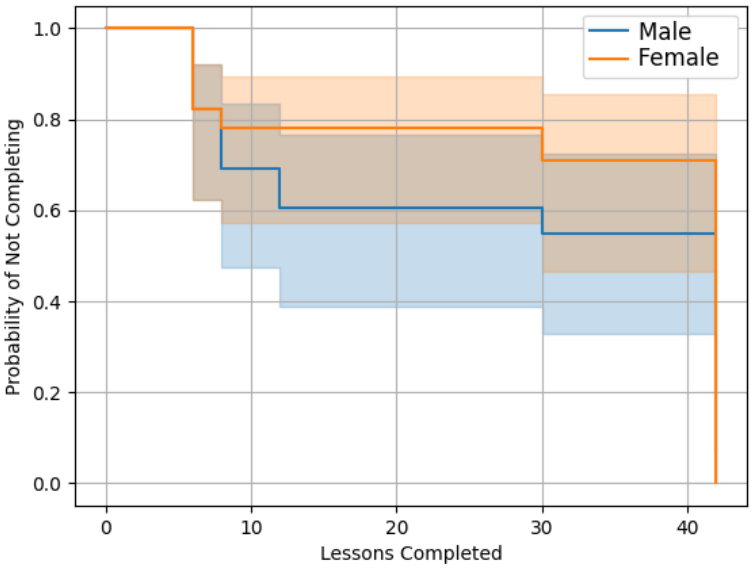


Рисунок 3.28. Крива Каплан-Мейєр, що розділена на групи за статтю

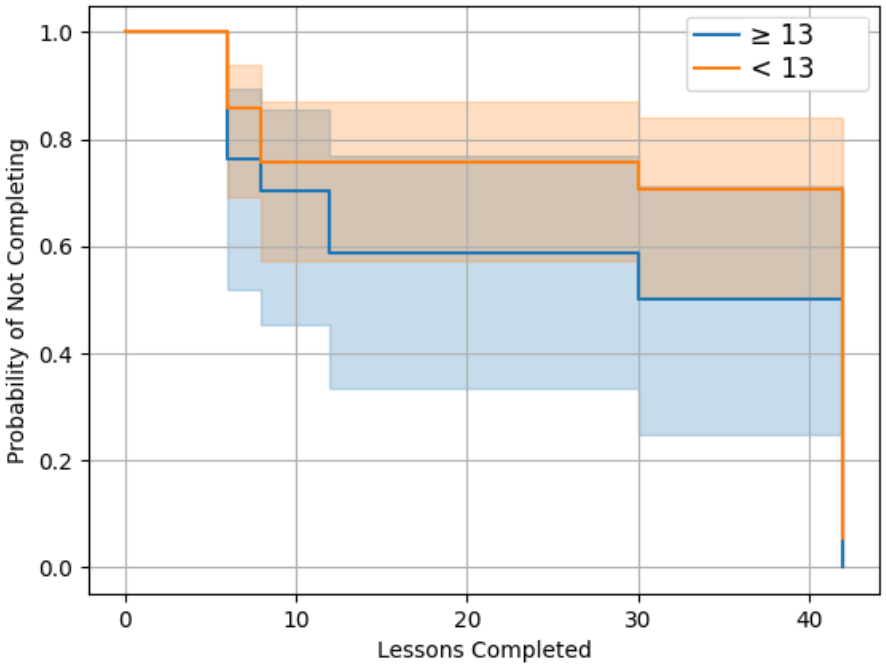


Рисунок 3.29. Крива Каплан-Мейєр, що розділена на групи за віком

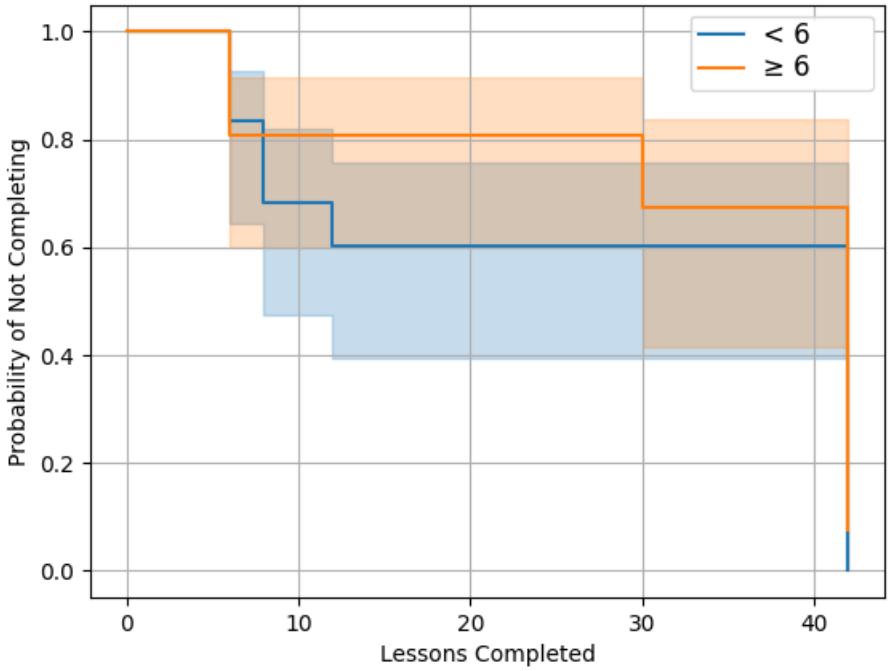


Рисунок 3.30. Крива Каплан-Мейєр, що розділена на групи за оцінкою вчителя

Тепер, перейдемо до побудови моделі Кокса. Для цього, використаємо бібліотеку «lifelines», а саме клас CoxPHFitter. Спершу, побудуємо таблицю

коефіцієнтів дисперсії інфляції між всіма змінними набору даних та цільової змінної.

	feature	VIF
0	Age	1.382429
1	Male	inf
2	Female	inf
3	Teacher_Mark	1.578326
4	Course_assigned	5.033808
5	t1	1.201048
6	t2	1.354058
7	t3	1.180121
8	t4	1.289448
9	t5	1.194524
10	t6	1.199509
11	t7	1.238756
12	t8	1.302225
13	t1_isRight	0.000000
14	t2_isRight	0.000000
15	t3_isRight	0.000000
16	t4_isRight	1.252885
17	t5_isRight	1.240732
18	t6_isRight	1.460453
19	t7_isRight	1.384153
20	t8_isRight	1.379583
21	Lesson_taken	22.959161
22	Course_Finished	13.207738

Рисунок 3.31 Таблиця коефіцієнтів дисперсії інфляції

Чітко видно, що змінні «Male» та «Female» мають нескінченний коефіцієнт, через що їх потрібно видалити. Також, видалимо змінні «t1_isRight», «t2_isRight» та «t3_isRight», оскільки вони мають нульовий коефіцієнт. Тепер побудуємо модель та навчимо на даних.

covariate	coef	exp(coef)	se(coef)	coef lower 95%	coef upper 95%	exp(coef) lower 95%	exp(coef) upper 95%
Age	-0.06	0.94	0.05	-0.16	0.05	0.85	1.05
Teacher_Mark	-0.17	0.84	0.04	-0.26	-0.09	0.77	0.92
t1	0.00	1.00	0.00	-0.00	0.00	1.00	1.00
t2	-0.00	1.00	0.00	-0.00	0.00	1.00	1.00
t3	-0.00	1.00	0.00	-0.00	0.00	1.00	1.00
t4	0.00	1.00	0.00	-0.00	0.00	1.00	1.00
t5	-0.00	1.00	0.00	-0.00	0.00	1.00	1.00
t6	0.00	1.00	0.00	-0.00	0.00	1.00	1.00
t7	-0.00	1.00	0.00	-0.00	0.00	1.00	1.00
t8	0.00	1.00	0.00	-0.00	0.00	1.00	1.00
t4_isRight	0.03	1.03	0.27	-0.50	0.56	0.61	1.76
t5_isRight	0.11	1.12	0.24	-0.35	0.58	0.70	1.78
t6_isRight	0.16	1.17	0.27	-0.37	0.69	0.69	1.99
t7_isRight	-0.19	0.83	0.25	-0.67	0.30	0.51	1.34
t8_isRight	0.01	1.01	0.26	-0.50	0.53	0.60	1.70

Рисунок 3.31 Таблиця коефіцієнтів впливу змінних на завершення курсу

Якщо коефіцієнт > 0, то змінна збільшує ризик "вибуття" студента. Якщо ж коефіцієнт < 0 – змінна зменшує ризик. Візуалізуємо таблицю коефіцієнтів моделі Кокса.

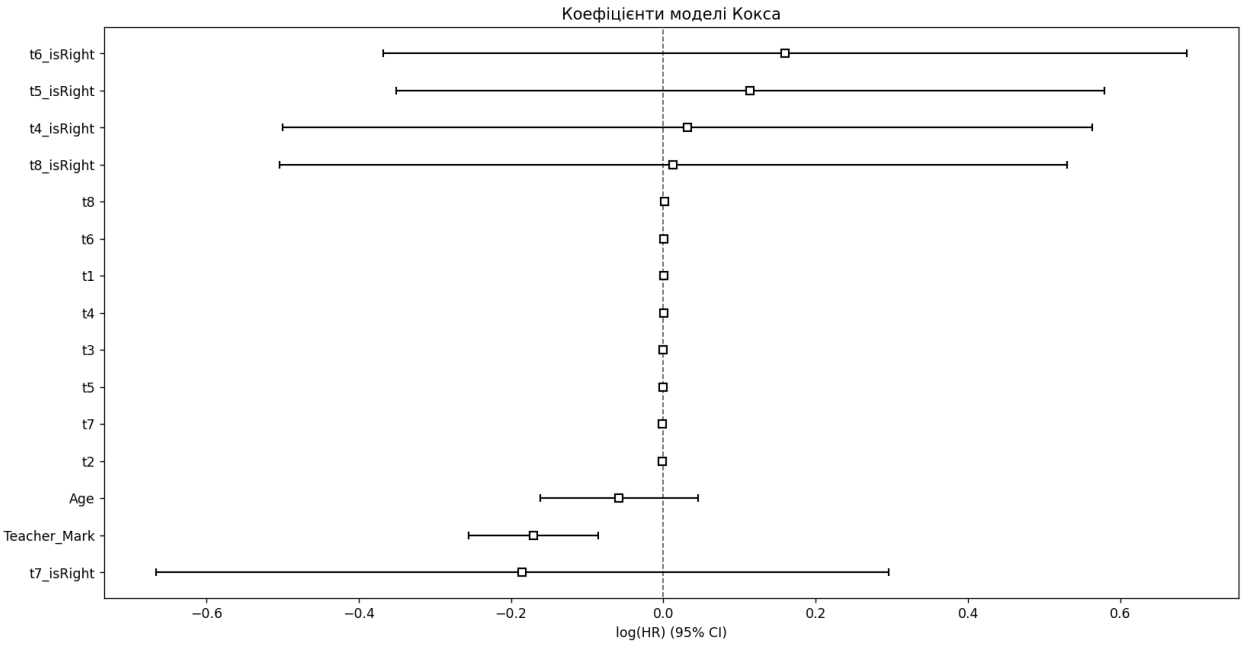


Рисунок 3.32. Візуалізація коефіцієнтів моделі Кокса

Перевіримо точність на тестовій частині вибірки.

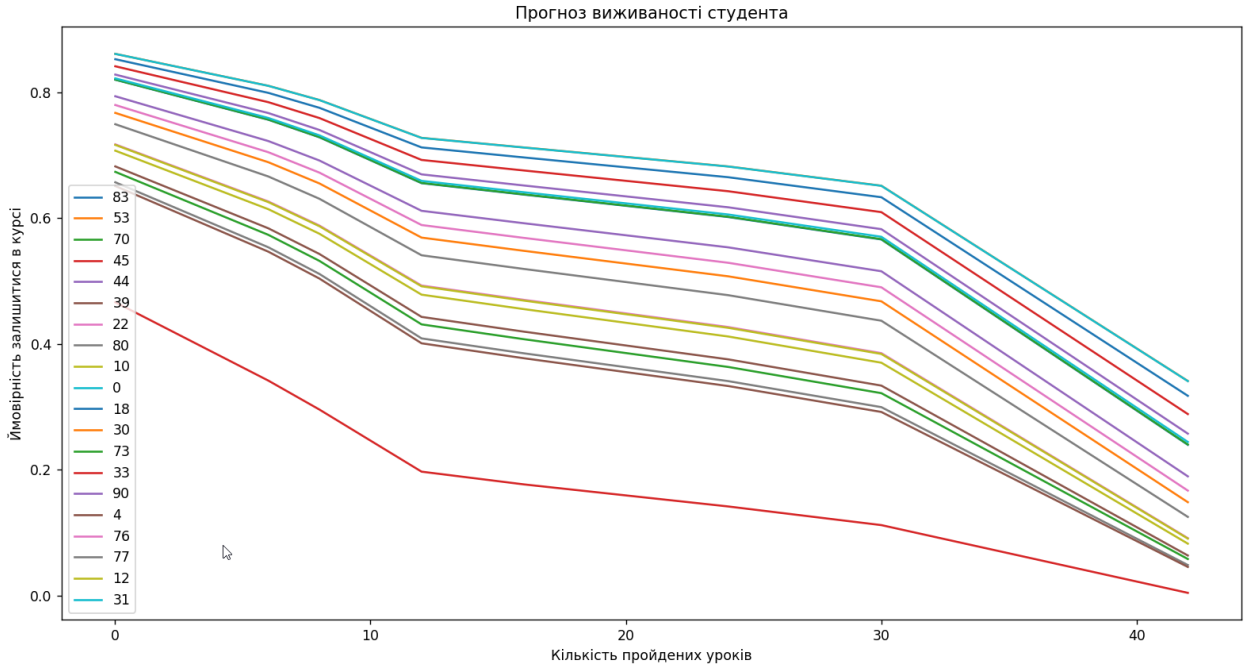


Рисунок 3.33. Результат прогнозування тестової вибірки за моделлю Кокса

За графіком видно, що студенти перестають відвідувати курс на 16 і 32 уроках. І дійсно, це вірно, оскільки, щоб пройти 16 і 32 уроки необхідно створити великий проєкт, що є перепоною для більшості студентів.

Перейдемо до тренування RankNet. В якості основної метрики буде взято оцінку вчителя (Teacher_Mark). Натренуємо модель.

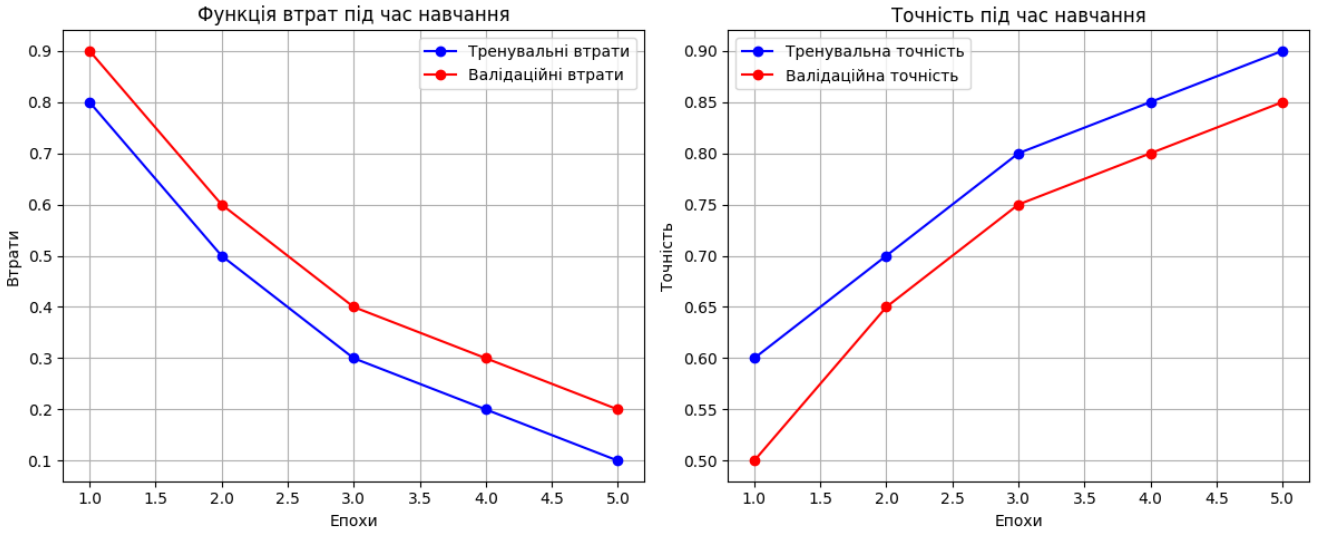


Рисунок 3.34. Тренування RankNet моделі

Для знаходження декількох студентів робота моделі просто викликається відповідну кількість разів.

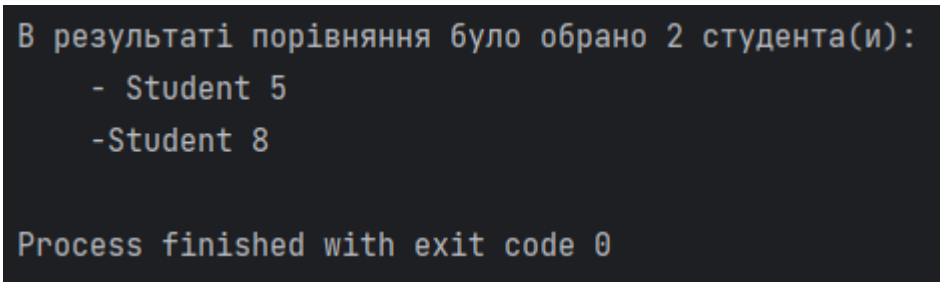


Рисунок 3.35. Результат роботи RankNet моделі

3.3 Висновки до третього розділу

У третьому розділі було практично реалізовано усі ключові етапи розробки десктопного додатку, призначеного для прогнозування успішності завершення курсу студентом школи програмування. З урахуванням рішень, обґрунтованих у попередніх розділах, було створено функціональний інтерфейс на основі бібліотеки «PyQt6», який забезпечує зручну взаємодію користувача із програмою.

Було реалізовано всі заплановані моделі — K-Nearest Neighbors, модель Кокса, а також нейронну мережу RankNet. Для цього використовувалися відповідні бібліотеки «scikit-learn», «lifelines» та «tensorflow», що дозволило досягнути гнучкості у підходах до моделювання та забезпечити надійні результати на основі реальних навчальних даних. Крім того, були імплементовані засоби попередньої обробки даних, візуалізації та інтеграції з реляційною базою даних MySQL.

ВИСНОВКИ

В межах дипломної роботи розроблено програмний продукт, що дозволяє прогнозувати ймовірність завершення навчання студентами школи програмування. Було вирішено три головні проблеми:

1. Виявлення студентів, що запишуться на курс після пробного заняття. Створено окрему сторінку «Аналіз студента» в настільному додатку, що реалізує класичний алгоритм K найближчих сусідів для вирішення проблеми.
2. Виявлення студентів, що готові відмовитись від подальшого проходження курсу. Проблему було вирішено, побудувавши модель Кокса. Ініціалізацію розрахунку за даною моделлю було винесено в окреме вікно «Аналіз виживання груп» настільного додатку.
3. Виявлення найкращих студентів для залучення на наступний курс. Для вирішення проблеми було створено модель RankNet, ініціалізація якої можлива з вкладки «Пошук кращого студента» настільного додатку.

В ході виконання задачі було:

- Проведено аналіз проблеми, обрано оптимальні технології.
- Реалізовано настільний додаток із підтримкою моделей машинного навчання та зручним інтерфейсом.

Результати мають практичну цінність для освітніх установ та можуть бути впроваджені для підвищення ефективності навчального процесу.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Hanushek E. A. Assessing the effects of school resources on student performance: An update //Educational evaluation and policy analysis. – 1997. – Т. 19. – №. 2. – С. 141-164.

2. Robinson V. M. J., Lloyd C. A., Rowe K. J. The impact of leadership on student outcomes: An analysis of the differential effects of leadership types //Educational administration quarterly. – 2008. – Т. 44. – №. 5. – С. 635-674.

3. Shiffrin R. M., Dumais S. T. The development of automatism //Cognitive skills and their acquisition. – Psychology Press, 2013. – С. 111-140.

4. Castells M. et al. Information technology, globalization and social development. – Geneva : UNRISD, 1999. – Т. 114.

5. Bocij P., Greasley A., Hickie S. Business information systems: Technology, development and management. – Pearson education, 2008.

6. Schwalbe K. Introduction to project management. – Boston : Course Technology Cengage Learning, 2009.

7. Bonchi F. et al. Social network analysis and mining for business applications //ACM Transactions on Intelligent Systems and Technology (TIST). – 2011. – Т. 2. – №. 3. – С. 1-37.

8. Hendricks K. B., Singhal V. R., Stratman J. K. The impact of enterprise systems on corporate performance: A study of ERP, SCM, and CRM system implementations //Journal of operations management. – 2007. – Т. 25. – №. 1. – С. 65-82.

9. Kankanhalli A. et al. An integrative study of information systems security effectiveness //International journal of information management. – 2003. – Т. 23. – №. 2. – С. 139-154.

10. Sarker I. H., Apu K. Mvc architecture driven design and implementation of java framework for developing desktop application //International Journal of Hybrid Information Technology. – 2014. – Т. 7. – №. 5. – С. 317-322.

11. Grossberg S., Grossberg S. Classical and instrumental learning by neural networks //Studies of Mind and Brain: Neural Principles of Learning, Perception, Development, Cognition, and Motor Control. – 1982. – С. 65-156.

12. Medsker L. R. et al. Recurrent neural networks //Design and Applications. – 2001. – Т. 5. – №. 64-67. – С. 2.

13. Jenkins S. P. Survival analysis //Unpublished manuscript, Institute for Social and Economic Research, University of Essex, Colchester, UK. – 2005. – Т. 42. – С. 54-56.

14. Guo G. et al. KNN model-based approach in classification //On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings. – Springer Berlin Heidelberg, 2003. – С. 986-996.

15. Rigatti S. J. Random forest //Journal of Insurance Medicine. – 2017. – Т. 47. – №. 1. – С. 31-39.

16. Bland J. M., Altman D. G. Survival probabilities (the Kaplan-Meier method) //Bmj. – 1998. – Т. 317. – №. 7172. – С. 1572-1580.

17. Therneau T. M. et al. The cox model. – Springer New York, 2000. – С. 39-77.

18. Song Y., Wang H., He X. Adapting deep ranknet for personalized search //Proceedings of the 7th ACM international conference on Web search and data mining. – 2014. – С. 83-92.

19. Summerfield M. Rapid GUI programming with Python and Qt: the definitive guide to PyQt programming. – Pearson Education, 2007.

20. DuBois P. MySQL. – Addison-Wesley, 2013.

21. McKinney W. et al. pandas: a foundational Python library for data analysis and statistics //Python for high performance and scientific computing. – 2011. – Т. 14. – №. 9. – С. 1-9.

Додаток А

Програмна реалізація очистки набору даних.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from lifelines import KaplanMeierFitter

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', None)

df = pd.read_excel('dataset.xlsx')
# print(df)
#df.info()
df['Male'] = (df['Gender'] == 'Male').astype(int)
df['Female'] = (df['Gender'] == 'Female').astype(int)

df['Act_add'] = (df['Source'] == 'Активна реклама').astype(int)
df['Pass_add'] = (df['Source'] == 'Пасивна реклама').astype(int)
df['Recomend'] = (df['Source'] == 'Рекомендація').astype(int)
df['Worker'] = (df['Source'] == 'Працівник').astype(int)

df.drop(['Gender', 'Source', "Student_name", "All_lesson_taken"], inplace=True, axis=True)

bool_columns = ['t1_isRight', 't2_isRight', 't3_isRight', 't4_isRight', 't5_isRight', 't6_isRight',
't7_isRight', 't8_isRight']
df[bool_columns] = df[bool_columns].astype(int)

print(df.info())

columns_to_normalize = ['Age', 't1', 't2', 't3', 't4', 't5', 't6', 't7', 't8', 'Teacher_Mark', 'Lesson_taken']
scaler = MinMaxScaler()
df[columns_to_normalize] = scaler.fit_transform(df[columns_to_normalize])
```