Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний

інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Катедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 6 з дисципліни

«Прикладні задачі машинного навчання»

«Прикладна задача машинного навчання»

Виконав студент ІП-11 Лесів Владислав Ігорович

Перевірив Нестерук Андрій Олександрович

Київ 2023

**Лабораторна робота 6**

**Прикладна задача машинного навчання**

**Постановка завдання.**

Створити, навчити й апробувати багатошарову нейронну мережу з прямою передачею сигналу для ухвалення рішення про зарахування до Університету абітурієнтів, які здали вступні іспити з математики, англійської та української мови.

Правила прийому наступні:

1. Рейтинг абітурієнтів формується за формулою , де - бал з іспиту з математики, - бал з іспиту з англійської мови, - бал з іспиту з української мови;

2. Мінімальний прохідний бал на вступ 160 для абітурієнтів без пільг;

3. З математики для абітурієнтів без пільг мінімальний бал іспиту не може бути менший 140 балів;

4. Абітурієнти, які мають пільги, зараховуються при мінімумі 120 балів з усіх іспитів і їх рейтинг не може бути меншим ніж 144 бали;

5. Університет може прийняти на навчання 350 абітурієнтів, з них не більше 10% це абітурієнти з пільгами;

6. Статистика минулих років показує, що в середньому до Університету подають документи 1500 абітурієнтів.

Для навчання мережі слід використовувати всі вивчені методи адаптації та навчання та провести аналіз їх ефективності. Слід також визначити мінімальну кількість шарів і нейронів, що забезпечує задовільне рішення поставленої задачі. У звіті навести архітектуру мережі та код реалізації.

Результат має бути візуалізований та представлений у формі Еxcel таблиці, як список зарахованих абітурієнтів.

**Хід роботи.**

1. Генерація балів з іспитів.

Для створення даних, з якими я буду працювати, використовую модуль random. Я обрав таку схему: створюю випадкове значення балу з іспиту з математики від 120 до 200 балів. Далі на основі цього значення створюю бали з англійської та української мови: якщо іспит з математики складено на менш ніж 140 балів, то решта іспитів генеруються від 120 до 160. Якщо значення математики від 140 до 170 балів, то решта іспитів будуть від 140 до 180. І якщо перший іспит вище за 170, то і решта іспитів будуть вищі за 170. Така схема дозволяє створити адекватні значення іспитів, коли для однієї людини вони коливаються, але не настільки разюче. Створюю з 6000 згенерованих «вступників» датафрейм.

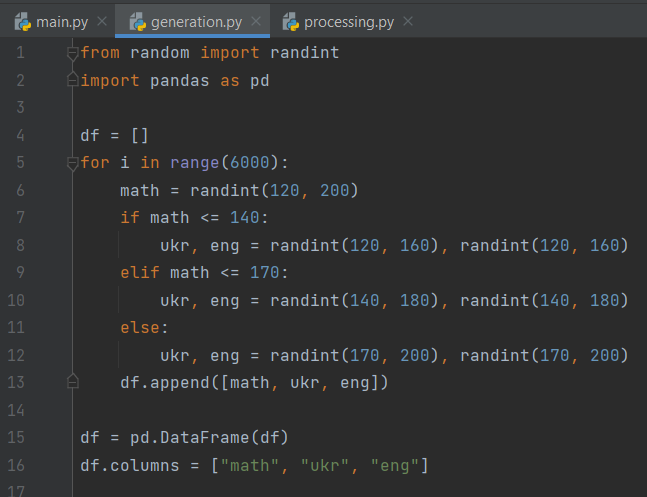


Рисунок 1 – Генерація результатів іспитів

Далі створюю колонку рейтингу, значення якої розраховую за даною формулою за допомогою результатів іспитів. Також створюю колонку наявности пільги і випадково розподіляю пільгу кожному дев’ятому вступнику. Врешті отриманий датафрейм завантажую в csv-файл.

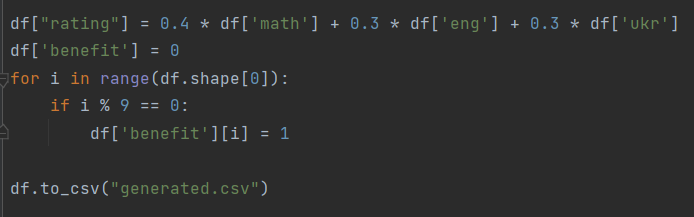


Рисунок 2 – Код доопрацьованого набору даних

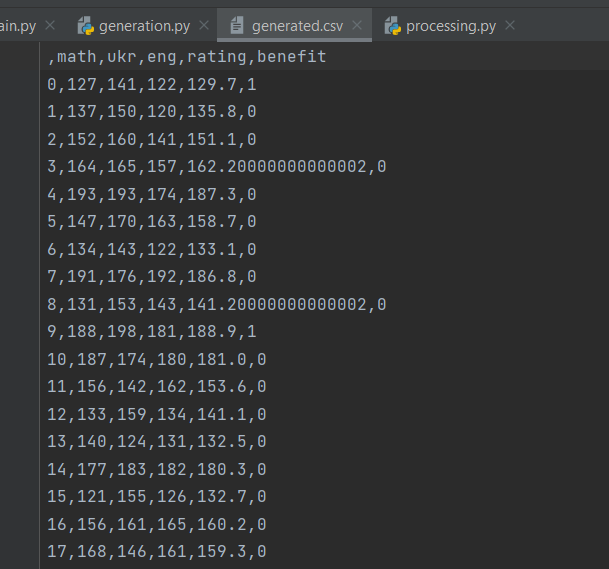


Рисунок 3 – Набір даних у форматі csv-файлу

1. Опрацьовую набір даних для розбивки його на списки абітурієнтів.

Завантажую попередньо створений файл і розбиваю його на 4 датафрейми по 1500 абітурієнтів, як було вказано в умові. Далі маю функцію process. Створюю для датафрейму колонку результату, де було 0 або 1 залежно від того, зарахований абітурієнт чи ні. Для початку задаю усім значення 2.

Спочатку розглядаю пільговиків. Відповідно якщо у такої людини усі іспити складено більш ніж на 120, а рейтинг більший за 144, то додаємо його у список можливих зарахувань. Якщо ж ні, одразу відсіюю, у результаті ставлю 0.

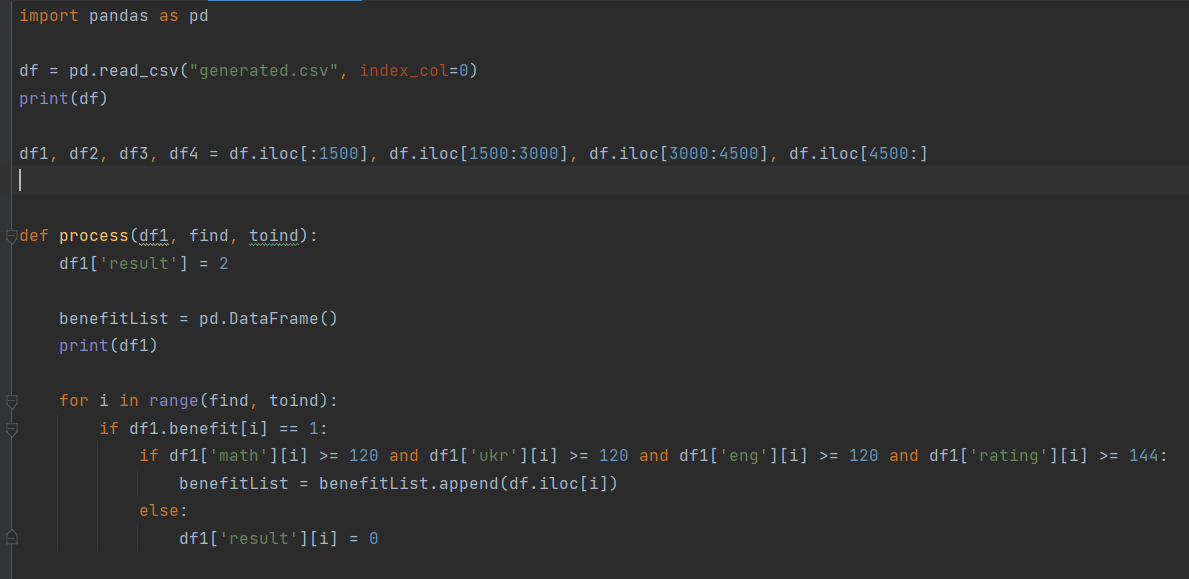


Рисунок 4 – Розбивка даних і попереднє відсіювання пільговиків

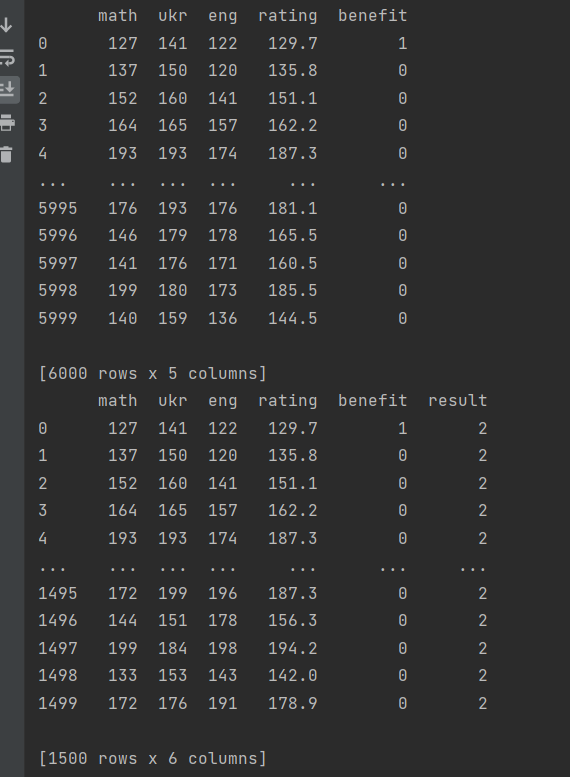


Рисунок 5 – Початковий набір і його частинка, яку починаємо доопрацьовувати

Далі сортую можливих вступників-пільговиків за спаданням і рахую їх кількість. Ми можемо взяти 10% від 350, тобто 35 пільговиків. Тож беремо 35 найкращих або ж усіх, якщо їх менше за 35 – відповідно в результаті вони мають 1.



Рисунок 6 – Знаходження вступників-пільговиків

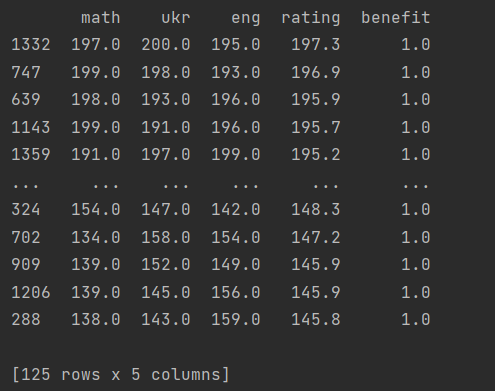


Рисунок 7 – Відсортовані за рейтингом пільговики

Далі визначаю кількість вільних місць після прийняття вступників. Тоді усім відсіяним пільговикам ставлю результат 0, а для звичайних абітурієнтів повторюю дії, схожі до пророблених раніше. Тільки тут іспит з математики має бути більшим за 140, а загальний рейтинг більше за 160. Усім відсіяним абітурієнтам в результаті 0, прийнятим – 1.

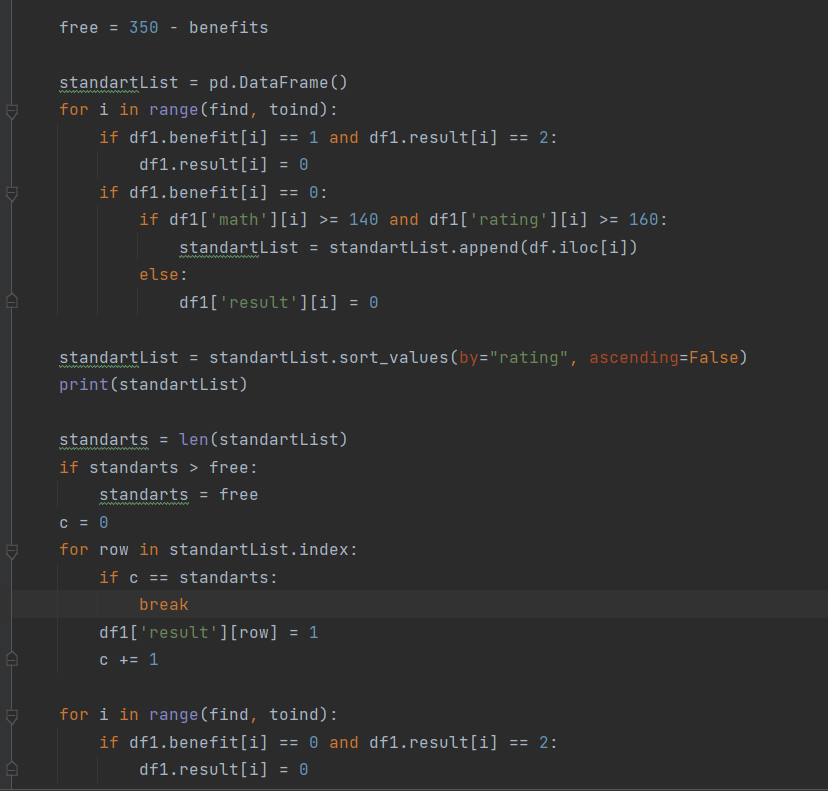


Рисунок 8 – Знаходження звичайних вступників

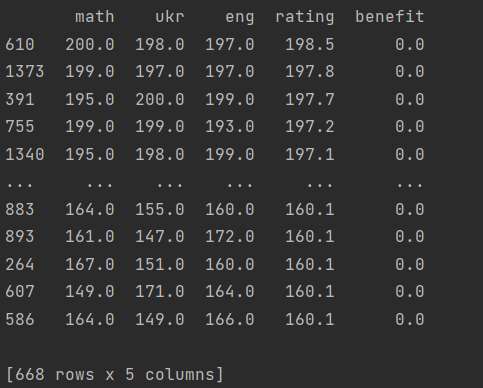


Рисунок 9 – Відсортовані звичайні абітурієнти

Щоб переконатися, що прийнято не більше абітурієнтів, ніж можливо, порахуємо кількість прийнятих у датасеті. А сам датасет запишемо в окремий файл. Такі дії проробимо для 4 вибірок.

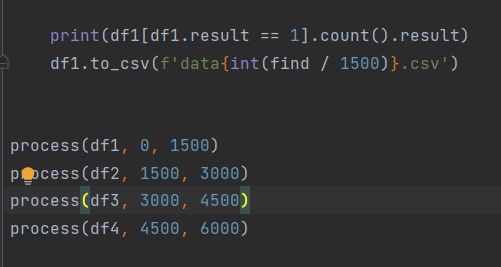


Рисунок 10 – Запис результатів у файл і виклик функції

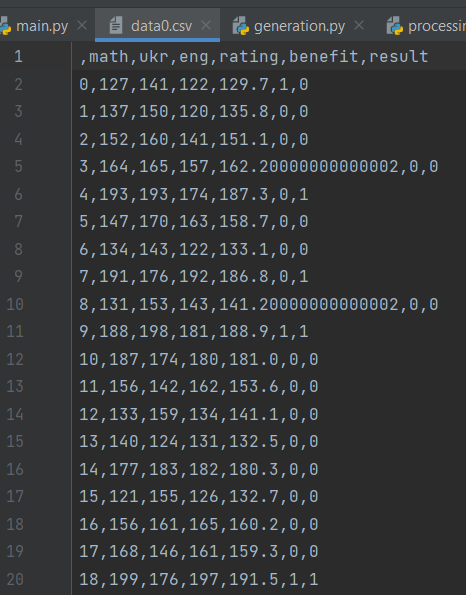


Рисунок 11 – Результат виконання функції для однієї вибірки

1. Побудова нейронної мережі.

Для нормалізації даних і відповідно кращої збіжності у мережі створюю функцію transform. Для X обираю усі колонки, крім останньої, а результат y – відповідно остання колонка. І для X ділю значення на 200, адже 200 – максимум, тому отримуємо значення від 0 до 1.

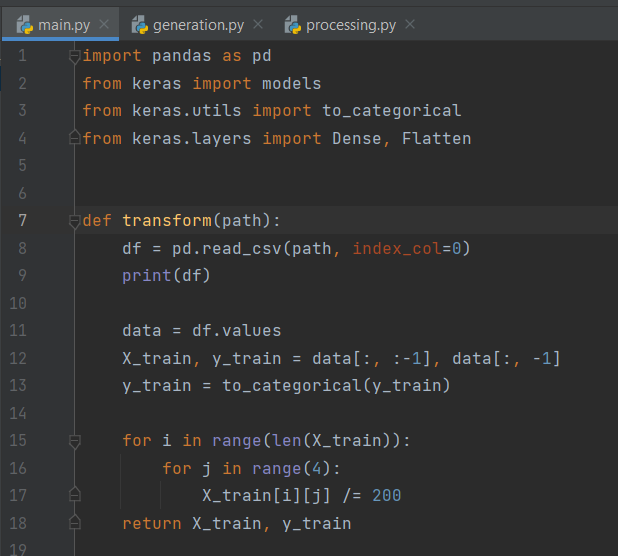


Рисунок 12 – Функція transform

Наступна функція – modell. Створюю модель. Першим шаром використовую Flatten для того, щоб перейти від двовимірного масиву до одновимірного і використати щільний шар. Далі відповідно маю Dense з 512 нейронами, а на виході – шар Dense з 2 нейронами, адже маємо результат – прийнято\не прийнято. Відповідно маємо задачу бінарної класифікації, тому використовуємо оптимальні параметри компіляції, loss=”binary\_crossentropy”. Навчаю модель у 15 епох і зберігаю її у файл для подальшого використання.

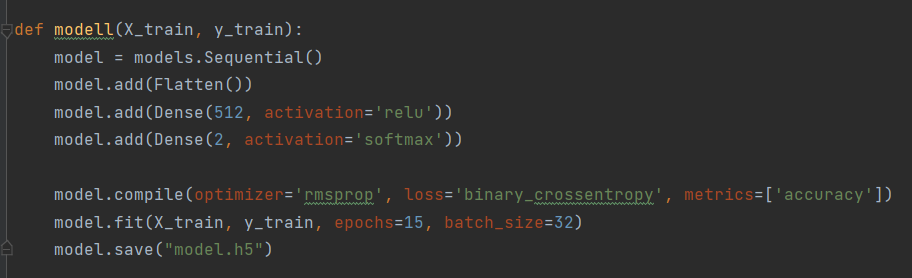


Рисунок 13 – Функція modell

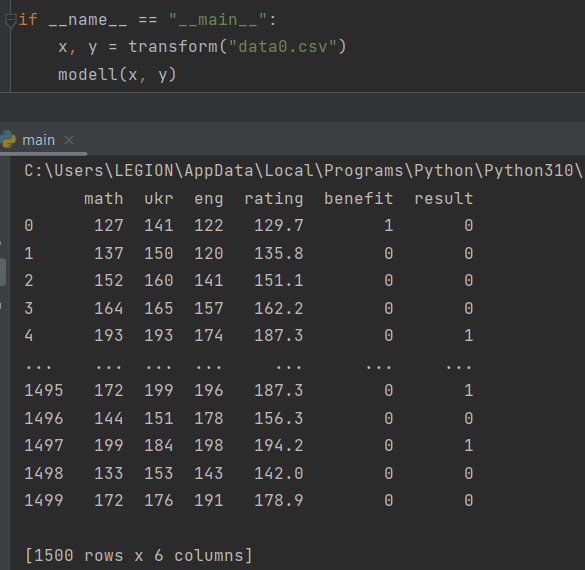


Рисунок 14 – Запуск функцій, вивід датафрейму

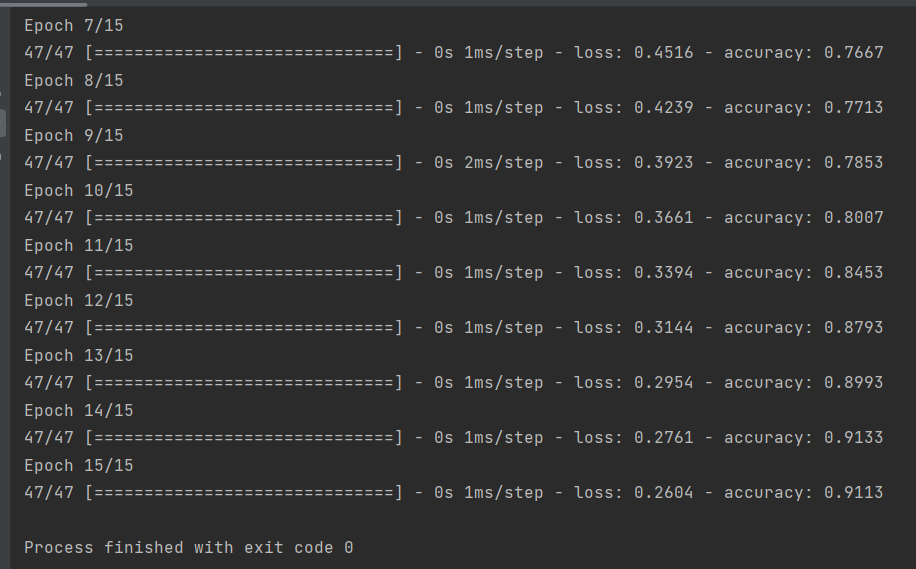


Рисунок 15 – Навчання мережі

Бачимо, що після навчання мережі на навчальних даних досягнуто точности 0.9113, що є досить хорошим результатом.

Для тестування моделі створюю функцію test.

Завантажую модель і файл, яким будемо оцінювати модель. Так само розділяю і нормалізую дані. Далі оцінюю якість моделі за допомогою функції evaluate, а також отримую передбачення за функцією predict. Обираємо в кожному передбаченні значення з найбільшою ймовірністю. Перевіряю: якщо значення дорівнює 1, тобто передбачено, що абітурієнта прийнято, то додаємо його в окремий список, який перетворюю в датафрейм. Нормалізовані дані для візуалізації знову перетворюю в 200-бальну шкалу. Сортую за спаданням за пільгами та рейтингом, тобто зверху списку буде абітурієнт з найбільшим рейтингом серед пільговиків. Отриманий список вивантажую в файл Excel.



Рисунок 16 – Функція test

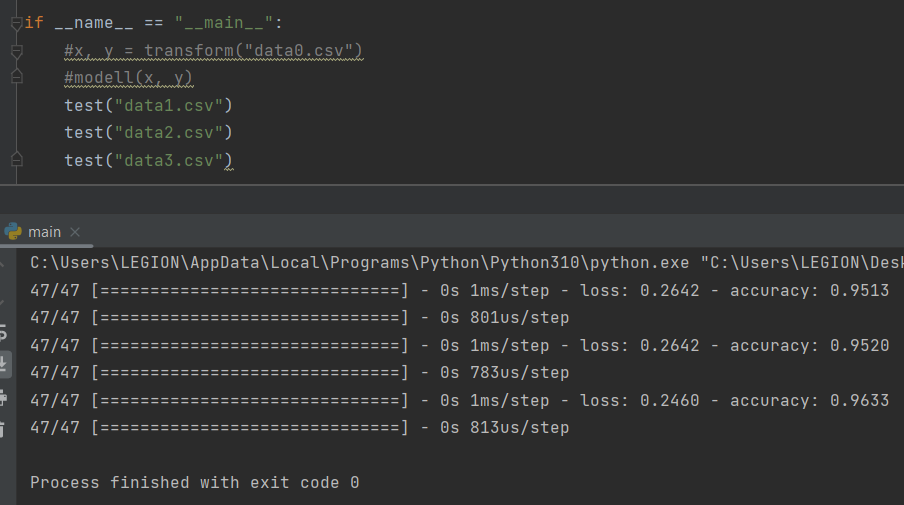


Рисунок 17 – Тестування моделі на 3 вибірках

Бачимо, що на трьох вибірках під час тестування отримано точність 0.9513, 0.9520, 0.9633. Тобто маємо дуже високу точність передбачення. Нейронна мережа працює ефективно.

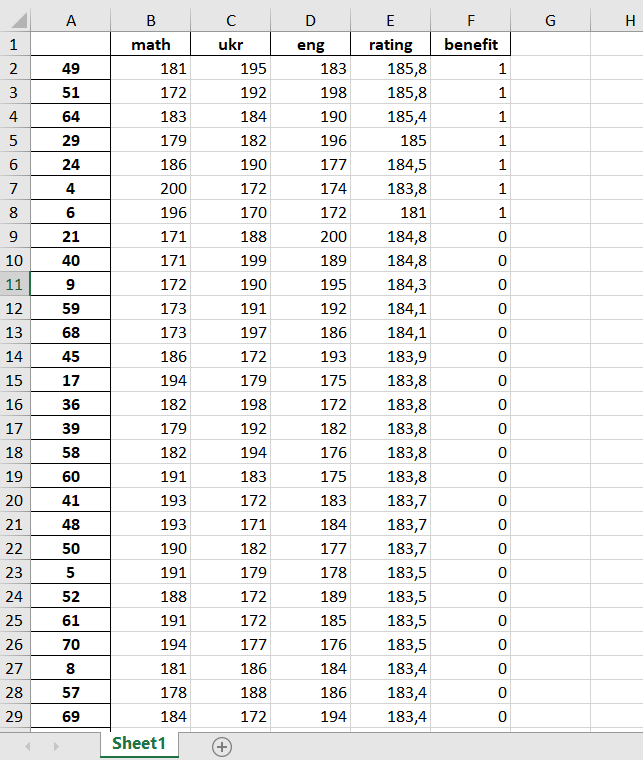


Рисунок 18 – Excel файл списку зарахованих студентів для однієї вибірки

Бачимо у першій колонці індекси зарахованих студентів у початковому датафреймі і далі відповідні показники. У першій вибірці зараховано 74 студенти, з них 7 – з пільгами.

**Висновок.**

Отже, у цій роботі я навчився розв’язувати прикладну задачу машинного навчання.

У результаті лабораторної роботи було створено, навчено й апробовано багатошарову нейронну мережу з прямою передачею сигналу для ухвалення рішення про зарахування до Університету абітурієнтів, які здали вступні іспити з математики, англійської та української мови. Було згенеровано результати іспитів. У результаті мережа має ефективність більше 90%, тож можна стверджувати, що розроблена мережа працює ефективно. Результати випробувань отримано як Excel файл.

Використовуючи програмний засіб PyCharm, мову програмування Python з модулем Keras, отримуємо коректний результат.