Лабораторна робота 1.2: Лінійна регресія з використанням реальних даних.

Завдання 1.

> Вибір варіанту завдання.

Відповідно за вказівкою викладача варіант 5.

> Ознайомтеся з описом набору даних та способом його отримання.

Для отримання набору даних потрібно вручну завантажити архів з посилання, розпакувати його та використати файл dav.csv з нього. Для завантаження даних з файлу використаємо модуль pandas.

Завдання 2.

> Завантажте набір даних за вашим номером варіанту.

Завантажуємо та розпаковуємо архів, передаємо дані з файлу для подальшої обробки.

> Здійсніть попередній аналіз даних (розмір вибірки, кількість ознак, типи даних).

Здійснивши попередній аналіз даних отримали наступні результати:

- Розмір вибірки: 731;
- Кількість ознак: 4;
- Типи даних: float64, int64.

> Обробіть пропущені значення та проведіть нормалізацію даних.

Обробка відбувається за допомогою:

```
SimpleImputer(strategy='mean').fit_transform(X_train)
```

Нормалізація даних проводиться за допомогою:

```
StandardScaler().fit_transform(X_train)
```

Завдання 3.

> Використовуйте метод train_test_split для розділення даних. Рекомендоване співвідношення: 80% для тренування, 20% для тестування.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

В X_train підуть значення, які використовуватимуться для навчання, в X_test — на яких перевірятиметься модель, y_train — цільові значення для X_train, y_test — цільові значення X_test.

Аргумент test_size=0.2 вказує на те, що для тестування вибереться 20% даних, решта піде на навчання моделі.

Завдання 4.

> Реалізуйте функцію обчислення вихідних значень моделі compute_module_output.

Функція обчислення моделі виконує обчилення прогнозу лінійної моделі, приймає на вхід:

- X — дані, m прикладів з n ознаками;
- w — вектор ваг, по одному для кожної ознаки;
- b — зміщення.

Функція виконує розрахунки за формулою:

$$f_{w,b}(x^{(i)}) = wx^{(i)} + b$$

Після розрахунку повертається вектор передбачень для всіх m прикладів.

> Реалізуйте функцію обчислення вартості compute_cost.

Функція обчислення вартосі розраховується за формулою обчислення лінійної регресії:

$$J(w,b) = \frac{1}{2m} \sum_{i=0}^{m-1} (f_{w,b}(x^{(i)}) - y^{(i)})^2$$

У якості аргументів функція приймає такі параметри:

- X — дані, m прикладів з n ознаками;
- y — цільові значення;
- w — вектор ваг;
- b — зміщення;

В результаті обчислення повертається значення функції вартості.

> Реалізуйте функцію обчислення градієнту compute_gradient.

Функція обчислює градієнт функції вартості для лінійної регресії — похідні функції вартості. Розраховується за формулами:

$$\frac{\partial J(w,b)}{\partial w_j} = \frac{1}{m} \sum_{i=0}^{m-1} (f_{w,b}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

$$\frac{\partial J(w,b)}{\partial b} = \frac{1}{m} \sum_{i=0}^{m-1} (f_{w,b}(x^{(i)}) - y^{(i)})$$

Приймає ті самі аргумети, що і функція compute_cost.

Повертрає градієнт по кожному w_j та по b.

> Реалізуйте функцію градієнту спуску gradient_descent.

Функція виконує ітераційне оновлення параметоів моделі лінійної регресії за методом градієнтного спуску. Обчислюється за формулами:

$$w_j = w_j - \alpha \frac{\partial J(w,b)}{\partial w_j}$$

$$b = b - \alpha \frac{\partial J(w,b)}{\partial b}$$

Приймає аргументи:

- X — Дані, m прикладів з n ознаками;
- y — цільові значення;
- w_in — початкові значення параметрів моделі;
- b_in — початкове значення параметра моделі;
- alpha —швидкість навчання;
- num_iters — кількість ітерацій градієнтного спуску.

Повертає:

- Оновленне значення вгаів;
- Оновлене значення b;
- Список значень функції вартості;
- Список значень параметрів [*w, b*] за ітераціями

Завдання 5.

> Ініціалізуйте початкові значення параметрів.

Рекомендовані початкові значнення параметрів було замінено на більш відповідні, через те що кофеіцієнт детермінації у тому випадку ~ -4.5, що є недопустимо. Тому початкові дані було замінено на:

- w_init = np.zeros(X_train.shape[1]) # параметр w для кожної ознаки
- b_init = 0 # початкове значення b
- alpha = 0.01 (0.0001 початкове) # швидкість навчання
- iterations = 350 (2000 початкове) # кількість ітерацій

> Виконайте алгоритм спуску для знаходження оптимальних параметрів.

Знайдені параметри (*w, b*):

w=[596.89623949,
 606.79268925,
 -402.12288497,
 -283.73185132]

b=4425.6534

> Відстежуйте зміну функції вартості під час навчання.

Ітерація 0: Вартість 1.20e+07, w[0]: 1.193e+01, b: 4.56098e+01

Ітерація 35: Вартість 6.28e+06, w[0]: 3.071e+02, b: 1.38465e+03

Ітерація 70: Вартість 3.58e+06, w[0]: 4.523e+02, b: 2.32660e+03

Ітерація 105: Вартість 2.28e+06, w[0]: 5.244e+02, b: 2.98921e+03

Ітерація 140: Вартість 1.63e+06, w[0]: 5.605e+02, b: 3.45532e+03

Ітерація 175: Вартість 1.32e+06, w[0]: 5.788e+02, b: 3.78321e+03

Ітерація 210: Вартість 1.16e+06, w[0]: 5.882e+02, b: 4.01386e+03

Ітерація 245: Вартість 1.08e+06, w[0]: 5.930e+02, b: 4.17611e+03

Ітерація 280: Вартість 1.05e+06, w[0]: 5.953e+02, b: 4.29024e+03

Ітерація 315: Вартість 1.03e+06, w[0]: 5.965e+02, b: 4.37053e+03

Ітерація 349: Вартість 1.02e+06, w[0]: 5.969e+02, b: 4.42565e+03

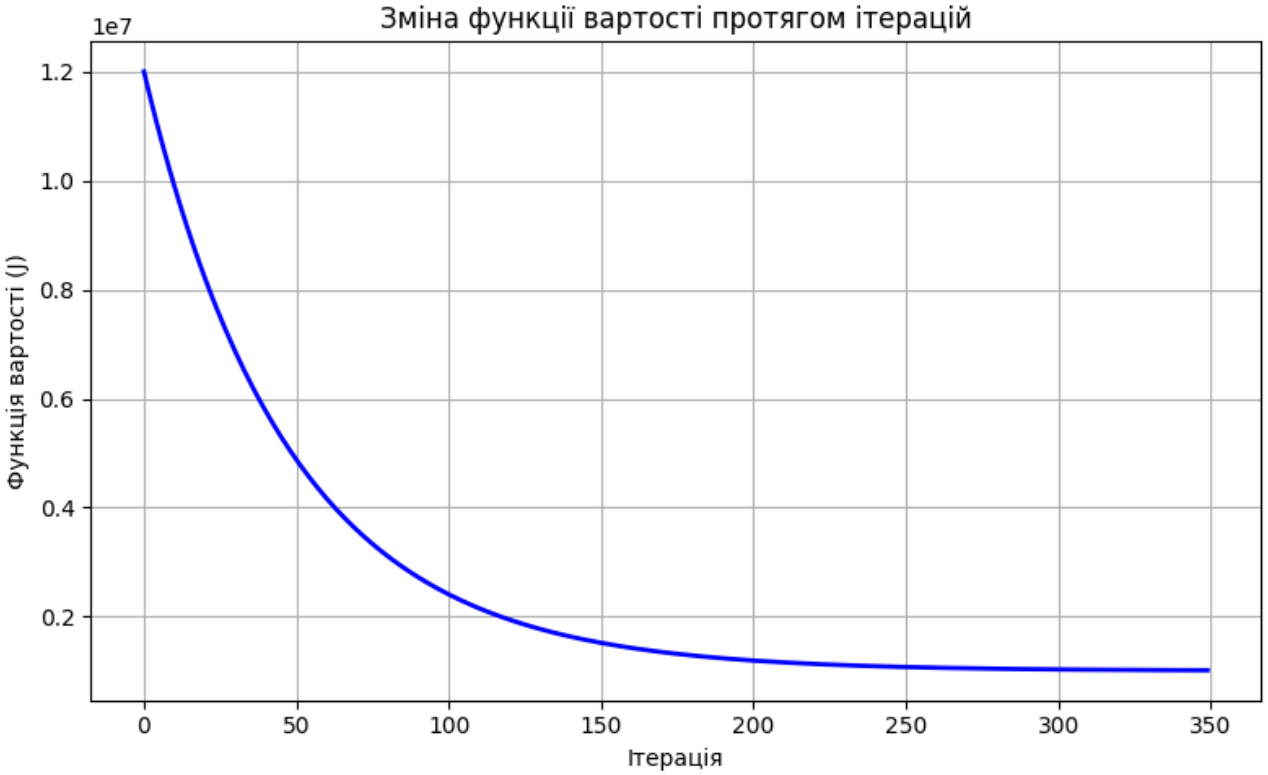

Рисунок 1 — Графік зміни функції вартості протягом ітерацій

Завдання 6.

> Обчисліть метрики якості моделі (R^2, MSE, MAE) на тренувальній вибірці.

Коефіцієнт детермінації (R^2): 0.4449

Середньоквадратична помилка (MSE): 2034650.6841

Середня абсолютна помилка (MAE): 1157.9300

> Проведіть оцінку моделі на тестовій вибірці.

Коефіцієнт детермінації (R^2): 0.0969

Середньоквадратична помилка (MSE): 3621356.1910

Середня абсолютна помилка (MAE): 1622.0126

Результати так собі. Але не від’ємні.

Завдання 7.

> Візуалізуйте дані та отриману модель

> Проаналізуйте процес навчання (зміну функції вартості)

Функція вартості монотонно зменшується, а крива має експоненціальний спад. З графіку на рисунку 1 видно, що модель стабільно поступово навчається, крок навчання підібрано коректно, і можливо, можна скоротити кількість ітерацій.

> Створіть графік прогнозів моделі порівняно з справжніми значеннями

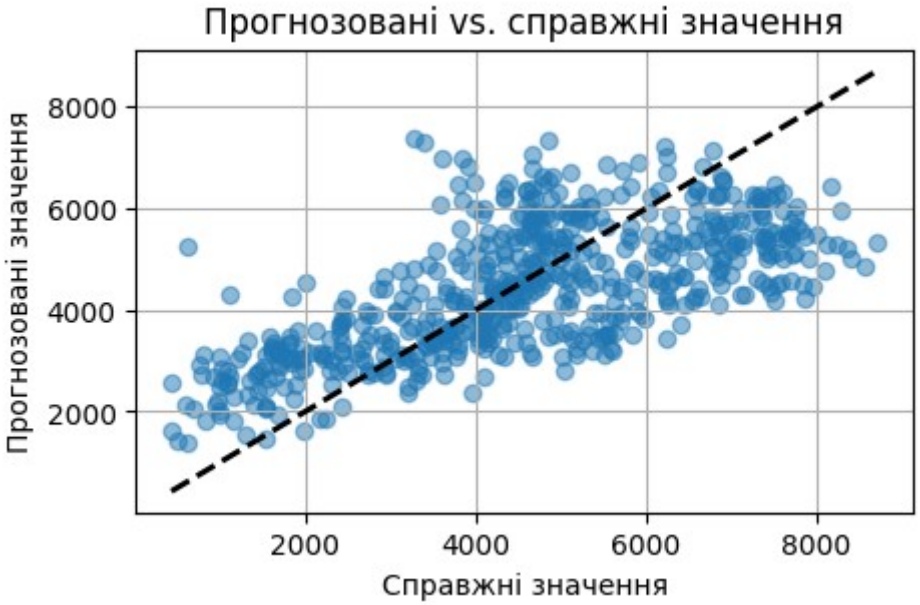

Рисунок 2 — Графік прогнозів моделі порівняно з справжніми значеннями

Завдання 8.

> Змініть значення швидкості навчання (alpha) та кількості ітерацій.

Візьмемо декілька значень швидкості навчання та декілька значень ітерацій:

alphas = [1, 0.1, 0.01, 0.001, 0.0001, 0.00001]

iterations_list = [501, 1001, 2501, 5001]

Поступово підставимо їх у функцію gradient_descent для обчислення за допомогою циклу.

Learning Rate	Ітерацій	Поведінка w і b	Коментар
1	501–5001	w → 1e+23 → 1e+213, b → 1e+300+	Розбігання, переповнення
0.1	501–5001	w → ~550 → 470, b → 4560	Швидка і стабільна збіжність
0.01	501–5001	w → 596 → 553, b → 4560	Стабільна, трохи повільніша збіжність
0.001	501–5001	w → 376 → 596, b → 1798 → 4530	Повільна, але впевнена збіжність
0.0001	501–5001	w → 56 → 375, b → 222 → 1794	Дуже повільна
1e-5	501–5001	w → 5.9 → 56, b → 22 → 222	Надто повільна, не збігається повністю навіть за 5000 ітерацій

Вибір кількості ітерацій має базуватись на тому, де графік функці вартості переходить у плато.

> Дослідіть вплив різних підходів до нормалізації даних.
Для досліду візьмемо три різних підходи: MinMaxScaler, RobustScaner, StandartScaner.

Порівняльна таблиця:

Скейлер	R^2	MSE	MAE
MinMaxScaler	0.2707	2.67e+06	1338.70
RobustScaler	0.4269	2.10e+06	1177.60
StandardScaler	0.4449	2.03e+06	1157.93

Для поточної задачі найкраще підходить StandardScaler, найвищий показник якості моделі та найменші похибки серед представлених. Непоганю альтернативою буде RobustScaler. Найменш стабільним є MinMaxScaler, що може призвести до погіршення результатів.

> Порівняйте результати з різними наборами ознак.

Набір ознак	R^2	MSE	MAE	Висновки
temp, atemp, hum, windspeed	0.4648	2055335.82	1158.67	Найкраща якість моделі (найвище R^2, найменші помилки)
temp, hum	0.4550	2094938.51	1167.01	Досить подібно до повного набору, але трохи гірше
temp, atemp, hum	0.4308	2250170.84	1206.64	Втрата точності — ймовірно, через виключення важливих ознак
atemp, windspeed	0.4177	2340944.73	1233.80	Найгірший результат — найбільші помилки, найменше R^2

> Які особливості має обраний вами набір даних? Які труднощі можуть виникнути при його аналізі?

Труднощі можуть виникати з пропущенми значеннями в даних, вмістом шуму, незбалансованості, перехресною кореляцією змін та нормалізацією даних. Всі ці параметри можуть достатньо вплинути на точність моделі.

> Як впливає нормалізація даних на результати лінійної регресії?

Нормалізація допомагає лінійній регресії, коли змінні мають різні масштаби. Вона пришвидшує навчання, покращує інтерпретацію коефіцієнтів, стабілізує регуляризацію та підвищує ефективність обчислень. Якщо змінні мають різні масштаби, нормалізація стає важливою для стабільних і точних результатів. Також може зменшити вплив викидів, оскільки всі дані приведено до одного масштабу, уникати домінування змінних з великими значеннями.

> Як визначити оптимальне значення швидкості навчання для градієнтного спуску?

Можна використати експериментальний підхід, підставляючи певне значення та спостерігаючи за поведінкою функції витрат. Додатково можна підключити виведення значень у графік для спрощення спостереження. Або можна використовувати автоматичні алгоритми, що самі налаштовують швидкість навчання.

> Що означає коефіцієнт при ознаці в лінійній регресії? Як інтерпретувати отримані коефіцієнти?

Коефіцієнт лінійної регресії показує, як змінна ознаки впливає на результат.

> Які показники свідчать про успішне навчання моделі?

Функція втрат, точність, точність на тестових даних, графік помилок.

> Які метрики використовуються для оцінки якості лінійної регресії?

Середня абсолютна помилка, середня квадратична помилка, коефіцієнт детермінації.

> Як визначити, чи є проблема перенавчання (overfitting) у моделі?

Проблему перенавченості можна помітити, якщо модель добре працює на тренувальних даних і погано на нових. Іншими словами, виходить так, що модель ніби «запам’ятала» дані замість того, щоб навчитися їх узагальнювати.

> Які переваги та недоліки лінійної регресії порівняно з іншими методами машинного навчання?

Переваги лінійної регресії у її простоті, швидкості для навчання та те, що вона має малий ризик перенавчання. Але є і мінуси: вона не підходить для нелінійних залежностей, чутлива до викидів, їй потрібна нормалізація та вона обмежена на кількість ознак.

> Як впливає кількість ознак на результати та швидкість навчання лінійної регресії?

Коли ознак небагато, то модель навчається швидко і просто, хоч і при цьому може бути недостатньо точною. Коли ознак багато, модель може більш точно описувати дані, але ціною зменшення швидкості навчання і зростанням ризику перенавчання.

> Які модифікації можна внести в алгоритм градієнтного спуску для покращення його роботи?

Можна змінвати розмір кроку швидкості навчання, нормалізувати дані і також ділити на менші частини, щоб зменшити навантаження.

Висновки: Під час виконання лабораторної роботи було реалізовано побудову моделі лінійної регресії на реальних даних: пройдені кроки від завантаження датасету, нормалізації моделі, реалізації функцій обчислення вартості та градієнтів, запуску граієнтного спуску. Було знайдено опитмальні параметри для моделі та оцінено точність за допомогою метрик: коефіцієнту детермінації, середньої абсолютної та середньої квадратичної похибок. У ході виконання модель продемонструвала посередні результати, на тестовій виборці R^2 склав лише 0.0969, що говорить про слабке узагальнення. Також було досліджено вплив швидкості навчання, кількості ітерацій і методів нормалізації на результати навчання моделі.