Київський національний університет імені Тараса Шевченка

Факультет комп`ютерних наук та кібернетики

Кафедра інтелектуальних програмних систем

Алгоритми та складність

Завдання №22-2

“Побудова лінійної регресії за відомими даними та мітками”

Варіант №2

Виконала студентка 2-го курсу

Групи ІПС-22

Челушкіна Валерія Олександрівна

Київ – 2024

**Завдання**

Реалізація побудови лінійної регресії за відомими даними та мітками для раціональних матриць типу T\*\*. В проекті також був використаний алгоритм Штрассена (для множення матриць).

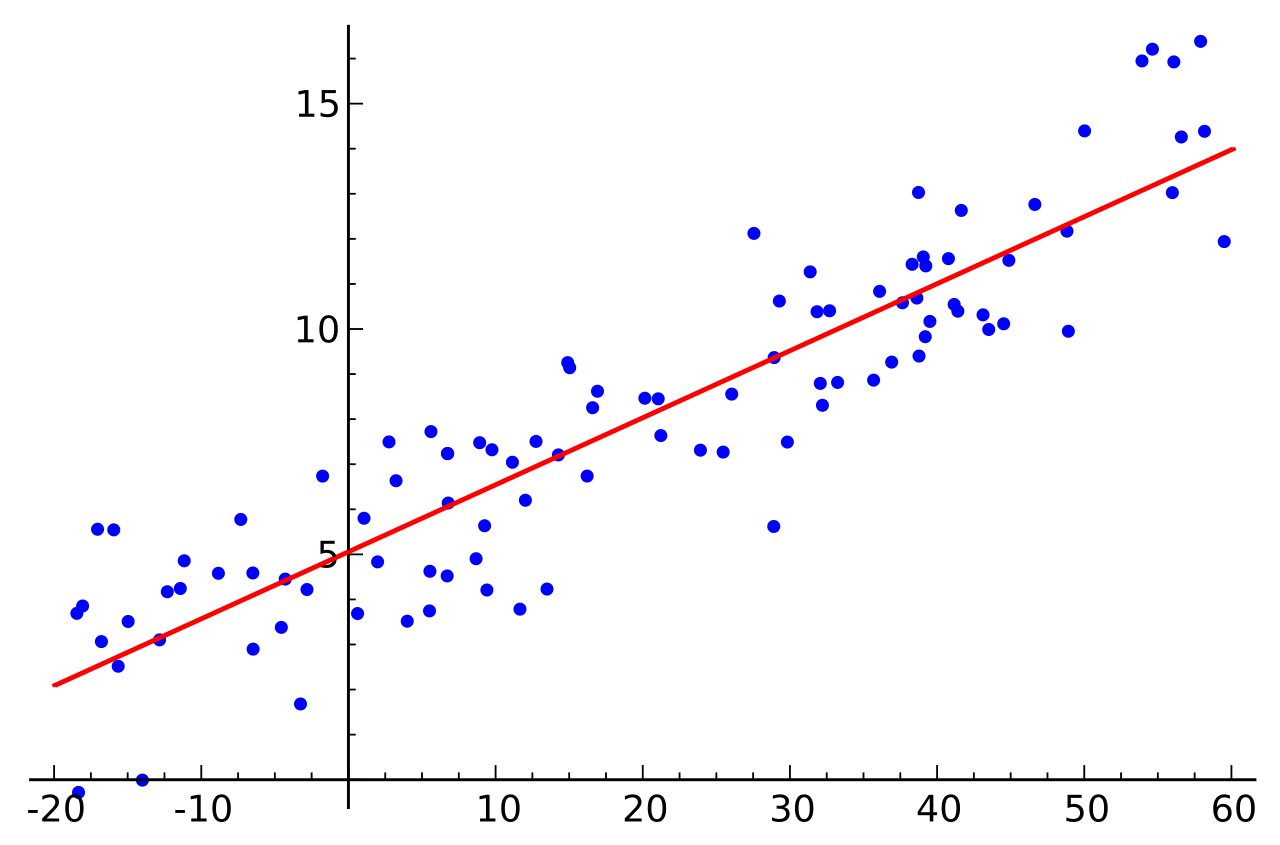
Відповідно цей алгоритм був реалізований в проекті Левицького Івана.

**Теорія**

Побудова лінійної регресії за відомими даними та мітками — це статистичний метод, який використовується для моделювання залежності між змінними, де одна змінна є залежною (цільовою), й інша (або декілька інших, у випадку множинної лінійної регресії) – незалежна (ознака).

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

Основна мета лінійної регресії — знайти лінійне співвідношення між змінними, яке дозволяє передбачити значення залежної змінної на основі значень незалежної змінної або змінних. Це досягається шляхом побудови рівняння прямої лінії, яке найкраще описує залежність між змінними, що дозволяє здійснювати прогнози, аналізувати вплив факторів та виявляти тенденції у даних. Знаходження рівняння можна обчислити шляхом мінімізації суми квадратів відхилень прогнозованих значень від фактичних (або Метод найменших квадратів).



*Формули для обчислення лінійної регресії:*

*Проста лінійна регресія (лінійна регресія з однією ознакою):*

де:

y – цільова змінна; змінна, яку намагаємось передбачити.

x – ознака (відомі дані);

– вільний член (зсув);

– коефіцієнт при ознаці (нахил прямої);

– випадкова похибка, яка враховує вплив інших факторів, не врахованих у моделі.

*Множинна лінійна регресія (лінійна регресія з n ознаками):*

**Алгоритм.**

1. Збір даних, що складаються із цільової змінної y (яку треба передбачити) та ознак x (які використовується для передбачення) для n спостережень.
2. Формування матриць.

Створюємо матрицю Х розміром (n x k+1), де кожен рядок відповідає спостереженню, а стовпці (починаючи з другого) – ознаці (від 1 до k). Перший стовпець матриці X обов’язково повністю складається з одиниць для врахування вільного члена w0.

Вектор *y* створюється із цільових значень розміром n.

1. Обчислення коефіцієнтів лінійної регресії за формулою:

Тобто:

* транспонуємо матрицю X;
* множимо на матрицю X;
* обертаємо матрицю, що вийшла;
* ще раз множимо на транспоновану матрицю X;
* результат множимо на вектор *y* – отримаємо вектор коефіцієнтів *w* розміром (k+1) – тобто , де i = 0, 1, …, k.

1. Підставляємо отримані коефіцієнти у нашу модель, отримуючи рівняння лінії, що описує залежність між змінними.

Надалі можна користуватись отриманим рівнянням для прогнозування цільової змінної за певною ознакою: .

**Складність.**

«Найскладнішими», тобто найвагомішими модулями, що впливають на загальну складність, є функції обернення та множення матриць за алгоритмом Штрассена.

Складність функції обернення в даній програмі становить O(n^3), тоді як складність алгоритму Штрассена —O(n^2.8). Враховуючи, що обернення використовується лише один раз, тоді як множення — тричі, доцільно вважати, що алгоритм Штрассена домінує над алгоритмом обернення матриці, і загальна складність алгоритму становитиме O(n^2.8).

Однак важливо враховувати, що для дуже великих матриць алгоритм Штрассена може виявитися менш ефективним, і в таких випадках може знадобитися повернення до класичного методу множення матриць.

**Мова реалізації алгоритму.**

С++

**Модулі програми.**

* **Клас Rational.**

*Поля класу*: int numerator, denominator – чисельник та знаменник числа відповідно.

*Методи класу:*

Rational(int numerator = 0, int denominator = 1) – конструктор.

Rational operator+(const Rational &other) const – перевиражене додавання.

Rational operator-(const Rational &other) const – перевиражене віднімання.

Rational operator\*(const Rational &other) const – перевиражений добуток.

Rational operator/(const Rational &other) const – перевиражена частка.

friend std::ostream &operator<<(std::ostream &os, const Rational &r) – перевиражений оператор виводу.

* **Основні методи програми.**

void Free\_Matrix(Rational \*\*matrix, int rows) – звільнення пам’яті.

Rational \*\*Create\_Matrix(int rows, int cols) – створення нової матриці.

Rational \*\*normal\_multi(Rational \*\*A, int rowsA, int colsA, Rational \*\*B, int rowsB, int colsB) – класичний добуток матриць за визначенням.

Rational \*\*convertToDoublePointer(Rational arr[][msize], const int SIZE)

Rational \*\*Add\_Matrix(Rational \*\*A, Rational \*\*B, const int SIZE) – додавання матриць.

Rational \*\*Subtraction\_Matrix(Rational \*\*A, Rational \*\*B, const int SIZE) – віднімання матриць.

Rational \*\*Strassen\_Algo(Rational \*\*A, Rational \*\*B, int SIZE) – алгоритм Штрассена.

void Show\_Matrix(Rational \*\*matrix, int rows, int cols, bool asFraction = true) – вивід матриці.

Rational \*\*Inverse\_Matrix(Rational \*\*matrix, int SIZE) – обернення матриці.

void Show\_Coefficients(Rational \*\*matrix, int rows, int cols, bool asFraction = true) – вивід коефіцієнтів.

Rational \*\*StrassenMultiplyNonSquare(Rational \*\*A, int rowsA, int colsA, Rational \*\*B, int rowsB, int colsB) – приведення неквадратних матриць для квадратного вигляду для обчислення за алгоритмом Штрассена.

Rational \*\*Transpose\_Matrix(Rational \*\*matrix, int rows, int cols) – транспонування матриці.

void Linear\_Regression(Rational \*\*X, Rational \*\*Y, int n, int m) – побудова лінійної регресії.

**Інтерфейс користувача.**

Введення даних відбувається через консоль користувачем.

*Вхідні дані:*

кількість спостережень;

кількість ознак;

заповнення цих даних у матрицю.

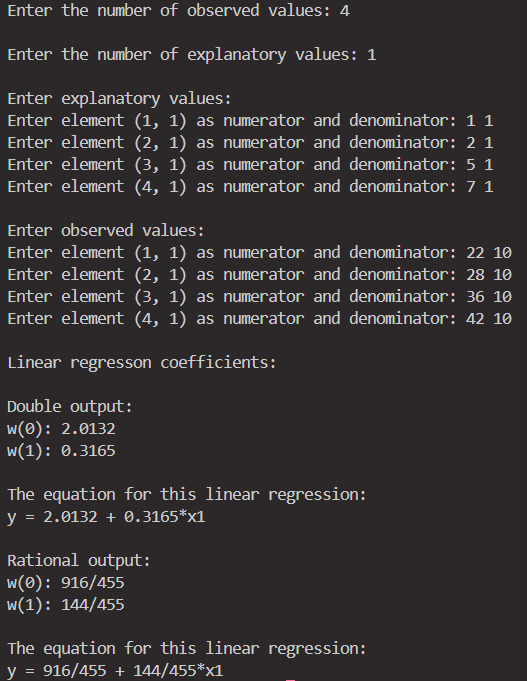
*Вихідні дані:*

коефіцієнти лінійної регресії (в раціональному і в більш звичному нам дробовому вигляді);

рівняння лінійної регресії.

**Тестові приклади**

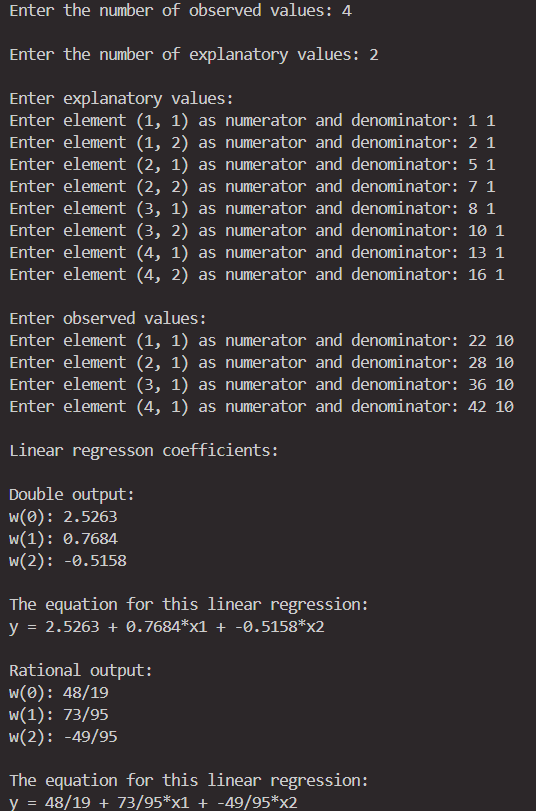
**Приклад №1**



Користувач вводить кількість спостережень, кількість ознак і заповнює ці значення. Проводиться побудова лінійної регресії і виводяться коефіцієнти та рівняння.

Для даних значень рівняння матиме вигляд: y=2.013+0.317x (обрахунки з матрицями були проведені вручну, відповіді співпадають).

**Приклад №2**

****

**Висновки.**

Отже, у даній роботі реалізували алгоритм побудови лінійної регресії за відомими мітками та даними. Був створений клас раціональних чисел для робот з матрицями у відповідному типі. Були реалізовані потрібні функції для роботи з побудови цього алгоритму. Після збору певних даних, програма надає користувачу рівняння і коефіцієнти даної лінійної регресії.

Лінійна регресія це потужний і широко використовуваний метод для моделювання залежності між цільовою змінною та ознаками. За допомогою нього можливо робити відносно точні та зручні прогнози.

Варто бути обережним при роботі з великою кількістю даних, бо доволі часто це може призвести до збільшення складності і часу виконання програми.

**Література.**

[**https://en.wikipedia.org/wiki/Linear\_regression**](https://en.wikipedia.org/wiki/Linear_regression)

[**https://en.wikipedia.org/wiki/Least\_squares**](https://en.wikipedia.org/wiki/Least_squares)

[**https://medium.com/@ayoakinkugbe/linear-regression-from-scratch-using-matrices-991df6e28f62**](https://medium.com/@ayoakinkugbe/linear-regression-from-scratch-using-matrices-991df6e28f62)

**https://mlu-explain.github.io/linear-regression/**