***Министерство образования Республики Беларусь***

***Учреждение образования***

***“Брестский государственный технический университет”***

***Кафедра ИИТ***

***Отчёт по лабораторной работе №7***

**Выполнил:**

Студент группы ИИ-23(1)

1 курса

Макаревич Н. Р.

**Проверил:**

Анфилец С.В.

Брест 2023

Вариант 2

**Задание:**

1. Вспомнить алгоритмы генерации случайных чисел. Использование случайных чисел для решения задач оптимизации.
2. [Вариант](file:///C:\Users\User\AppData\Local\Temp\Rar$DIa2228.6919\Лабораторная%20работа%20%237.docx#oxbpb8m99tne) в соответствии с номером в журнале.
3. Данные записаны в файлы в csv-формате. Необходимо загрузить их в программу.
4. Реализовать линейную модель для аппроксимации процесса.

Модель линейной регрессии имеет следующий вид:

,

где - выходное значение модели,

xi - входные значения,

wi - параметры модели, которые нам необходимо настроить,

n - размерность входных значений.

1. Настроить параметры модели методом случайного поиска (см. лекции). Для оценки качества модели можно использовать среднюю квадратичную ошибку:
2. В процессе обучения выводить информацию об ошибке. И в результате вывести ответ: Как выглядит ваша модель со всем набором её параметров.
3. Предложить другую модель (использовать тригонометрические функции и пр.), выбрать параметры для нее. Сравнить ошибку с линейной моделью.

Код программы:

#include <iostream>

#include <fstream>

#include <vector>

#include <cmath>

#include <cstdlib>

#include <ctime>

using namespace std;

vector<double> readFromFile(string filename) {

vector<double> data;

ifstream file(filename);

if (file.is\_open()) {

double value;

while (file >> value) {

data.push\_back(value);

}

file.close();

}

else {

cout << "Error: could not open file " << filename << endl;

}

return data;

}

double linear(vector<double> x, vector<double> w) {

double y = 0.0;

double w0 = 1;

for (int i = 0; i < x.size(); i++) {

y += w0 + (x[i] \* w[i]);

w0 = w[i];

}

return y;

}

double MSE(vector<double> y\_pred, vector<double> y\_true) {

double mse = 0.0;

for (int i = 0; i < y\_pred.size(); i++) {

mse += pow(y\_pred[i] - y\_true[i], 2);

}

mse /= y\_pred.size();

return mse;

}

double relative\_error(vector<double> y\_pred, vector<double> y\_true) {

double re = 0.0;

for (int i = 0; i < y\_pred.size(); i++) {

re += abs(y\_pred[i] - y\_true[i]) / y\_true[i];

}

re /= y\_pred.size();

return re;

}

double randomWeight() {

return (double)rand() / RAND\_MAX \* 2 - 1;

}

vector<double> random\_search(vector<double> x, vector<double> y, int n\_iter) {

vector<double> best\_w(x.size(), 0.0);

double best\_error = INFINITY;

for (int iter = 0; iter < n\_iter; iter++) {

vector<double> w(x.size(), 0.0);

for (int i = 0; i < w.size(); i++) {

w[i] = randomWeight();

}

vector<double> y\_pred(1, linear(x, w));

double error = MSE(y\_pred, y);

if (error < best\_error) {

best\_w = w;

best\_error = error;

}

}

return best\_w;

}

double trigSin(vector<double> x, vector<double> w) {

double y = 0.0;

double w0 = 1;

for (int i = 0; i < x.size(); i++) {

y += w0 + w[i] \* sin(x[i]);

w0 = w[i];

}

return y;

}

vector<double> random\_search\_trig(vector<double> x, vector<double> y, int n\_iter) {

vector<double> best\_w(x.size(), 0.0);

double best\_error = INFINITY;

for (int iter = 0; iter < n\_iter; iter++) {

vector<double> w(x.size(), 0.0);

for (int i = 0; i < w.size(); i++) {

w[i] = randomWeight();

}

vector<double> y\_pred(1, trigSin(x, w));

double error = MSE(y\_pred, y);

if (error < best\_error) {

best\_w = w;

best\_error = error;

}

}

return best\_w;

}

int main() {

srand(time(NULL));

setlocale(LC\_ALL, "Russian");

vector<double> x = readFromFile("x.csv");

vector<double> y = readFromFile("y.csv");

int n\_iter = 100;

vector<double> w = random\_search(x, y, n\_iter);

vector<double> y\_previous(1, linear(x, w));

double re = relative\_error(y\_previous, y);

cout << "Для линейной:" << endl;

cout << "Полученный у: " << y\_previous[0] << endl;

cout << "Ожидаемый у: " << y[0] << endl;

cout << "Относительная ошибка: " << re << endl;

vector<double> w\_trig = random\_search\_trig(x, y, n\_iter);

vector<double> y\_pred\_trig(1, trigSin(x, w\_trig));

double re\_trig = relative\_error(y\_pred\_trig, y);

cout << endl << "Для тригонометрической(sin):" << endl;

cout << "Полученный у: " << y\_pred\_trig[0] << endl;

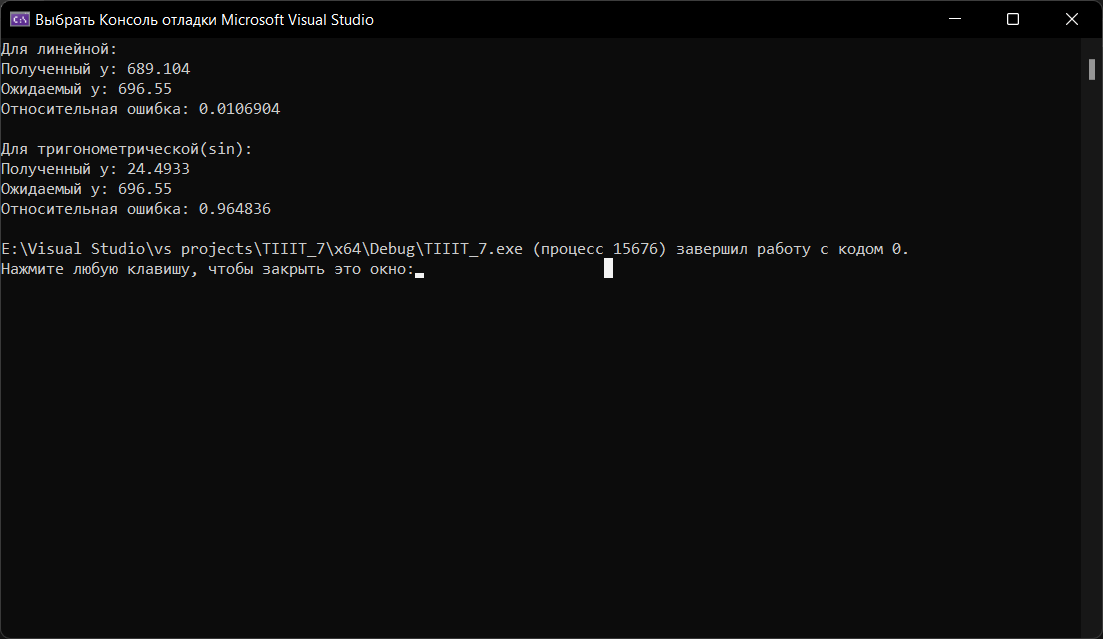
cout << "Ожидаемый у: " << y[0] << endl;

cout << "Относительная ошибка: " << re\_trig << endl;

return 0;

}

Вывод программы:



Вывод:

Была реализована линейная модель для аппроксимации процесса на языке С++.