**实验七 一元线性回归**

**一、实验目的**

理解线性回归算法的基本操作和思想。

**二、实验分析**

对于一元线性回归模型, 假设从总体中获取了n组观察值（X1，Y1），（X2，Y2）， …，（Xn，Yn）。对于平面中的这n个点，可以使用无数条曲线来拟合。要求样本回归函数尽可能好地拟合这组值。综合起来看，这条直线处于样本数据的中心位置最合理。 选择最佳拟合曲线的标准可以确定为：使总残差达到最小。

  最常用的是最小二乘法，所选择的回归模型应该使所有观察值的残差平方和达到最小。

**三、实验内容**

**源代码：**

#include<iostream>

#include<fstream>

#include<vector>

using namespace std;

class LeastSquare{

double a,b;

int n;

public:

LeastSquare(const vector<double> &x, const vector <double> & y)

{

n=x.size();

double t1=0, t2=0, t3=0, t4=0;

for(int i=0;i<n;i++)

{

t1+=x[i]/n; //x的均值

t2+=x[i]\*x[i]; //x方的和

t3+=y[i]/n; //y的均值

t4+=x[i]\*y[i]; //x\*y的和

}

a=(t3\*t2-t1\*t4)/(t2-n\*t1\*t1);

b=(t4-n\*t1\*t3)/(t2-n\*t1\*t1);

//a,b两个参数根据上面的计算结果得出

}

double getY(const double x) const

{

return a+b\*x;

}

void print() const

{

cout<<"y="<<b<<"\*x"<<"+"<<a<<endl;

}

};

int main()

{

double x0;

vector<double> x;

ifstream in("Datafile.txt");

if(!in)

std::cout<<"文件不存在！";

else

{

for(double d;in>>d;)

x.push\_back(d);

int sz=x.size();

vector<double> y(x.begin()+sz/2,x.end());

x.resize(sz/2);

LeastSquare ls(x,y);

ls.print();

cout<<"输入x的值:\n";

while(cin>>x0)

{

cout<<"y的值为:"<<ls.getY(x0)<<endl;

cout<<"输入x的值:"<<endl;

}

}

return 0;

}

**运行结果：**



