基于BP神经网络模型的矿石加工质量预测与控制

摘要

开头段：本文主要探究了在给定的条件下如何提高矿石的加工质量从而达到节能减排，实现“双减”这一伟大目标。重点对在只有温度变化的情况下，矿石的质量和合格率等各种指标进行分析。使用了×××，×××，×××，×××方法，建立了BP神经网络模型。

**针对问题一：**

**针对问题二：**

**针对问题三：**

**针对问题四：**

结尾段：

关键字：BP神经网络模型、

1. 问题重述

1.1问题一

建立适当数学模型，根据给定的原矿参数的数据和系统指定的温度，预测产品质量，并将其填入下表（表一）。

1.2问题二

根据题一和已知数据，在原矿参数和产品质量已知的情况下，给出误差最小的系统设定温度。

1.3问题三

根据前两题的结果和表三、附件二的数据，在给定的系统设定温度下，求出产品的合格率结果填入下表（表4），并且建立合适的数学模型对此进行准确性分析

1.4问题四

根据问题三和附件二中的数据，探究如何在给定合格率的条件下设定系统温度。并做以下分析：（1）敏感性分析（2）准确性分析（3）探究表5中给定的合格率能否达到（若能则完成下表表格，不能则不做操作）

1. 问题分析

**2.1**问题一的分析

剔除异常数据

数据统计处理

1.数据预处理

算法实现

神经网络预测

2.建立预测模型

问题一

算法步骤介绍

预测结果整理

预测结果误差分析数值对比

3.误差分析

确定最优神经网络层数与训练次数

预测结果与实验结果对比

4.预测结果

2.2问题二的分析

确定训练集

得到集中温度图

1.确定数据集

温度值取中值

算法步骤介绍

2.建立预测模型

问题二

预测结果误差分析数值对比

确定最优神经网络层数与训练次数

3.误差分析

预测结果与实验结果对比

4.预测结果

2.3问题三的分析

过程数据与原矿数据匹配

得到集中温度图

1.数据处理

过程数据插入值处理

评估并解释依据

算法步骤介绍

2.建立插入值模型

算法步骤介绍

问题三

确定最优神经网络层数与训练次数

3.神经网络模型预测

预测结果与实验结果对比

算法描述

4.合格率判断

统计过程

5.预测表格提供数据

2.4问题四的分析

采用问题三种的数据并且以温度作为数据变量

1.数据准备

合格产生临界指标为产品指标

求类中心

2.确定温度集

模糊某类算法

问题四

利用问题三训练过程

3.正向训练

以类中心为初始温度，相邻连线为可能存在的温度

4.代值处理

依次代值判断

1. 模型假设

假设一：假设在矿石加工的过程中，每次调节温度后的两小时之内不传入新的调温指令

假设二：针对原矿参数，假设在全天生产过程任意时间原矿参数不变。

假设三：

1. 符号说明

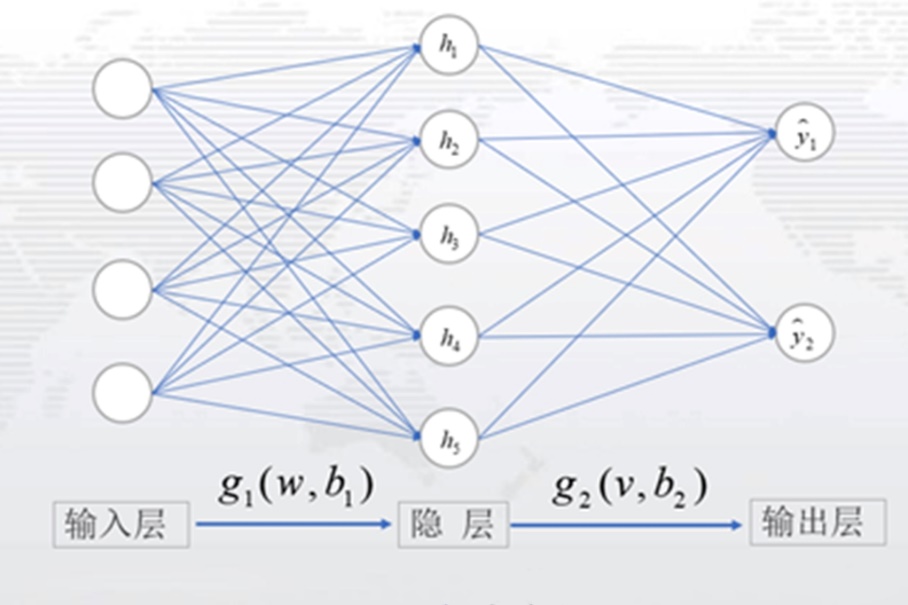
|  |  |  |
| --- | --- | --- |
| 符号 | 说明 | 单位 |
| X | 输入层数据 | 无 |
| W， | 输入层到隐藏层参数 | 无 |
| V,    G1,g2 | 隐藏层到输出层  学习率  激活函数 | 无  无  无 |

1. 问题一的模型建立与求解

**5.1 数据分析**

**5.2 基于神经网络的指标预测模型** 为了预测给定温度下的产品质量，我们先根据附件的数据建立温度和产品质量之间的关系，再代入表1所给的系统温度得到指标A、B、C、D的预测。  
 我们对附件所给的数据进行分析,得到己知数据为2022年1月13日到2022年1月22日的系统温度和产品质量以及2022年1月13日2022年到1月24日的原矿参数，考虑到温度和产品质量的时间维度不一样，我们取一段时间的温度的平均值作为数据使用，并且全天生产过程中的任意时间原矿参数不变。

**5.2.1神经网络算法介绍**

****

**图1 深度为1层的神经网络**

设定：从输入层数据为X，输入层到隐藏层参数为w,b1，隐藏层到输出层为v,b2，激活函数用为g1,g2.于是模型设定为：

输入层到隐藏层：

 （3-1）

隐藏层到输出层：

 （3-2）

模型：

 （3-3）

损失函数：

 （3-4）

其中：

，





以上述的模型设定为例，下面介绍BP算法步骤，了解反向传播是如何实现模型的参数更新。

**5.2.2神经网络算法实施步骤**

（1）初始化网络中的权值和偏置项，分别记为

 （3-5）

（2）激活前向传播，得到各层输出和损失函数的期望值

 （3-6）

其中，表示参数集合，y表示真实值， 表示预测值，表示对总的误差值求平均，所以一般情况下，输出单元多少维，误差值求平均就除以多少；本模型设定中，输出值为2维列数据，顾用误差值除以2。一般情况下，损失函数期望值表示为：

 （3-7）

这是一组n维数据的输出，若是有m维这样的数据，损失函数的期望值为：

 (3-8) 一般情况下，输出数据为1维或者2维，输出的数据有多组。

（3）根据损失函数，计算损失函数关于输出单元的梯度值或偏导数，根据链式法则有：

 （3-9）

隐藏单元的误差项，即计算损失函数关于隐藏单元的梯度值或偏导数，根据链式法则有：

 （3-8）

（4）更新神经网络中的权值和偏置项

输出单元参数更新： （3-9）

隐藏单元参数更新： （3-10）

其中，为学习率，k=1,2……n表示更新次数或迭代次数，k=1表示第一次更新，以此类推。

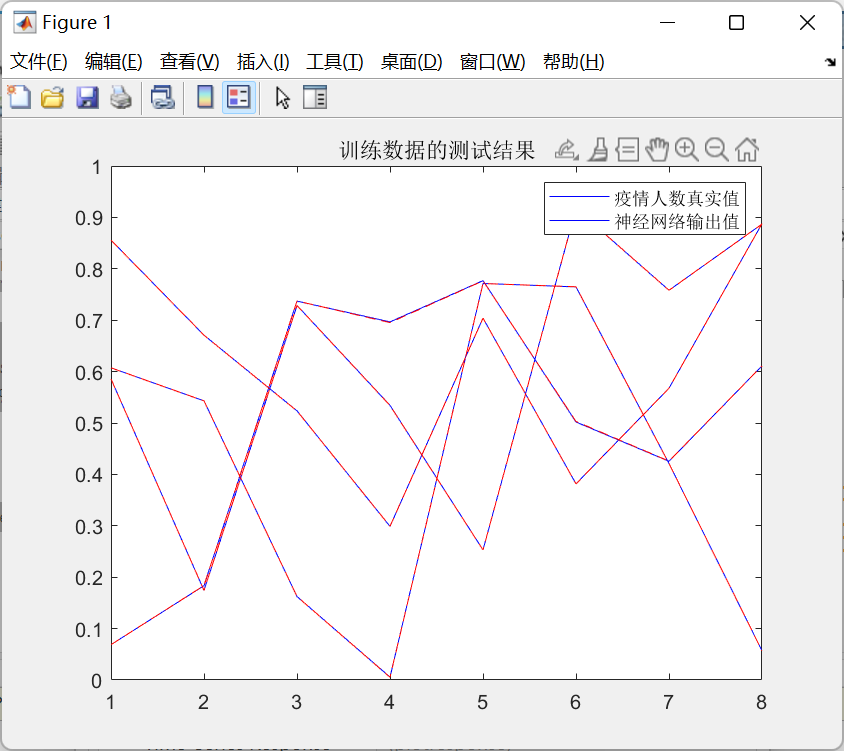
（5）重复步骤2-4，直到损失函数小于事先给定的阈值或迭代次数用完为止，输出此时的参数即为目前最佳参数。

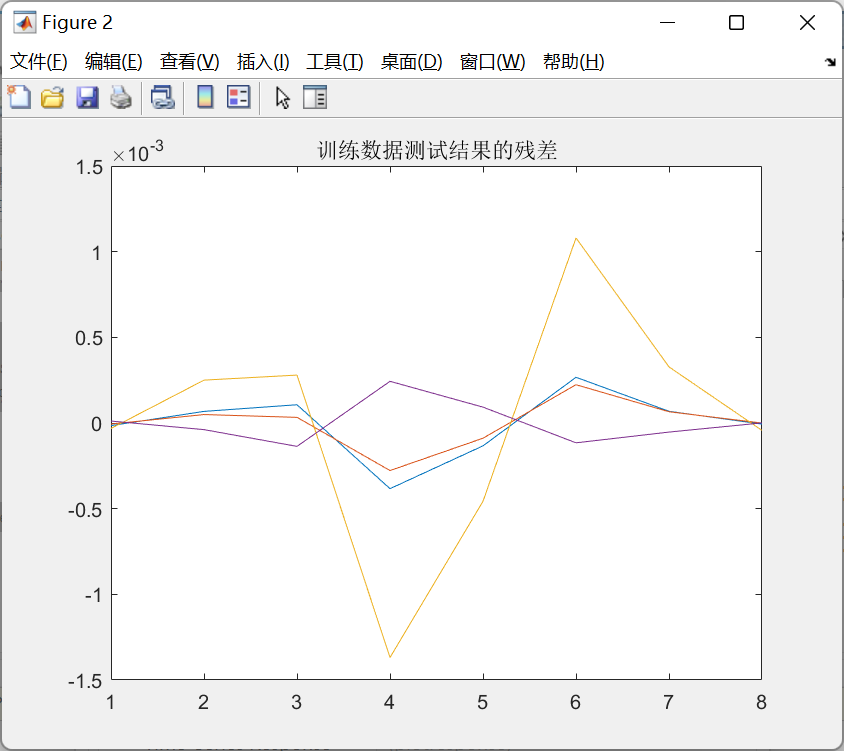
**5.3 模型求解**

在神经网络算法的基础运用上建立本题的预测模型，运用 MATLAB 软件（代码见附件），采用神经网络模型进行预测。得到结果，填入表1：

表1 问题1结果

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 时间 | 系统I设定温度 | 系统II设定温度 | 指标A | 指标B | 指标C | 指标D |
| 2022-01-23 | 1404.89 | 859.77 | 80.323845 | 11.523582 | 11.523582 | 17.482018 |
| 2022-01-23 | 1151.75 | 859.77 | 79.16022 | 12.132239 | 12.132239 | 15.5586 |





1. 问题二的模型建立与求解

6.1 模型建立

由原矿参数和产品目标质量，来预测产品目标质量所对应的系统温度，属于一个应变量依赖多个自变量的变化关系，因此我们选用神经网络模型进行温度的预测。与问题一的模型一致。

6.2模型求解

基于问题一建立的BP神经网络预测模型可对温度进行预测。运用 MATLAB 软件（代码见附件），得到结果，填入表2：

表2 问题2结果

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 时间 | 指标A | 指标B | 指标C | 指标D | 系统I设定温度 | 系统II设定温度 |
| 2022-01-24 | 79.17 | 22.72 | 10.51 | 17.05 | 1242.6108 | 781.8634 |
| 2022-01-24 | 80.10 | 23.34 | 11.03 | 13.29 | 1188.1708 | 753.2423 |

1. 问题三的模型建立与求解
2. 问题四的模型建立与求解
3. 模型评价与推广
4. 模型优化
5. 参考文献

[1]

附录：

MATLAB代码：

clear

clc

A=[1006.537401 1006.537401 79.46541667 23.90583333 11.71958333 15.87708333

1386.752458 1386.752458 79.99 22.93875 11.13625 15.34833333

989.6350487 988.6833866 79.66208333 23.68041667 11.33833333 16.81458333

742.0366134 742.0263491 79.21166667 23.66791667 11.3468 15.16208333

912.1740764 913.0282684 79.78833333 23.08 12.00916667 15.34083333

1343.263621 1343.263621 80.92666667 21.90625 12.57166667 16.25041667

1377.808097 1377.808097 80.44208333 22.51826087 11.09916667 16.8532

1088.969953 1088.969953 79.1525 24.595 12.62166667 18.68875

];

B=rand(8,4);

[m,n]=size(A)

m=8

x=A'

y=B'

trainx =x;

trainy =y;

[ww,mm]=size(trainx);

testx = [1404.89 859.77

1151.75 859.77

];

net=elmannet(1:2,15,'traingdx');

net.trainParam.show=1;

net.trainParam.epochs=10000;

net.trainParam.goal=0.000001;

net.trainParam.max\_fail=5;

net=init(net);

[trainx1, st1] = mapminmax(trainx);

[trainy1, st2] = mapminmax(trainy);

testx1 = mapminmax('apply',testx,st1);

[net,per] = train(net,trainx1,trainy1);

train\_ty1 = sim(net, trainx1);

train\_ty = mapminmax('reverse', train\_ty1, st2);

test\_ty1 = sim(net, testx1%);

test\_ty = mapminmax('reverse', test\_ty1, st2);

figure(1)

x=1:length(train\_ty);

plot(x,trainy,'b-');

hold on

plot(x,train\_ty,'r--')

legend('真实值','神经网络输出值')

title('训练数据的测试结果');

figure(2)

plot(x, train\_ty - trainy)

title('训练数据的测试结果的残差')

mse1 = mse(train\_ty - trainy);

fprintf(' mse = \n %f\n', mse1)

disp(' 相对误差')

d=(train\_ty - trainy)./trainy;

fprintf('%f ', (train\_ty - trainy)./trainy );

fprintf('\n')

disp(' 预测值')

fprintf('%f ', test\_ty%预测值 );

fprintf('\n')