实验报告

作者：刘庆星
学校：南京航空航天大学

一、 问题介绍

1.1 线性回归模型训练

给定含有1000条记录的数据集mlm.csv，其中每条记录均包含两个自变量x,y和一个因变量z，它们之间存在较为明显的线性关系。对该数据集进行三维可视化分析，并训练出良好的线性回归模型。

1.2 SVM与神经网络训练效果对比研究

（此数据集来源于UCI Machine Learning Repository）

给定一个鸢尾花数据集，包含150条记录，每条记录包含萼片长度sepal length、萼片宽度sepal width、花瓣长度petal length和花瓣宽度petal width四个数值型特征，以及它的所属类别（分别为Iris-setosa,Iris-versicolor,Iris-virginica三者之一）。如traing.txt所示

在此我使用SVM和神经网络对该数据集训练出一个良好的非线性分类器，并试着比较他们的训练效果。

二、关键难点

2.1 SVM难点

1. 如何定义SVM训练的绘图效果的函数。包括SVC参数选择结果（等高图和3D视图）。
2. 如何确定SVM的输入集和输出集。
3. 在网格算法寻优中，如何更新迭代参数。
4. 如何确定确定和函数类型　包括不敏感损失函数系数和支持向量机类型。

2.2 神经网络难点

1. 数据归一化、对p和t进行字标准化预处理、经验公式计算隐层神经元节点、N隐层结点数。
2. 如何确定神经网络训练的迭代次数、学习率、目标精度、动量因子等相关参数。

三、项目简介

3.1 SVM支持向量机

支持向量机是建立在 VC 维理论和结构风险最小原理基础上，根据有限的样本信息在模型的复杂性和学习能力之间寻求最佳折衷，以期获得最好的推广能力，其基本模型为在特征空间上找到最佳的分离超平面使得训练集上正负样本间隔最大。

3.2 BP 神经网络

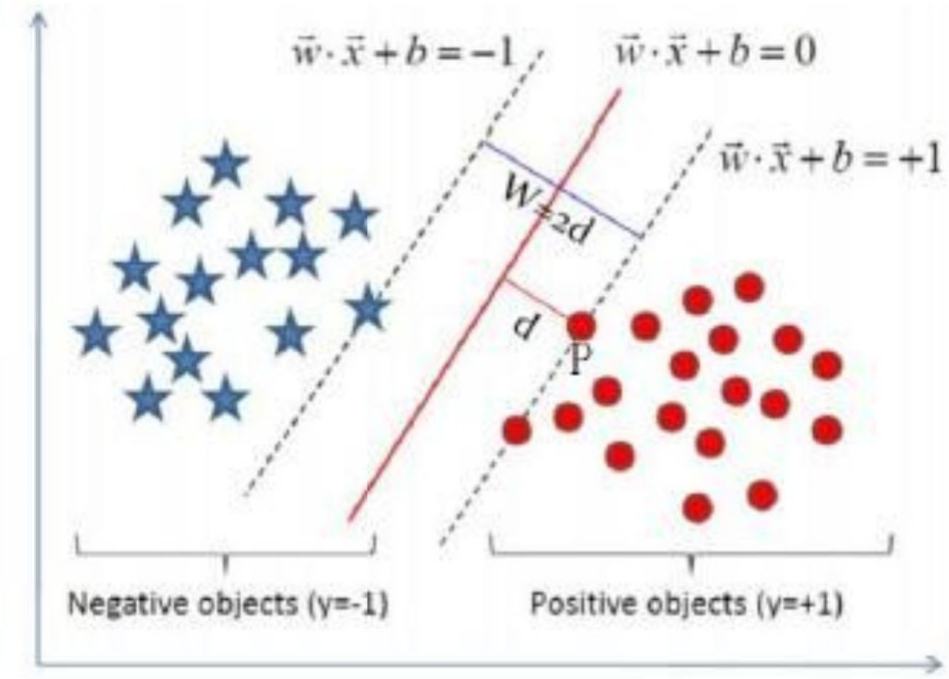
神经网络是一种运算模型，由大量的节点（或称“神经元”，或“单元”）和之间相互联接构成。每个节点代表一种特定的输出函数，称为激励函数（activation function）。每两个节点间的连接都代表一个对于通过该连接信号的加权值，称之为权重（weight），这相当于人工神经网络的记忆。网络的输出则依网络的连接方式，权重值和激励函数的不同而不同。而网络自身通常都是对自然界某种算法或者函数的逼近，也可能是对一种逻辑策略的表达。

四、数学模型

4.1 SVM

SVM 算法旨在找出一个具有最大分类间隔的超平面作为分类器，其中分类间隔 W=2d，d 等于任一支持向量到该超平面的距离。

具体而言则如下：



那么求解最优超平面的问题就转换成了另外一个问题——寻找特定的w和b，

使得 d 最大化。

$$d = \frac{\left|w^T x + b\right|}{\|w\|}$$

但我们不能随便找一个由 w 和 b 确定的、且离支持向量无限远的超平面作为最优分类器，这个最优超平面应该还要具有良好的分类效果，因此我们还需要进一步对 w 和 b 设置一些约束条件。

$$\begin{cases} w^T x_i + b > 0 & y_i = 1 \\ w^T x_i + b < 0 & y_i = -1 \end{cases} \quad \wedge \quad \begin{cases} \dfrac{w^T x_i + b}{\|w\|} \geq d & \forall y_i = 1 \\ \dfrac{w^T x_i + b}{\|w\|} \leq -d & \forall y_i = -1 \end{cases}$$

将两个基本约束条件再进一步整合，得到最后的约束条件。

$$y_i \left(w^T x_i + b \right) \geq 1 \quad \forall x_i$$

而由于支持向量总是位于最优超平面附近的虚线上，满足 $\left|w^T x_i + b\right| = 1$ 因此对 d 求解最大化的问题又可以进一步转为在相同约束条件下求解参数 w 范式 $\|w\|$ 最小值的问题。

即我们的 SVM 的基本数学模型是：

最终的目标函数：$\min \frac{1}{2}\|w\|^2$

约束条件：$s.t.\ \ \mathrm{y}_i \left(w^T x_i + b \right) \geq 1, i = 1, 2, \ldots, n$

接下来我们便要开始着手对 w 和 b 进行求解了，值得注意的是，我们的基本数学模型是一个满足 KKT 条件的不等式约束优化问题，但是我们的求解过程类似于有等式约束的优化问题，后者使用的是拉格朗日极值法。

为了方便求解，我们利用目标函数和约束条件搭建拉格朗日函数，其中 $\alpha_i \geq 0$ 被称作拉格朗日乘子，也是一个系数变量。

$$L\left(w, b, \alpha\right) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i (y_i (w^T x_i + b) - 1)$$

若我们把 $y_i(w^T x_i + b)$ 看成一个变量 k，而将 α_i 看作一个自变量来求解 $L(w,b,\alpha)$ 最大值，那么就很有意思，我们可以得到下面的函数形式

$$\max_{\alpha_i \geq 0} L(w,b,\alpha) = \begin{cases} \frac{1}{2}\|w\|^2 & x \in k \geq 1 \\ +\infty & x \in k < 1 \end{cases}$$

可以看出，在原限制条件下对拉格朗日函数求极大值，就等价于原目标函数，那么原本我们对原目标函数求极小值的操作就可以转移成对拉格朗日函数极大值再求极小值的行为了。即

最终的目标函数：$\min \frac{1}{2}\|w\|^2$

约束条件：$s.t.\ \mathrm{y}_i\left(w^T x_i + b\right) \geq 1, i = 1,2,\ldots,n$

$$\Leftrightarrow \min_{w,b} \max_{\alpha_i \geq 0} L(w,b,\alpha)$$

由于先对拉格朗日函数求 α_i 的偏导有些复杂，因此我们通过拉格朗日对偶性，先对参数 w 和 b 进行偏导计算

$$\min_{w,b} \max_{\alpha_i \geq 0} L(w,b,\alpha) \Leftrightarrow \max_{\alpha_i \geq 0} \min_{w,b} L(w,b,\alpha)$$

通过计算知

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0$$

将上面的结果代入拉格朗日函数，得

$$L(w,b,\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j {x_i}^T x_j$$

那么此时，我们只需要在一定限制条件下，进一步对 $L(w,b,\alpha)$ 进行极大值求解就可以了，即现在的问题变成如下形式：

$$\max_{\alpha} L(w,b,\alpha) = \max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j {x_i}^T x_j$$

$$s.t. \alpha_i \geq 0, i = 1,2,\ldots,n$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

结合样本集和限制条件 $\sum_{i=1}^{n} \alpha_i y_i = 0$ 我们可以通过对 $L(w,b,\alpha)$ 进行 α_i 偏导，求出 α_i 的具体值，而如果 α_i 不满足另外一个限制条件 $\alpha_i \geq 0$，我们就认为极值点位于边界处，通过依次代值计算结果，最后选择结果最大的点作为极值点即可。

这样我们就计算出了最佳 α_i，然后再通过下面的公式

$$\begin{cases} w = \sum_{i=1}^{n} \alpha_i y_i x_i \\ \vec{w} \cdot \vec{x} + b = \pm 1 \end{cases}$$

我们就可以得到参数 w 和 b 的最优值，从而确定一个最优超平面。

如果仔细观察线性 SVM 的数学模型，我们会发现它是基于数据百分百线性可分的假设，但实际数据中总是存在着一些噪音，导致数据集无法通过直线百分百进行划分，因此我们考虑放松一下模型的限制条件，使得我们求得最优超平面只需要可以正确划分近似百分百的数据就好了。为此我们在原线性 SVM 的数学模型中引入松弛变量 ξ 和惩罚参数 C，即

最终的目标函数：$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N} \xi_i$

约束条件：$s.t.\ y_i(w^T x_i + b) \geq 1 - \xi_i$,i=1,2,…,n

可以看出当 C 很大时，ξ_i 将非常小，那么 $y_i(w^T x_i + b) \geq 1 - \xi_i$，是近似于 $y_i(w^T x_i + b) \geq 1$，意味着分类严格不能有错误，也就是等价于我们的原线性 SVM 的数学模型；而当 C 很小时就意味着可以有更大的错误容忍。

同样的，在求解过程中我们利用目标函数和约束条件搭建拉格朗日函数。

$$L(w,b,\xi,\alpha,\mu)=\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i-\sum_{i=1}^{N}\alpha_i\left(y_i(w^Tx_i+b)-1+\xi_i\right)-\sum_{i=i}^{N}\mu_i\xi_i$$

把原问题转化成对拉格朗日函数求导的问题。

最终的目标函数：$\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i$

约束条件：$s.t.\ y_i(w^Tx_i+b)\geq 1-\xi_i\ ,i=1,2,\ldots,n$

$$\Leftrightarrow \max_{\substack{\alpha_i\geq 0\\ \mu_i\geq 0}}\min_{w,b,\xi_i} L(w,b,\alpha)$$

对 w,b,ξ_i 求偏导，得到新的限制条件：

$$\frac{\partial L}{\partial w}=0\Rightarrow w=\sum_{i=1}^{n}\alpha_i y_i x_i$$

$$\frac{\partial L}{\partial b}=0\Rightarrow \sum_{i=1}^{n}\alpha_i y_i=0$$

$$\frac{\partial L}{\partial \xi_i}=0\Rightarrow C-\alpha_i-\mu_i=0$$

问题的形式变成：

$$\min_{\alpha}\ \ \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j\left(x_i\cdot x_j\right)-\sum_{i=1}^{N}\alpha_i$$

$$s.t.\ \ \sum_{i=1}^{N}\alpha_i y_i=0\quad 0\leq\alpha_i\leq C, i=1,2,\ldots,N$$

再进一步对 α_i 求偏导，再结合限制条件，就可以求出具体的系数值了，从而确定一个超平面，使得其可以容忍一定程度分类错误，增强了原线性 SVM 模型的泛化能力。

更进一步，我们引入核函数，使得 SVM 模型具有解决非线性问题的能力，并提升我们的运算速度。它主要被用于 SVM 模型的最后的数学模型中，即加入核函数后，我们的数学模型问题变换成：

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \kappa\left(x_i, x_j\right)$$

$$s.t. \alpha_i \geq 0, i = 1, 2, \ldots, n$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

通过核函数，我们不必再将数据映射到高维空间中再进行内积的计算，只需要在原来的低维空间中计算内积就好了，这大大加快了我们的运算速度。

五、解决方法及代码核心介绍

5.1 三维线性拟合

数据输出函数定义为

```
function [MIX_train,DATA_train,MIX_test,DATA_test]=online_dataproduce()
%数据处理以及提取各数据集输入输出
data=xlsread('data.xlsx');N=size(data,1);
num=floor(0.8*N);order=randperm(N);
MIX_train=data(order(1:num),1:end-1)';%训练集输入
DATA_train=data(order(1:num),end)';%训练集输出
MIX_test=data(order(num+1:end),1:end-1)';%测试集输入
DATA_test=data(order(num+1:N),end)';%测试集输出
%% 可视化
figure
plot3(data(:,1),data(:,2),data(:,3),'r*')
xlabel('x 坐标')
ylabel('y 坐标')
zlabel('z 坐标')
grid on
end
```

5.2 SVM 和神经网络比较

（说明：具体数据在 data.xlsx 中，第一列为 sepal length、第二列 sepal width、第三列为 petal length、第四列为 petal width。1 代表类型 Iris-setosa、2 代表 Iris-versicolor、3 代表 Iris-virginica）

5.2.1 SVM 算法实现

```
%数据的输入和输出
data=xlsread('data.xlsx');
N=size(data,1);n=floor(N*0.8);order=randperm(N);
MIX_train=data(order(1:n),1:end-1)';%训练集输入
DATA_train=data(order(1:n),end)';%训练集输出
MIX_test=data(order(n+1:end),1:end-1)';%测试集输入
DATA_test=data(order(n+1:end),end)';%测试集输出
网格算法寻优
[bestmse,bestc,bestg]=SVMcgForRegress(tn_train,pn_train,-5,5,-5,5,5,0.5,0.5,0.05);%参数可改
cmd = [' -t 2',' -c ',num2str(bestc),' -g ',num2str(bestg),' -s 0 -p 0.01'];%t 核函数类型 p 不敏感损失函数系数 s 支持向量机类型
SVM 训练函数为 svmtrain(tn_train,pn_train,cmd);
```

5.2.2 SVM 仿真预测

```
[Predict_1,error_1,tt1] = svmpredict(tn_train,pn_train,model);
[Predict_2,error_2,tt2] = svmpredict(tn_test,pn_test,model);
```

此外，对训练的数据后我使用一组数据进行测试

```
% data=xlsread('data_pred.xlsx');
% data = mapminmax('apply',data',inputps);
% data = data';
% Predict_3 = svmpredict(ones(size(data,1),1),data,model);
% disp(['预测类别为：' num2str(Predict_3')])
```

5.2.3 神经网络实现

5.2.3.1 部分参数设置

```
[inputn,mininput,maxinput,outputn,minoutput,maxoutput]=premnmx(input_train,output_train); %对 p 和 t 进行字标准化预处理
N=floor(sqrt(size(input_train,2)))+1;%经验公式计算隐层神经元节点数
net=newff(minmax(inputn),[N,size(output_test,1)],{'tansig','purelin'},'trainlm');%N 隐层结点数，'tansig','purelin'传递函数，'trainlm'训练函数
%trangdm 带动量的梯度下降
net.trainParam.epochs=500;%迭代次数
net.trainParam.lr=0.01;%学习率
net.trainParam.goal=0.000001;%目标精度
net.trainParam.mc=0.9;%动量因子
```

5.2.3.2 网络训练部分

```
net=train(net,inputn,outputn);%对训练数据进行训练
InputWeights=net.iw{1,1};%提取出权值
```

```
LayerWeights=net.lw{2,1};
bias1=net.b{1};%提取出阈值
bias2=net.b{2};
```

5.2.3.4 测试集代码

```
inputn_test = tramnmx(input_test,mininput,maxinput);
an=sim(net,inputn_test);
test_simu=postmnmx(an,minoutput,maxoutput);
test_simu=round(test_simu);test_simu(test_simu>3)=3;test_simu(test_simu<1)=1;
```

5.2.3.5 训练集。

```
inputn_train = tramnmx(input_train,mininput,maxinput);
an=sim(net,inputn_train);
train_simu=postmnmx(an,minoutput,maxoutput);
train_simu=round(train_simu);train_simu(train_simu>3)=3;train_simu(train_simu<1
)=1;
```

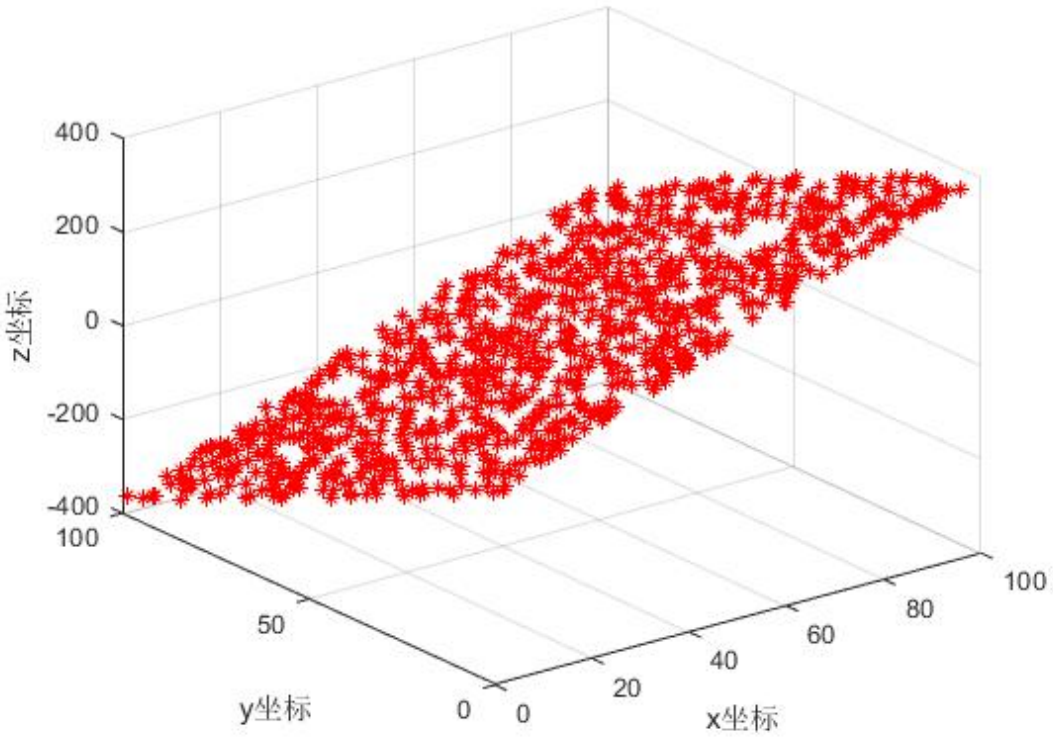
5.2.3.6 SVM 和 BP 神经网络训练效果对比代码

```
train_acc1=sum(t_train'==train_simu)/length(t_train);%bp
train_acc2=sum(t_train==Predict_1)/length(t_train);%svm
test_acc1=sum(t_test'==test_simu)/length(t_test);%bp
test_acc2=sum(t_test==Predict_2)/length(t_test);%svm

figure
plot(1:length(t_test),Predict_2,'r-*',1:length(t_test),t_test,'b:o',1:length(t_
test),test_simu,'k-*')
grid on
legend('svm 预测值','真实值','bp 预测值')
xlabel('样本编号')
ylabel('类别')

string_2 = {'测试集预测类型对比';
           ['svm 准确率 = ' num2str(test_acc2*100)];
           ['bp 准确率 = ' num2str(test_acc1*100)];};
title(string_2)

figure
plot(1:length(t_train),Predict_1,'r-*',1:length(t_train),t_train,'b:o',1:length
(t_train),train_simu,'k-*')
grid on
legend('svm 预测值','真实值','bp 预测值')
```

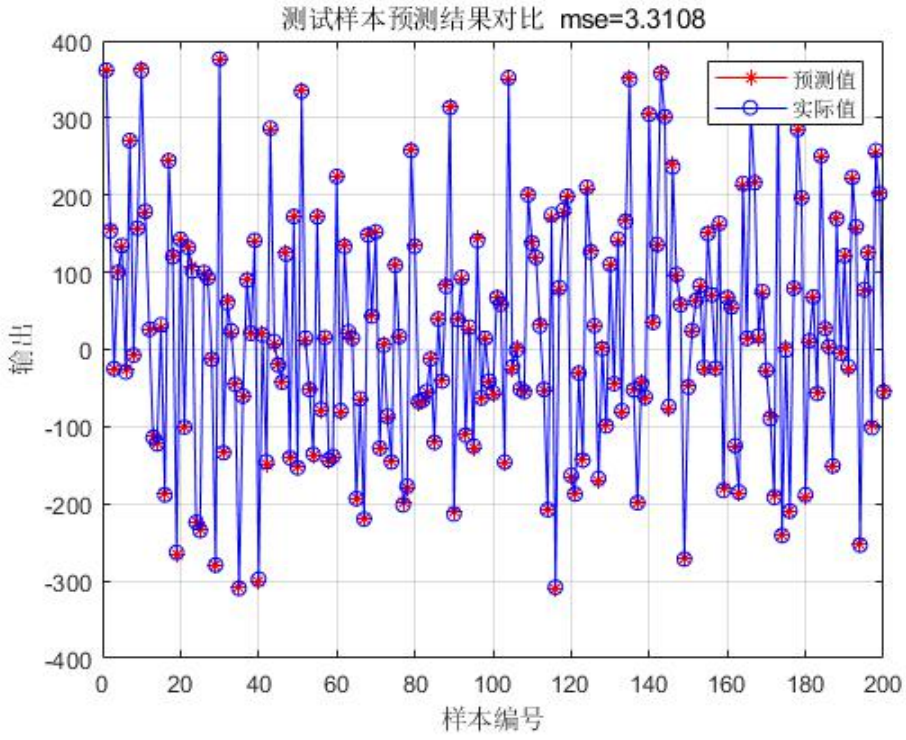
```
xlabel('样本编号')
ylabel('类别')

string_2 = {'训练集预测类型对比';
           ['svm准确率 = ' num2str(train_acc2*100)];
           ['bp准确率 = ' num2str(train_acc1*100)]};
title(string_2)
```

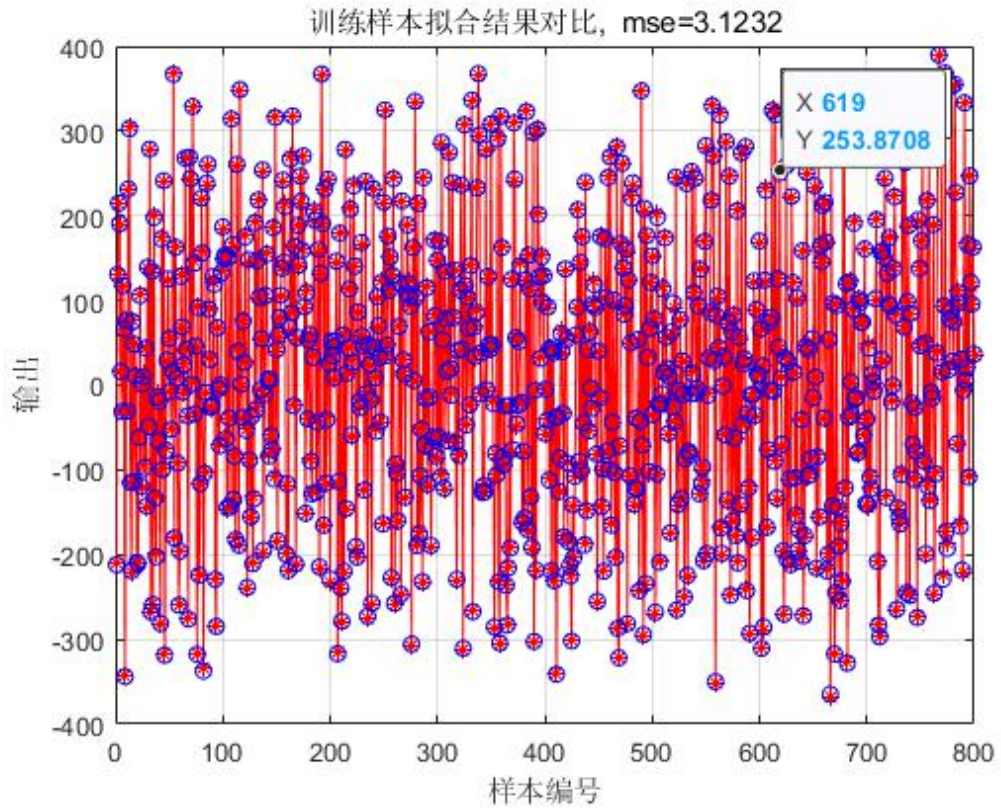
六、数据测试

6.1 三维数据可视化

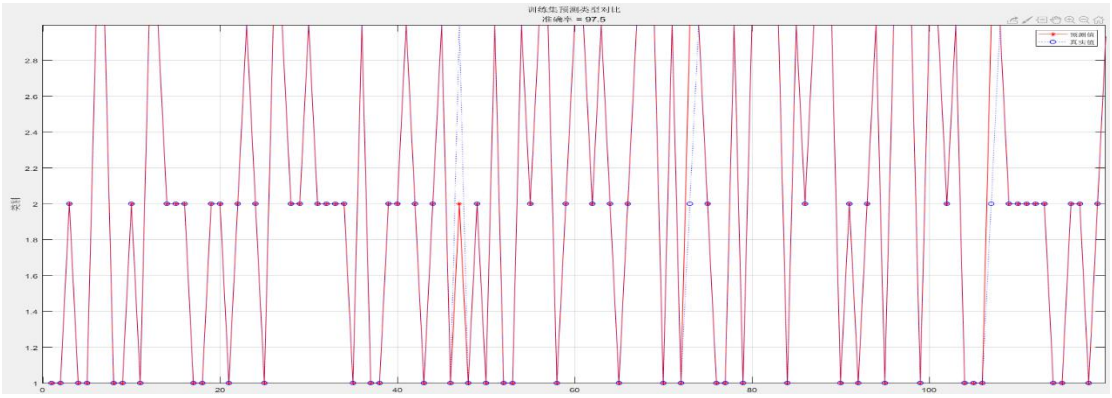


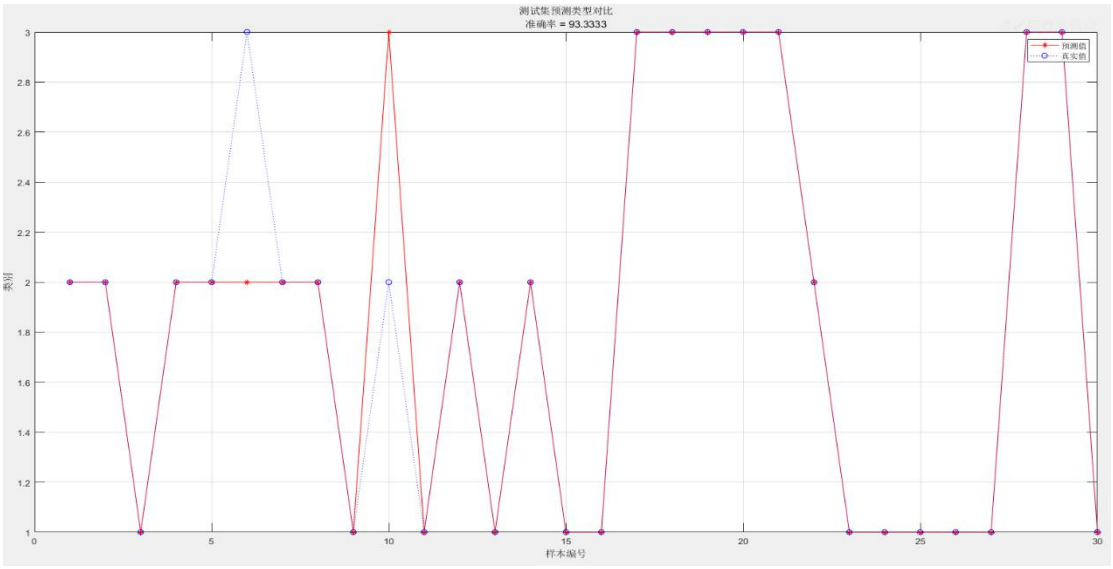
在三维可视化中，我们近似发现这呈一个平面，说明其线性效果好。


测试样本预测结果对比 mse=3.3108
预测值
实际值
输出
样本编号
400
300
200
100
0
-100
-200
-300
-400
0
20
40
60
80
100
120
140
160
180
200


训练样本拟合结果对比，mse=3.1232
X 619
Y 253.8708
输出
样本编号
400
300
200
100
0
-100
-200
-300
-400
0
100
200
300
400
500
600
700
800

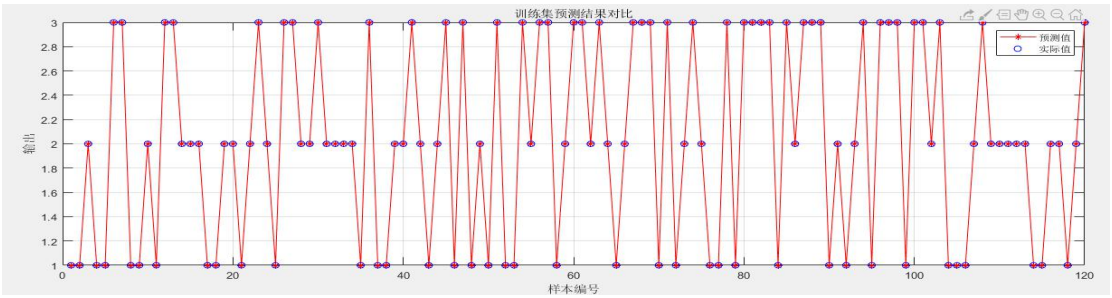
6.2 SVM 训练结果

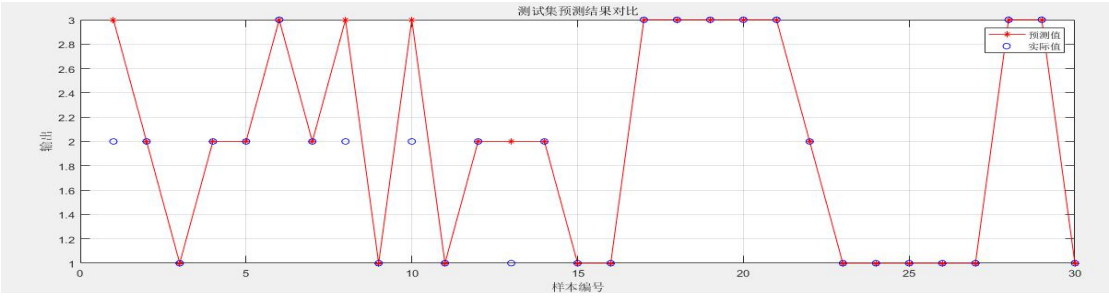




使用原来作为训练的数据集进行测试，其中 120 个数据中有 3 个数据未被测试正确，准确达到 97.5%的，其中蓝线表示 SVM 的训练效果，红线表示数据集的原来类别；测试数据集中，30 个数据中有 2 个数据未被识别成功，**整体比例和原训练数据集大致吻合**。据 CPU 时钟显示，第一次运行时间为 35.621935s。

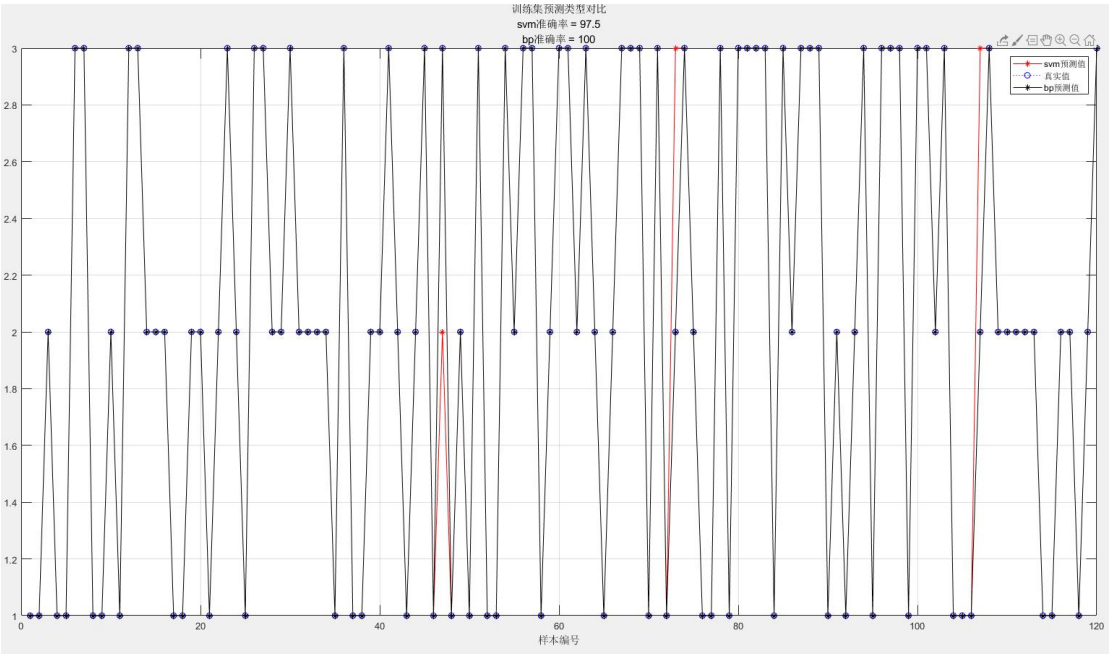
6.3 神经网络训练结果





使用原来作为训练的数据集进行测试，其中 120 个数据测试全部正确；其中蓝线表示 BP 神经网络的训练效果，红线表示数据集的原来类别；测试数据集中，30 个数据中有 4 个数据未被识别成功，说明原数据训练集数据**不具有客观性**，神经网络的各神经元层之间的加权系数仅对原训练集具有一定效果，不具有代表性。应考虑**加大数据训练集的数目**。据 CPU 时钟显示，第一次运行时间为 43.183421s。

6.4 SVM 和神经网络对比



我们发现神经网络在本案例中的**训练效果略好于 SVM**（支持向量机），但在测试数据集中，**神经网络的效果比 SVM 要差**。我们考虑该数据集的样本容量较小，且类别之间的差别不大。在本案例中，神经网络的迭代次数为 500，可以考虑**加大迭代次数以获取更好的训练效果**。考虑从到 matlab 中有集成的神经网络训练工具箱，其运行速度略慢于 SVM。