**模式识别实验**

基于IRIS数据集的感知器和BP算法

3115315017

高榕

# 实验目的：

1. 理解线性分类器的分类原理；
2. 掌握感知器算法，利用他对输入的数据进行分类；
3. 理解BP算法，使用BP算法对输入数据进行分类。

# 实验原理：

1. 感知器算法

感知器算法是通过训练模式的迭代和学习算法，产生线性可分的模式判别函数。感知器算法就是通过对训练模式样本集的“学习”得出判别函数的系数解。在本次实验中，我们主要是采用硬限幅函数进行分类。

感知器的训练算法如下：

设输入矢量其中每一个模式类别已知，它们分别属于ω1类和ω2类。

（1）置步数k=1，令增量ρ为某正的常数，分别赋给初始增广权矢量w(1)的各分量较小的任意值；

（2）输入训练模式，计算判别函数值(k)；

（3）调整增广权重矢量，规则是：

a.如果ω1和(k)0，则w(k+1)=w(k)+ ρ；

b.如果ω2和(k)0，则w(k+1)=w(k)-ρ；

c.如果ω1和(k)0，或ω2和(k)0，则w(k+1)=w(k)

（4）如果k，令k=k+1，返回至（2）。如果k=N，检验判别函数(k)对是否正确分类。如果是，结束；如果不是，令k=1，返回至（2）。

如果训练模式已经符号规范化，即ω2已经乘以-1（包括增广分量1），则校正权矢量的规则可统一为



如果(k)0分类正确，则为第一个表达式，如果(k)0错误分类则为第二个表达式。

在全部模式训练完一轮之后只要还有模式分类错误，则需要进行第二轮迭代，再用全部训练模式训练一次，建立新的权矢量。如果对训练模式还有错分，则进行第三轮迭代依此类推，直至对所有驯良模式均能正确分类为止，此时的w即为所求的权矢量。

1. BP算法

由于硬限幅函数是非可微函数,不能实现多层神经网络的一种有效的LMS学习算法。而BP算法中所用到的是Sigmoid型函数，它既具有完成分类所需的非线性特性，又具有实现LMS算法所需的可微特性。采用S型函数的神经元的输入和输出之间的关系为：



采用了S型函数就可将用于单神经元的LMS学习算法适当推广，得到一种适用于前向多层神经网络的有效学习算法。

我们现在研究一个采用S型函数的前向三层神经网络来说明其原理。

对于训练样本p,它的输入是N维矢量X，X=，网络的第一，二，三层分别包括J，K，M个神经元，它的总输出是一个M维矢量，Y=，第i层到第i+1层之间的权重系数用来表示。可设前向三层神经网络输出各个分量的理想值是,i=0,1,……M-1，而这些分量的实际值是, i=0,1,……M-1，理想值和实际值之间的误差是。各输出误差的平方和可以表示为：



现在我们希望改变网络中的各个加权系数，使得尽可能的减小。为此我们可以采取最陡下降算法的公式来调整权重系数。公式如下：



式中的是学习的步幅，它应随学习过程而变化。

对于通用神经层，它的各个输出与各个输入之间的关系可以表示为：



如果设，则

式中的表示s型函数。我们不难看出上式即为给输入加一个恒等于1的部分，在神经网络中也应相应的加一个节点，而且这个节点的权系数就是这一层网络的阈值。经推倒可得权系数调整的递推公式如下：

 （7）

对输出层：

对隐含层：

现对于三层神经网络有



l=3时（输出层）



l=2时（隐含层）





l=1时（第一层）



其中：  

可见，这一算法的计算过程是先计算第三层（即输出层）的各项“误差分量”，然

后用计算第二层（隐含层）的“等效误差分量” ，最后再用计算第一层（隐含层）的“等效误差分量” 。只要算出这些误差分量，系数调整量即可立即求得。所以，这是一种由输出层向输入层逐步反推的学习算法，故称之为“逆推”学习算法，或BP算法。

# 实验内容：

1. 感知器算法实验

本实验利用感知器算法的原理，随机抽取两类的部分数据，然后，用这两类的其他数据来验证分类是否正确。这里是利用感知器两两分类的原理。实验可得结果如下表。其中r1是输入x1所得的分类正确率，r2是输入x2所得的分类正确率，r3是输入x3所得的分类正确率。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 训练样本数 | 输入不同组数据后的训练次数和分类正确率 | | | | | |
| x1和x2 | | x1和x3 | | x2和x3 | |
| m=5 | r1=100%  r2=100% | k=20  图3.1.1 | r1=100%  r3=100% | k=30  图3.1.6 | r2=98%  r3=50% | k=75  图3.1.11 |
| m=15 | r1=100%  r2=100% | k=45  图3.1.2 | r1=100%  r3=100% | k=30  图3.1.7 | r2=86%  r3=98% | k=150  图3.1.12 |
| m=25 | r1=100%  r2=100% | k=50  图3.1.3 | r1=100%  r3=100% | k=25  图3.1.8 | r2=94%  r3=88% | k=175  图3.1.13 |
| m=35 | r1=100%  r2=100% | k=70  图3.1.4 | r1=100%  r3=100% | k=35  图3.1.9 | r2=82%  r3=100% | k=105  图3.1.14 |
| m=45 | r1=100%  r2=100% | k=135  图3.1.5 | r1=100%  r3=100% | k=45  图3.1.10 | r2=70%  r3=100% | k=135  图3.1.15 |

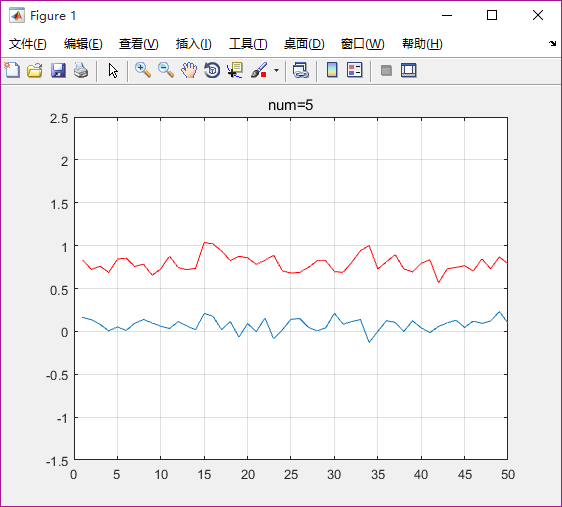
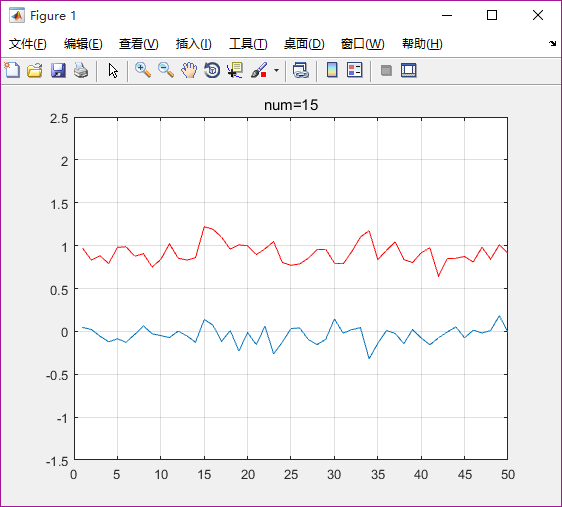
 

图3.1.1 图3.1.2

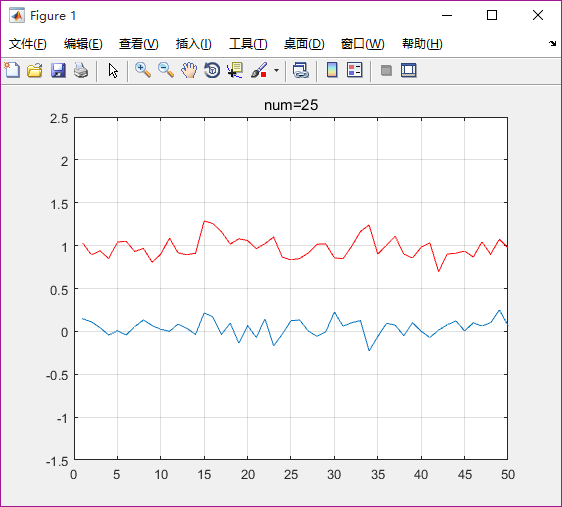
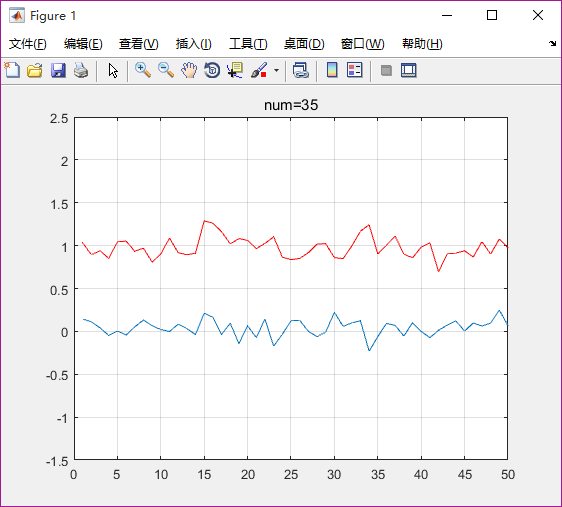
 

图3.1.3 图3.1.4

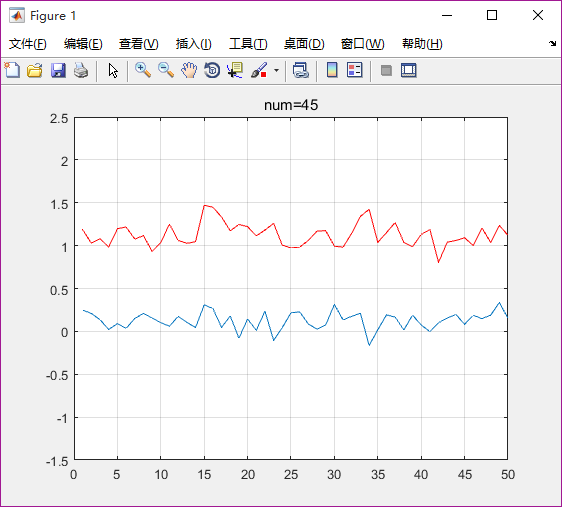
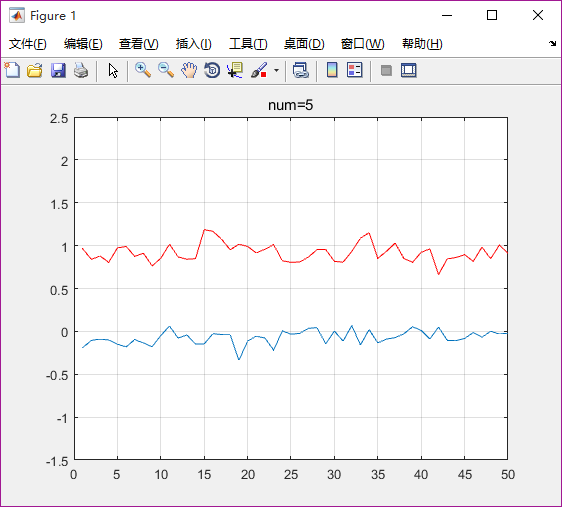
 

图3.1.5 图3.1.6

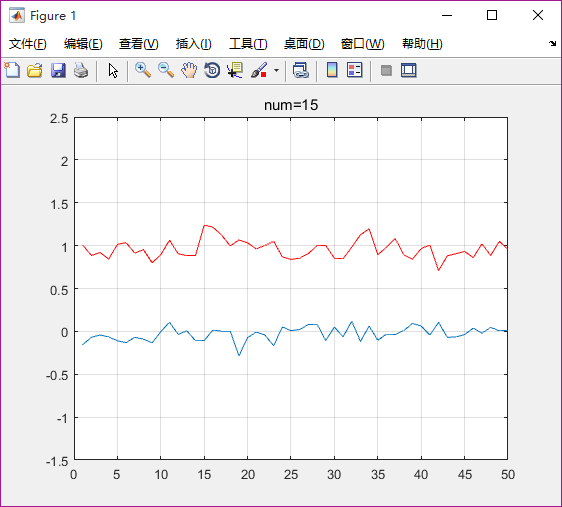
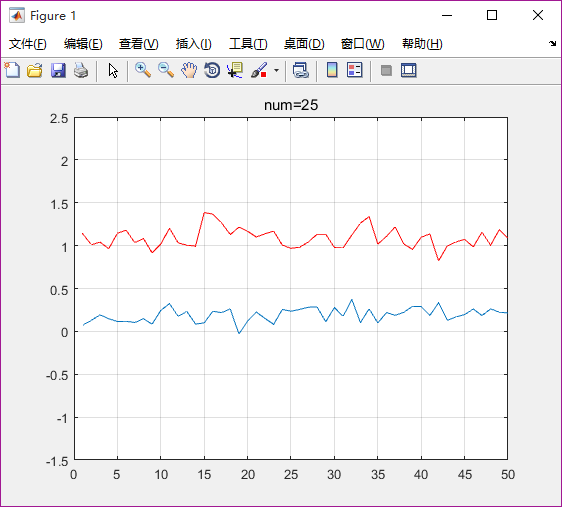
 

图3.1.7 图3.1.8

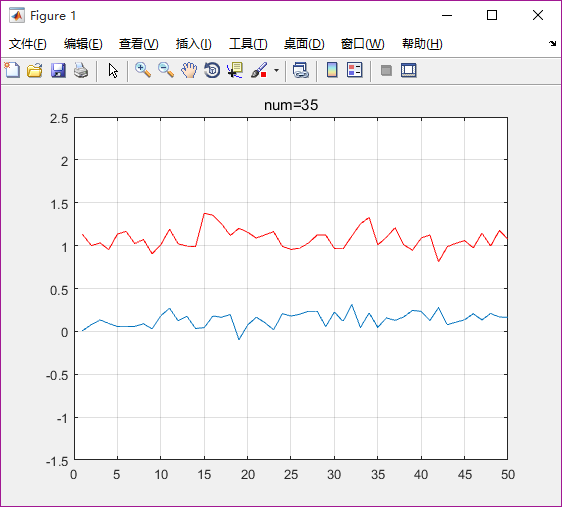
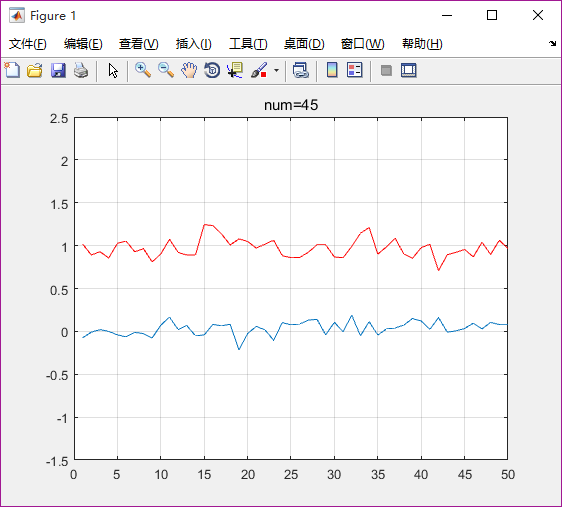
 

图3.1.9 图3.1.10

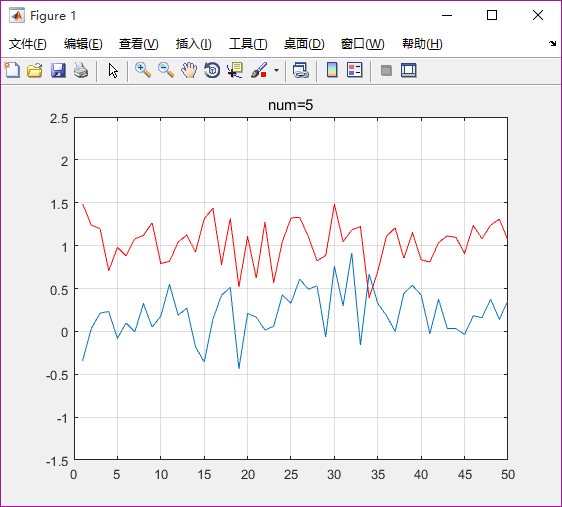
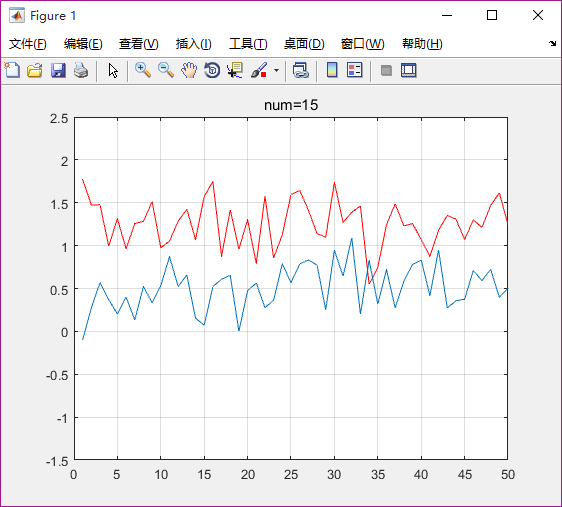
 

图3.1.11 图3.1.12

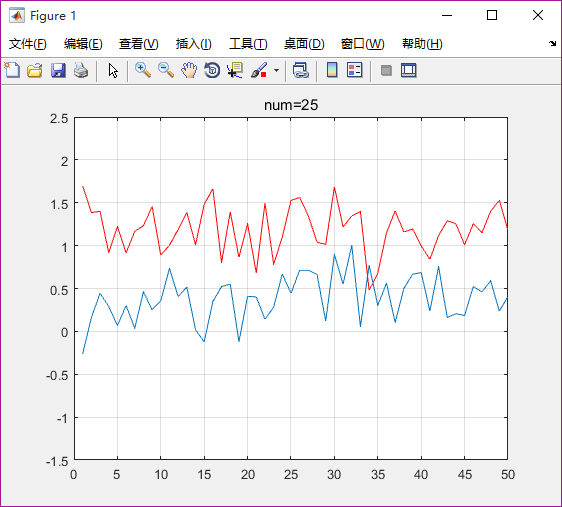
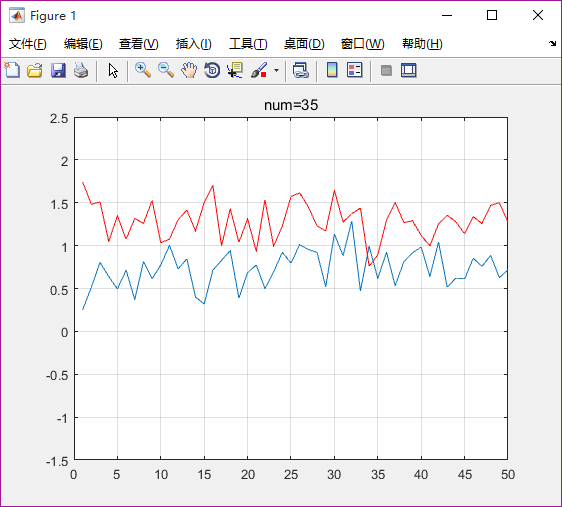
 

图3.1.13 图3.1.14

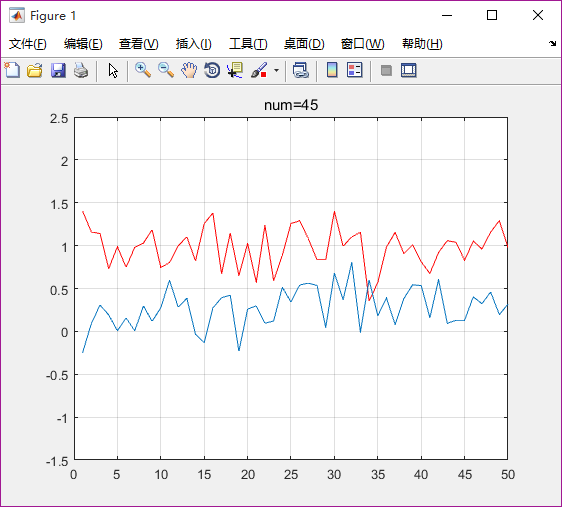


图3.1.15

由实验可以得到，第一类鸢尾花和第二类鸢尾花、第一类鸢尾花和第三类鸢尾花之间是线性可分的，其分类正确率基本上可以达到95％以上。而第二类鸢尾花和第三类鸢尾花之间则是无法分开的，分类效果不明显。而且迭代次数也是基本随着所选训练样本数的增加而增加，但无论怎样，对于第一类和第二类、第一类和第三类之间，都得到了想要的结果，而第二类和第三类之间无法用上述方法准确分类。

1. BP算法实验

（1）异或

利用前向两层神经网络来完成异或分类，输入的样本为x1=[0 1 -1]，x2=[1 0 -1]，x3=[0 0 -1]，x4=[1 1 -1]将这四个样本分成两类。其中，x1和x2是属于w1类，x3和x4属于w2类。第一层（输入层）采用三个神经元，两个输入神经元和一个恒等于1的神经元，第二层（输出层）设置一个神经元，第二层输出的值若小于0.5就表示为w1类，大于0.5就表示w2类。这里采用惯性系数调整算法。训练指数选为a＝0.5。在本实验中，我们记录训练次数和做不同次实验的分类正确率。经过实验可以得到如下表格：

|  |  |  |  |
| --- | --- | --- | --- |
| 实验次数 | p=1 | p=25 | p=50 |
| 训练次数 | k=3686 | k=2970 | k=2942 |
| 正确率 | lv=100% | lv=100% | lv=100% |

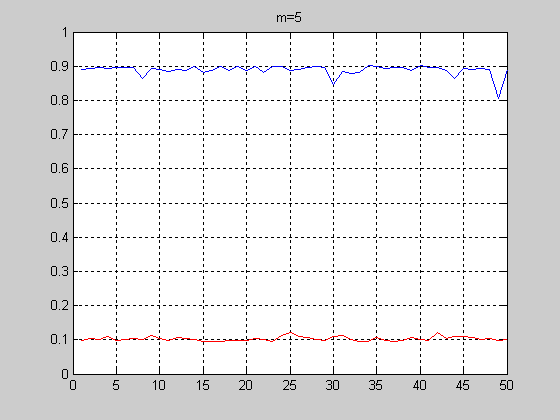
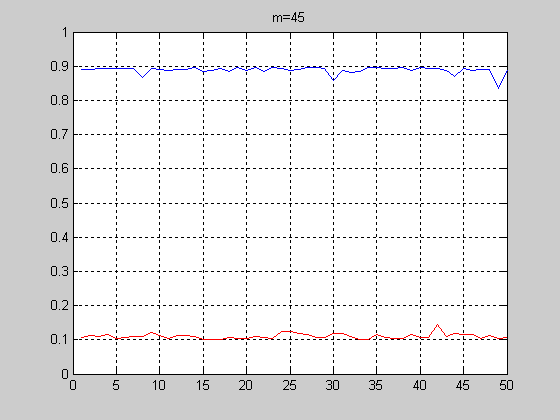
由实验可以看出，分别做1次、25次和50次实验，所得到的分类正确率都是100％，所以说利用前向两层神经网络来完成异或分类问题可以得到很好的分类效果。

（2）BP算法实验

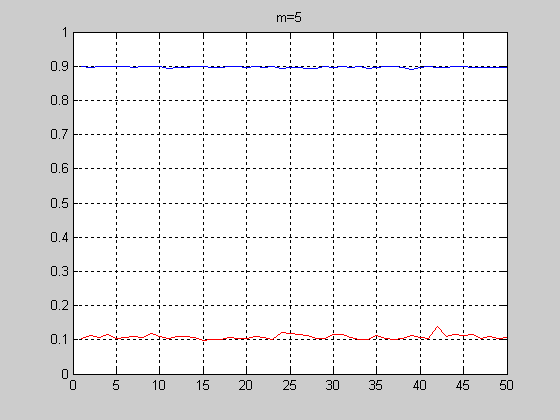
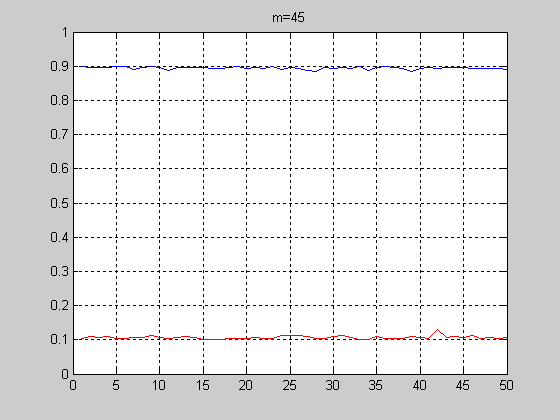
采用前向两层神经网络来实现x1、x2和x3两两之间的分类。分别选取不同个数（m）的样本进行训练，得到相应的权系数，然后再将全部数据代入进行分类，记录训练次数和分类正确率。可得如下结果：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 训练样本数 | 输入不同组数据后的训练次数和分类正确率 | | | | | |
| x1和x2 | | x1和x3 | | x2和x3 | |
| 正确率 | 迭代  系数 | 正确率 | 迭代  次数 | 正确率 | 迭代  次数 |
| m=5 | r1=100%  r2=100% | 1334 | r1=100%  r3=100% | 1336 | r2=92%  r3=86% | 4452 |
| m=15 | r1=100%  r2=100% | 915 | r1=100%  r3=100% | 1276 | r2=92%  r3=98% | 2213 |
| m=25 | r1=100%  r2=100% | 989 | r1=100%  r3=100% | 1131 | r2=92%  r3=96% | 1970 |
| m=35 | r1=100%  r2=100% | 961 | r1=100%  r3=100% | 1044 | r2=92%  r3=98% | 1410 |
| m=45 | r1=100%  r2=100% | 867 | r1=100%  r3=100% | 1039 | r2=90%  r3=100% | 2006 |

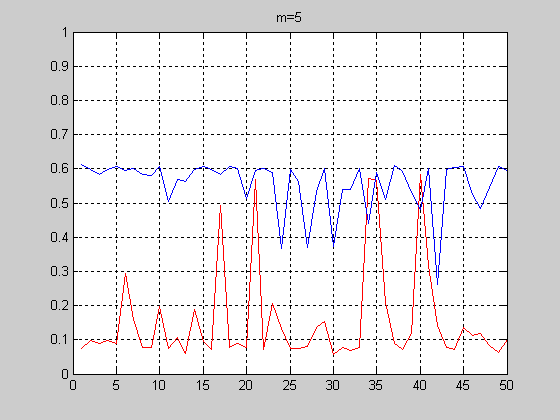
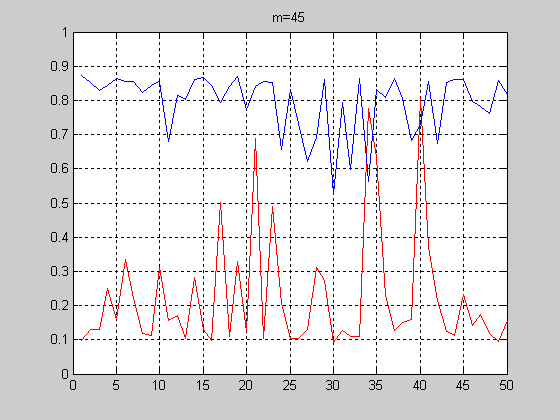
x1和x2之间分别选取训练样本数为5和45时的分类效果：

x1和x3之间分别选取训练样本数为5和45时的分类效果：

x2和x3之间分别选取训练样本数为5和45时的分类情况：

下面是分别进行不同次实验所得正确率的统计结果：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x1和x3  之间 | 实验次数 | 训练样本  个数 | 分类正确次数 | 正确率 |
| p=20 | m=5 | r1=19，r2=20 | 97.5％ |
| m=15 | r1=20，r2=20 | 100％ |
| m=25 | r1=20，r2=20 | 100％ |
| m=35 | r1=19，r2=20 | 97.5％ |
| m=45 | r1=20，r2=20 | 100％ |
| p=50 | m=5 | r1=50，r2=50 | 100％ |
| m=15 | r1=50，r2=50 | 100％ |
| m=25 | r1=49，r2=50 | 99％ |
| m=35 | r1=50，r2=50 | 100％ |
| m=45 | r1=50，r2=50 | 100％ |
| p=100 | m=5 | r1=98，r2=100 | 99％ |
| m=15 | r1=97，r2=100 | 98.5％ |
| m=25 | r1=100,r2=100 | 100％ |
| m=35 | r1=97，r2=100 | 98.5％ |
| m=45 | r1=99，r2=100 | 99.5％ |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x1和x3  之间  x1和x2  之间 | 实验次数 | 训练样本  个数 | 分类正确次数 | 正确率 |
| p=20 | m=5 | r1=20，r2=20 | 100％ |
| m=15 | r1=19，r2=20 | 97.5％ |
| m=25 | r1=19，r2=20 | 97.5％ |
| m=35 | r1=20，r2=20 | 100％ |
| m=45 | r1=20，r2=20 | 100％ |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x1和x3  之间 | 实验次数 | 训练样本个数 | 分类正确次数 | 正确率 | |
| p=50 | m=5 | r1=48，r2=50 | 98％ |
| m=15 | r1=49，r2=50 | 99％ |
| m=25 | r1=49，r2=50 | 99％ |
| m=35 | r1=50，r2=50 | 100％ |
| m=45 | r1=50，r2=50 | 100％ |
| p=100 | m=5 | r1=95，r2=100 | 97.5％ |
| m=15 | r1=99，r2=100 | 99.5％ |
| m=25 | r1=97,r2=100 | 98.5％ |
| m=35 | r1=99，r2=100 | 99.5％ |
| m=45 | r1=98，r2=100 | 99％ |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| x2和x3  之间 | 实验次数 | 训练样本  个数 | 分类正确次数 | 正确率 |
| p=20 | m=5 | r1=4，r2=4 | 20％ |
| m=15 | r1=0，r2=20 | 50％ |
| m=25 | r1=4，r2=11 | 37.5％ |
| m=35 | r1=4，r2=20 | 37.5％ |
| m=45 | r1=0，r2=11 | 27.5％ |
| p=50 | m=5 | r1=3，r2=2 | 5％ |
| m=15 | r1=0，r2=43 | 43％ |
| m=25 | r1=13，r2=21 | 34％ |
| m=35 | r1=1，r2=13 | 14％ |
| m=45 | r1=0，r2=40 | 40％ |

分析实验结果可以得出：无论从一次实验所得结果还是从做多次实验所得的统计结果上来看，利用前向二层神经网络实现x1、x2和x3两两之间的分类，对于x1和x2之间、x1和x3之间的分类效果是比较明显的，可以很好的将两类分开。而对于x2和x3之间却很难用此种办法达到很好的分类效果。

# 实验小结：

本次实验主要是进行线性可分类问题的实验，实验中分别运用感知器算法和BP算法进行x1、x2和x3两两之间的分类，结果得出，只有x1和x2之间及x1和x3之间是线性可分的，而x2和x3之间是无法用这两种办法分开的。本次实验中，在运用BP算法进行x1、x2和x3之间的分类之前，还进行了异或问题的实验，实验中采用两层前向神经网络达到了很好的分类效果，为进行后面的实验打好了基础。

通过本次实验，使我对感知器算法和BP算法有了更加深入的认识，以前在神经网络这门课程中所学的理论问题，在本次实验中通过具体动手进行编程，使得对当时所学知识有了更加深入的认识。在实验中也遇到了各种各样的问题，通过思考解决后，对我以后的学习很有帮助。

# 实验代码：

1. clear all
2. close all
3. clc
4. data = load('iris\_data.txt');
5. iris\_1 = data(1:50,:);
6. iris\_2 = data(51:100,:);
7. iris\_3 = data(101:150,:);
9. pairs = input(’输入训练样本类型 pairs = （举例：类型一三，输入13）');
11. num = input(''输入训练样本个数 num = ');
13. T1=zeros(num,4);
14. T2=zeros(num,4);
15. T3=zeros(num,4);
16. index=randperm(50);
17. for i=1:1:num
18. T1(i,:)=iris\_1(index(i),:);
19. T2(i,:)=iris\_2(index(i),:);
20. T3(i,:)=iris\_3(index(i),:);
21. end
23. W0=zeros(1,4);
24. u=0.02;
25. e=0.01;
26. s=1;
27. k=0;
28. W=W0;
29. if pairs == 13
30. while (s>=e)
31. for i=1:1:num
32. k=k+1;
33. if rem(k,2)==1
34. err=1-W\*T1(i,:)';
35. W=W+u\*err\*T1(i,:);
36. else err=0-W\*T3(i,:)';
37. W=W+u\*err\*T3(i,:);
38. end
39. pf(k)=err^2;
40. s=pf(k);
41. end
42. end
43. elseif pairs == 12
44. while (s>=e)
45. for i=1:1:num
46. k=k+1;
47. if rem(k,2)==1
48. err=1-W\*T1(i,:)';
49. W=W+u\*err\*T1(i,:);
50. else err=0-W\*T2(i,:)';
51. W=W+u\*err\*T2(i,:);
52. end
53. pf(k)=err^2;
54. s=pf(k);
55. end
56. end
57. elseif pairs == 23
58. while (s>=e)
59. for i=1:1:num
60. k=k+1;
61. if rem(k,2)==1
62. err=1-W\*T2(i,:)';
63. W=W+u\*err\*T2(i,:);
64. else err=0-W\*T3(i,:)';
65. W=W+u\*err\*T3(i,:);
66. end
67. pf(k)=err^2;
68. s=pf(k);
69. end
70. end
71. else
72. disp('样本类型输入错误，程序退出');
73. return;
74. end
75. k

78. r1=0;r2=0;
79. if pairs == 12
80. for i=1:1:50
81. y1(i)=W\*iris\_1(i,:)';
82. if y1(i)>0.5
83. r1=r1+1;
84. end
85. y2(i)=W\*iris\_2(i,:)';
86. if y2(i)<=0.5
87. r2=r2+1;
88. end
89. end
90. elseif pairs == 13
91. for i=1:1:50
92. y1(i)=W\*iris\_1(i,:)';
93. if y1(i)>0.5
94. r1=r1+1;
95. end
96. y2(i)=W\*iris\_3(i,:)';
97. if y2(i)<=0.5
98. r2=r2+1;
99. end
100. end
101. else
102. for i=1:1:50
103. y1(i)=W\*iris\_2(i,:)';
104. if y1(i)>0.5
105. r1=r1+1;
106. end
107. y2(i)=W\*iris\_3(i,:)';
108. if y2(i)<=0.5
109. r2=r2+1;
110. end
111. end
112. end
113. plot([1:50],y1,'r');
114. axis([0,50,-1.5,2.5]);
115. title(['num=',num2str(num)]);
116. hold on
117. plot([1:50],y2);
118. grid on
119. r1=r1/50
120. r2=r2/50
121. close all;
122. clc;
123. data = load('iris\_data.txt');
124. iris\_1 = data(1:50,:);
125. iris\_2 = data(51:100,:);
126. iris\_3 = data(101:150,:);
127. m=input('训练样本数');
128. pairs = input('输入训练样本类型 pairs = （举例：类型一三，输入13）');
130. T1=zeros(m,4);
131. T2=zeros(m,4);
132. T3=zeros(m,4);
133. index=randperm(50);
134. for i=1:1:m
135. x1(i,:)=iris\_1(index(i),:);
136. x2(i,:)=iris\_2(index(i),:);
137. x3(i,:)=iris\_3(index(i),:);
138. end
140. p=0;r11=0;r22=0;
141. while p<50
142. a=0.5;
143. pf=1;
144. w1=rand(4,5);
145. w2=rand(6,1);
146. k=1;tt=1;
147. while pf>0.000001
148. if k>m
149. k=1;
150. end
151. if pairs == 12
152. if mod(k,2)==1
153. g1=x1(k,:);
154. d=0.1;
155. end
156. if mod(k,2)==0
157. g1=x2(k,:);
158. d=0.9;
159. end
160. elseif pairs == 13
161. if mod(k,2)==1
162. g1=x1(k,:);
163. d=0.1;
164. end
165. if mod(k,2)==0
166. g1=x3(k,:);
167. d=0.9;
168. end
169. elseif pairs == 23
170. if mod(k,2)==1
171. g1=x2(k,:);
172. d=0.1;
173. end
174. if mod(k,2)==0
175. g1=x3(k,:);
176. d=0.9;
177. end
178. else
179. disp('样本输入类型错误，程序退出');
180. return;
181. end
183. g2=g1\*w1;
184. o1=1./(1+exp(-g2));
185. o1\_y=[o1 -1];
186. g3=o1\_y\*w2;
187. o2=1./(1+exp(-g3));
188. det2=a\*2\*(d-o2)\*o2\*(1-o2)\*o1\_y;
189. w2=w2+det2';
190. s=2\*(d-o2)\*o2\*(1-o2);
191. w22=w2';
192. det1=a\*g1'\*(s\*w22(:,1:5).\*o1.\*(1-o1));
193. w1=w1+det1;
194. err=d-o2;
195. pf=err^2;
196. k=k+1;
197. tt=tt+1;
198. end
200. r1=0;r2=0;
201. for j=1:1:50
202. if pairs == 12
203. t1=iris\_1(j,:)\*w1;
204. elseif pairs == 13
205. t1=iris\_1(j,:)\*w1;
206. else
207. t1=iris\_2(j,:)\*w1;
208. end
209. y1=1./(1+exp(-t1));
210. y1\_y=[y1 -1];
211. I2=y1\_y\*w2;
212. y2(j)=1./(1+exp(-I2));
213. if y2(j)<0.5
214. r1=r1+1;
215. end
216. end
217. lv1=r1/50
218. if r1==50
219. r11=r11+1;
220. end
221. for j=1:1:50
222. if pairs == 12
223. t1=iris\_2(j,:)\*w1;
224. elseif pairs == 13
225. t1=iris\_3(j,:)\*w1;
226. else
227. t1=iris\_3(j,:)\*w1;
228. end
229. y1=1./(1+exp(-t1));
230. y1\_y=[y1 -1];
231. I2=y1\_y\*w2;
232. y22(j)=1./(1+exp(-I2));
233. if y22(j)>0.5
234. r2=r2+1;
235. end
236. end
237. lv2=r2/50
238. if r2==50
239. r22=r22+1;
240. end
241. p=p+1
242. end
243. r11/50
244. r22/50
245. plot([1:50],y2,'r');
246. axis([0,50,0,1]);
247. title(['m=',num2str(m)]);
248. hold on
249. plot([1:50],y22);
250. grid on
251. lv=(r11+r22)/100