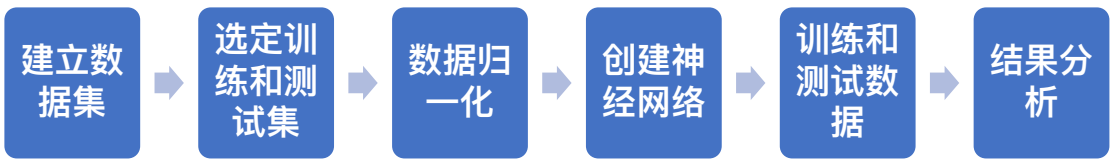
神经网络实现 Iris 数据集多类分类

1. 背景

Iris 数据集也称为鸢尾花数据集，该数据包含 150 个样本，有三种类别，分别为山鸢尾、变色鸢尾和维吉尼亚鸢尾，每个样本有 4 个特征属性，四个特征属性分别为花萼宽度、花萼宽度、花瓣宽度、花瓣长度。我们将 iris 数据集的 50%作为训练样本，另外 50%作为测试样本，其中每组花中各有 25 个样本，最后我们得到结果的准确率为 95%左右。软件为 matlab，BP 神经网络算法，由于神经网络的知识蛮多的，我也无法写全,所以可以参考一下吴恩达机器学习和周志华机器学习的内容，对神经网络有一定的了解。

2. 流程图



3. 完整代码

```
%读取训练数据
[f1,f2,f3,f4,class] = textread('trainData.txt' , '%f%f%f%f%f',150);
%读取测试数据
[t1 t2 t3 t4 c] = textread('testData.txt' , '%f%f%f%f%f',150);
% [A,B,C,...] = textread(filename,format,N)，将指定的 foemat 文件读取到 A，B，C 中，直至 150 个样本读取完成，读取完成后 f 中分别代表数据集的某一个属性值得全部数据，class 代表结果

%特征值归一化，归一化到【-1-1】之间
[input,minI,maxI] = premnmx( [f1 , f2 , f3 , f4 ]') ;

%测试数据归一化
testInput = tramnmx ( [t1,t2,t3,t4]' , minI, maxI )

%构造输出矩阵
s = length( class ) ;
output = zeros( s , 3   ) ;
for i = 1 : s
    output( i , class( i )   ) = 1 ;
```

```
end

%创建神经网络，这里的代码采用的旧式用法，发现编译器会报问题，新旧版的结果也会造成比较大的差异，
net = newff( minmax(input) , [10 3] , { 'logsig' 'purelin' } , 'traingdx' ) ;
%用新版的代替上面的
%net = newff( input,output', [10] , { 'logsig' 'purelin' } , 'traingdx' )
% net.divideFcn = '';

%设置训练参数
net.trainparam.show = 50 ;
net.trainparam.epochs = 500 ;
net.trainparam.goal = 0.01 ;
net.trainParam.lr = 0.01 ;

%开始训练
net = train( net, input , output' ) ;
%仿真
Y = sim( net , testInput )

%统计识别正确率
[s1 , s2] = size( Y ) ;
hitNum = 0 ;
for i = 1 : s2
   [m , Index] = max( Y( : ,  i ) ) ;
      if( Index   == c(i)     )
            hitNum = hitNum + 1 ;
      end
end

sprintf('识别率是  %3.3f%%',100 * hitNum / s2 )
```

4. 代码分析

1) **再谈归一化，对上篇文章进行补充，为什么要归一化处理？（为了让语言更为准确一点，我又在网上搜索为什么要归一化，结果如下）**
   a) 输入数据的单位不一样，有些数据的范围可能特别大，导致的结果是神经网络收敛慢、训练时间长
   b) 数据范围大的输入在模式分类中的作用可能会偏大，而数据范围小的输入作用就可能会偏小
   c) 由于神经网络输出层的激活函数的值域是有限制的，因此需要将网

络训练的目标数据映射到激活函数的值域。例如神经网络的输出层若采用 S 形激活函数，由于 S 形函数的值域限制在(0,1)，也就是说神经网络的输出只能限制在(0,1)，所以训练数据的输出就要归一化到[0,1]区间

d) S 形激活函数在(0,1)区间以外区域很平缓，区分度太小。例如 S 形函数 f(X)在参数 a=1 时，f(100)与 f(5)只相差 0.0067
这里我找到了 matlab 中各种归一化函数用法

```
premnmx、tramnmx、postmnmx、mapminmax
premnmx函数用于将网络的输入数据或输出数据进行归一化，归一化后的数据将分布在[-1,1]区间内。
premnmx语句的语法格式是：[Pn,minp,maxp,Tn,mint,maxt]=premnmx(P,T)，其中P，T分别为原始输入和输出数据。

    在训练网络时如果所用的是经过归一化的样本数据，那么以后使用网络时所用的新数据也应该和样本数据接受相同的预处
tramnmx语句的语法格式是：[PN]=tramnmx(P,minp,maxp)
其中P和PN分别为变换前、后的输入数据，maxp和minp分别为premnmx函数找到的最大值和最小值。

网络输出结果需要进行反归一化还原成原始的数据，常用的函数是：postmnmx。
postmnmx语句的语法格式是：[PN] = postmnmx(P,minp,maxp)
其中P和PN分别为变换前、后的输入数据，maxp和minp分别为premnmx函数找到的最大值和最小值。
还有一个函数是mapminmax，该函数可以把矩阵的每一行归一到[-1 1].
mapminmax语句的语法格式是：[y1,PS] = mapminmax(x1)
其中x1 是需要归一的矩阵 y1是结果。
当需要对另外一组数据做归一时，就可以用下面的方法做相同的归一了
y2 = mapminmax('apply',x2,PS)
当需要把归一的数据还原时，可以用以下命令：
x1_again = mapminmax('reverse',y1,PS)

prestd、poststd、trastd
prestd归一到单位方差和零均值。
pminp和maxp分别为P中的最小值和最大值。mint和maxt分别为T的最小值和最大值。
```

复制

2)

a) 使用 matlab 创建前馈神经网络，我们会使用到三个函数，newff，train，sim，用上述代码会有提示，让我们使用新的 newff，新旧识别效果也有点不同，旧版的识别正确率率高一点，新版相对于旧版需要设置一个 net.divideFcn，我在网上百度了一下 https://www.cnblogs.com/litthorse/p/9276455.html

b) 参数设置

net.trainparam.goal ：神经网络训练的目标误差

net.trainparam.show ： 显示中间结果的周期

net.trainparam.epochs ：最大迭代次数

net.trainParam.lr ： 学习率

c) Train 用法

语法：[net, tr, Y1, E] = train(net, X, Y)

参数：

X：网络实际输入

Y：网络应有输出

tr：训练跟踪信息

Y1：网络实际输出

E：误差矩阵

d) Sim 语法

语法：Y=sim(net,X)

参数：

net：网络

X：输入给网络的K×N矩阵，其中K为网络输入个数，N为数据样本数

Y：输出矩阵Q×N，其中Q为网络输出个数

e) 结果

程序的识别率最终达到了 95%左右

5. 小结

报告前期，我对神经网络首先进行了一定时间的学习，主要包括看了吴恩达的机器学习（网易云课堂），也参考了周志华的《机器学习》一书，对于神经网络有了一定的了解，才开始去选定数据集学习，由于之前已经写了 SVM，我发现多类问题其实还是差不多的，首先都要对数据进行一定的处理，主要的差异就是可能算法的不同，对数据归一化的要求也有所不同。