MLPC

MLPC，多层感知器分类器，是一种基于前馈人工神经网络的分类器，多层感知器是一种多层的前馈神经网络模型。

每个层完全连接到网络的下一层。

输入层中的节点表示输入数据。

所有其他节点通过输入与节点的权重w和偏置b的线性组合，并应用激活函数，将输入映射到输出。

前馈型神经网络，指其从输入层开始只接收前一层的输入，并把计算结果输出到后一层，并不会给前一层有所反馈，整个过程可以使用有向无环图来表示。该类型的神经网络由三层组成，分别是输入层，一个或多个隐层，输出层，如图1所示：

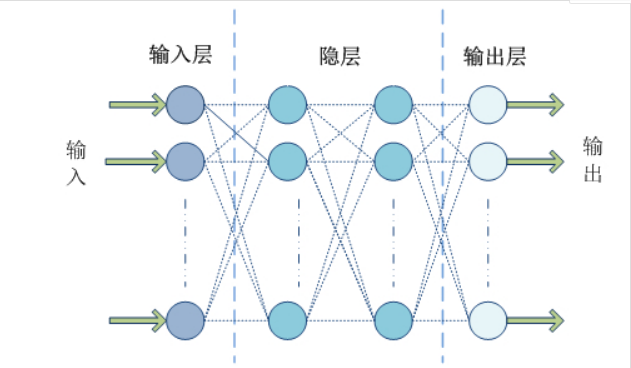


图1神经网络

1. 神经元模型

神经网络是由许多的神经元连接而成，神经元的计算方式是神经网络模型的最重要部分。

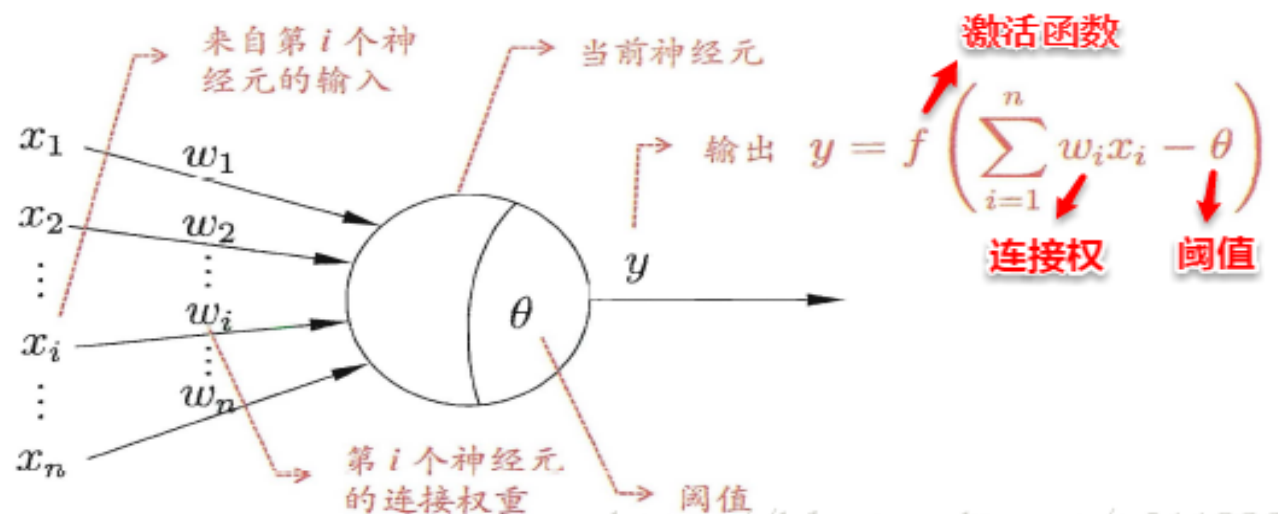


图2神经元模型

表示第i个相邻神经元向本神经元发送的输入量。

表示的权重，表示对它的影响。

表示当前神经元的增量。

假设当前神经元的初始值为0，增量减阈值大于0，则神经元处于兴奋状态，用激活函数表示它向下一个神经元发送的输入。

2.常用激活函数

sigmoid函数：



把较大范围的输入值挤压到（0，1）输出值范围内。

Sigmoid函数的一阶导数为：



缺点:指数运算会耗费大量计算资源。容易出现梯度消失现象。值不是零均值，导致神经网络的收敛速度缓慢。

Tanh函数：



优点：tanh是0均值的，网络的收敛速度要比sigmoid快，实际应用中tanh会比sigmoid更好。

缺点：指数运算会费大量计算资源,容易出现梯度消失现象。

3.多层网络模型

MLPC采用了BP(反向传播，Back Propagation) 算法，BP算法的学习目的是对网络的连接权值进行调整，使得调整后的网络对任一输入都能得到所期望的输出。BP 算法名称里的反向传播指的是该算法在训练网络的过程中逐层反向传递误差，逐一修改神经元间的连接权值，以使网络对输入信息经过计算后所得到的输出能达到期望的误差（模型如图3）。

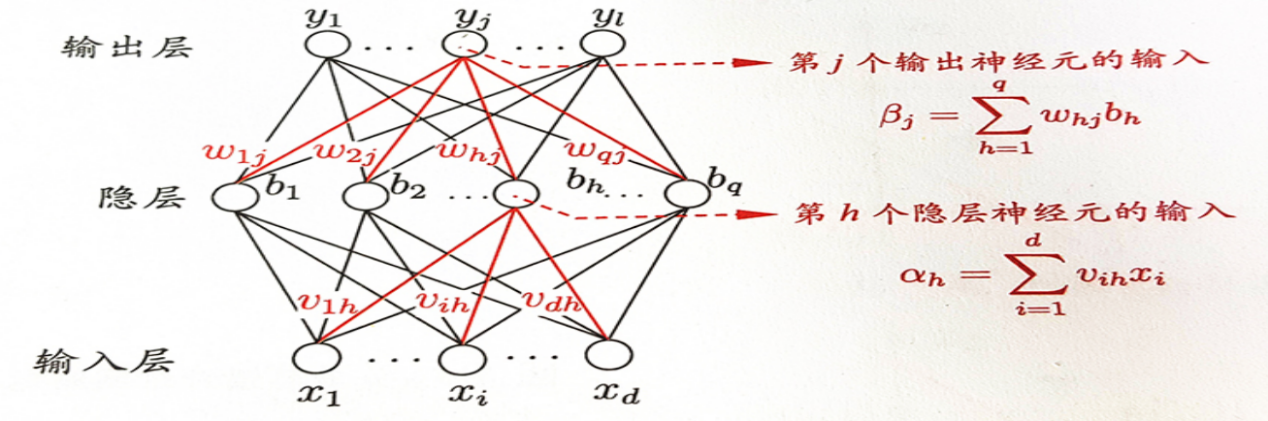


图3多层神经网络模型

3.1网络训练简述

根据训练数据来调整神经元之间的连接权,以及每个功能神经元的阈值；输出值与标记值比较，若有误差，将误差反向由输出层向输入层传播，逐层反向传递误差；在传递过程中，利用梯度下降算法对神经元权值进行调整，逐一修改神经元之间的连接权值。

目的是使网络对输入信息经过计算后所得到的输出能达到期望的误差。

训练集：



其中，，一个d维向量,

,一个l维向量，

d表示输入层神经元的个数，

l表示输出层神经元的个数，

q表示隐含层神经元的个数。

对于训练样例，神经网络的输出为。

其中，，其中f使sigmoid激活函数，

为输出层第j个神经元的输入，

表示输出层第j个神经元的阈值，

表示隐含层第h个神经元的输入与输出层第j个神经元的连接权，

表示第h个隐含层神经元的输出。



其中，表示第h个隐含层神经元的输入，

表示输入层第i个神经元与隐含层第h个神经元的连接权，

表示隐含层第h个神经元的阈值。

3.2损失函数

度量神经网络的实际输出与预计输出之间的差距。比如，神经网络的输出是一个l维向量，预计的输出是也是一个l维向量。

损失函数表示为：



使用梯度下降法更新连接权，使它不断下降，直至小于规定的阈值。

4.参数介绍

Spark的多层感知器隐层神经元使用sigmoid函数作为激活函数，输出层使用的是softmax函数。

sigmoid函数：



softmax函数：



其中输出层的节点数量N对应类别的数量。

多层感知器分类器采用反向传播学习模型（BP算法）训练神经网络，并用于优化的逻辑损失函数和L-BFGS作为优化程序。

MLPC可调的几个重要参数:

featuresCol：输入数据 DataFrame 中指标特征列的名称。

labelCol：输入数据 DataFrame 中标签列的名称。

layers：这个参数是一个整型数组类型，第一个元素需要和特征向量的维度相等，最后一个元素需要训练数据的标签数相等，如 2 分类问题就写 2。中间的元素有多少个就代表神经网络有多少个隐层，元素的取值代表了该层的神经元的个数。

maxIter：优化算法求解的最大迭代次数。默认值是 100。

predictionCol:预测结果的列名称。

5.神经网络分析

5.1数据展示

使用数据为信用相关的数据，大概进行一个信用判断（1或者0），数据有21列，如creditability，balance，duration，history，purpose等。在进行分类之前，需要选取特征，我们使用20列数据当作特征项，进行分类，前6行数据如表1。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| balance | duration | history | purpose | amount | savings |
| 1 | 18 | 4 | 2 | 1049 | 1 |
| 1 | 9 | 4 | 0 | 2799 | 1 |
| 2 | 12 | 2 | 9 | 841 | 2 |
| 1 | 12 | 4 | 0 | 2122 | 1 |
| 1 | 12 | 4 | 0 | 2171 | 1 |
| 1 | 10 | 4 | 0 | 2241 | 1 |

表1 前6行数据展示

5.2特征选择

得到数据之后我们进行神经网络搭建，我们将原21列数据中的20列作为属性，形成特征，进行分类划分；我们将所有样本数据进行切分，前百分之七十的数据作为训练数据，后百分之三十的数据作为测试数据，创建一个输入数据维数为20，网络层数为50，每层20个神经元，结果维数为2（因为我们的结果就是对两个类别的分类情况），会从训练数据的学习中，得到一个训练好的模型，然后对测试数据进行预测，与之前的类别进行对比，得到准确率。

我们拿到初始数据之后要得到形成特征，我们将20个不同属性数据放在一个列表里，当作一个特征。部分特征（前5行）数据展示如表2。

|  |
| --- |
| features |
| [1.0,18.0,4.0,2.0,1049.0,1.0...] |
| [1.0,9.0,4.0,0.0,2799.0,1.0...] |
| [2.0,12.0,2.0,9.0,841.0,2.0...] |
| [1.0,12.0,4.0,0.0,2122.0,1.0...] |
| [1.0,12.0,4.0,0.0,2171.0,1.0...] |

表2 前5行数据特征

5.3打标签

进行样本训练前，需要给每一行数据打上标签，可以明显的辨别样本所属的类别，然后再才进行分类。前3列标签数据与原数据情况展示（表3）。

|  |  |
| --- | --- |
| creditability | label |
| 1.0 | 0.0 |
| 1.0 | 0.0 |
| 1.0 | 0.0 |

表3 前3列数据标签

因为会将所有样本分为两类，而前面样本大多数都是相同的，所以在观察数据前几列的时候，所打标签都会为0，后面类别不同的，比如creditability=0的样本，他的标签label=1.0。

5.4分类结果

由训练数据对神经网络进行模型权重设置了以后，得到一个误差最小的模型，当作分类模型使用，对测试数据进行分类，会得到一个rawPrediction，就是暂时计算的分类数据，它是数值的，还会得到一个probabilit，就是对暂时分类情况的可能性判定数据，如果我们得到一个样本暂时分为了1类，但是可靠性小，说明这个样本为1类的可能性低，就将这个样本归为0类。还有prediction数据，就是经过模型计算得出的分类情况，具体样本分类展示部分数据（表4）。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| creditability | label | rawPrediction | probability | prediction |
| 0.0 | 1.0 | [-1.3512775627005...] | [0.39662629679080...] | 1.0 |
| 0.0 | 1.0 | [-1.3895843137397...] | [0.37925422966846...] | 1.0 |
| 0.0 | 1.0 | [-1.2885933905290...] | [0.42551400438286...] | 1.0 |
| 0.0 | 1.0 | [-0.6285927056891...] | [0.71641843222291...] | 0.0 |
| 0.0 | 1.0 | [-0.9438403556128... | [0.58603888422544... | 0.0 |

表4 前5列数据预测情况

prediction数据主要与label数据进行比较，如果相同，那说明分类正确，否则预测样本分类错误。从表4可以发现前3个样本标签和预测的类别是相同的，说明这3个样本分类正确，但是后两个却不相同，说明后两个样本分类错误。

最后我们可以得到一个所有预测样本数据分类的总情况数据，Test set accuracy=0.7885,表明我们的预测数据样本分类准确率为78.85%。

5.5调整网络结构

第一次网络搭建的时候，为了避免耗费大量的时间，我们构建了一个简单一点的神经网络，网络层数和神经元个数都比较少，输入神经元为20个，输入神经元为了接受数据维数，因为我们选取20个属性作为特征，所以输入神经元个数是固定的；初始神经网络层数为10，每层神经元个数为20，当时设想是每一个属性对应每层神经元，输出神经元为2；得到准确率为0.6845。

然后固定了神经网络的层数，为10层，单一的调整了神经元个数，得到的数值如表5。

|  |  |
| --- | --- |
| Layers=[20,10,a,2] | Test set accuracy |
| [20,10,4,2] | 0.6845 |
| [20,10,20,2] | 0.6845 |
| [20,10,50,2] | 0.6845 |
| [20,10,100,2] | 0.6845 |

表5 10层网络的神经元变化影响

发现神经网络层数为10的时候，我们单一的改变每一层神经元的个数对准确率没有影响。我们认为神经网络层太少，所以因为神经元数量变化导致模型精确度没有提升，所以重新固定神经网络层数为50（表6）。

|  |  |
| --- | --- |
| Layers=[20,50,a,2] | Test set accuracy |
| [20,50,4,2] | 0.7706 |
| [20,50,20,2] | 0.7885 |
| [20,50,50,2] | 0.6845 |
| [20,50,100,2] | 0.6881 |

表6 50层网络的神经元变化影响

当神经网络层数固定为50层的时候，因为每层神经元的数目变化，影响到了神经网络模型的精确度，当神经网络层数为50层的时候，每层网络神经元个数为20个的网络模型最优，准确率为78.85%。

对神经网络每层神经元的单一变化影响得出结论：当每层神经元的数量增加时，模型的精确度不一定变化，当每层神经元个数影响模型精确度时，每层神经元的增加，模型精确度不一定变优。而且单一的改变神经网络的某一层神经元个数或者单一的改变网络层数对该数据所形成的模型的精确度影响不大。我们对网络层数和每层神经元个数一起做改变，得到表7。

|  |  |
| --- | --- |
| Layers=[20,a,b,2] | Test set accuracy |
| [20,10,20,2] | 0.6845 |
| [20,50,20,2] | 0.7885 |
| [20,50,100,2] | 0.6881 |
| [20,80,20,2] | 0.7562 |
| [20,80,100,2] | 0.7168 |

表7 参数改变影响

我们得到layers=[20,50,10,2]和layers=[20,50,20,2]时，神经网络模型精确度最高，数值为0.7885。可以从表7第一行和第三行发现，当神经网络模型复杂了很多的时候，精确度的变化根本不大，但是神经网络变得复杂的时候，对机器的占用却很大；而且神经网络变得更复杂，准确性也不一定变好。

### MPLC模型处理客户流失分类问题结果对比分析

2.1 选取数据

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 性别 | 年龄 | 平均菜品数 | 距今时间 | 消费次数 | 平均消费金额 | 是否流失 |
| 男 | 43 | 26 | 501 | 3 | 1048 | 1 |
| 男 | 58 | 19 | 594 | 2 | 650.5 | -1 |
| 女 | 22 | 12 | 615 | 2 | 534.5 | -1 |
| 男 | 43 | 26 | 501 | 3 | 1048 | 1 |

特征包括：性别、平均菜品数、距今时间、消费次数、平均消费金额，

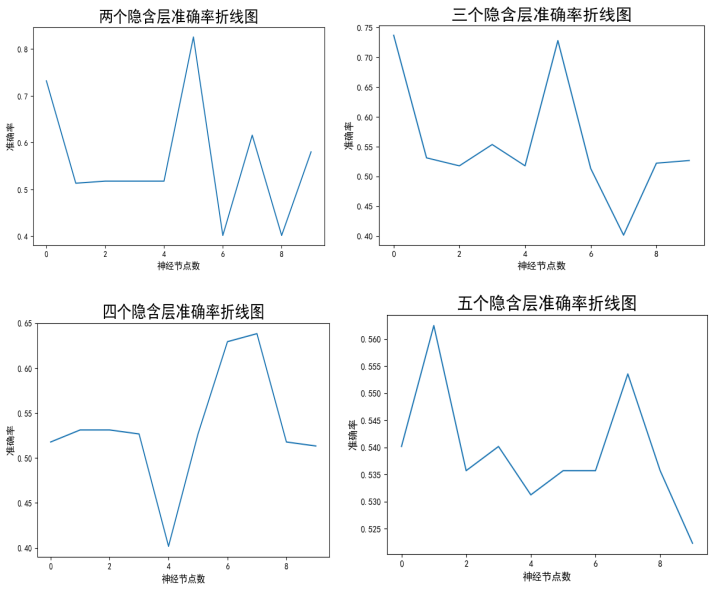
客户类别：未流失（1）、准流失（0）、已流失（-1）

2.2 不同模型准确率对比

|  |  |  |
| --- | --- | --- |
| 隐含层数目 | 神经节点数 | 准确率 |
| 2 | （11,10)-(2,1) | ['0.7321428571428571', '0.5133928571428571', '0.5178571428571429', '0.5178571428571429', '0.5178571428571429', '0.8258928571428571', '0.4017857142857143', '0.6160714285714286', '0.4017857142857143', '0.5803571428571429'] |
| 3 | (12,11,10)-(3,2,1) | ['0.7366071428571429', '0.53125', '0.5178571428571429', '0.5535714285714286', '0.5178571428571429', '0.7276785714285714', '0.5133928571428571', '0.4017857142857143', '0.5223214285714286', '0.5267857142857143'] |
| 4 | (13,12,11,10)-(4,3,2,1) | ['0.5178571428571429', '0.53125', '0.53125', '0.5267857142857143', '0.4017857142857143', '0.5267857142857143', '0.6294642857142857', '0.6383928571428571', '0.5178571428571429', '0.5133928571428571'] |
| 5 | (14,13,12,11,10)-(5,4,3,2,1) | ['0.5401785714285714', '0.5625', '0.5357142857142857', '0.5401785714285714', '0.53125', '0.5357142857142857', '0.5357142857142857', '0.5535714285714286', '0.5357142857142857',0.5223214285714286] |

设计神经网络模型，分别对比隐含层为2、3、4、5时，不同神经节点数对应的准确率；

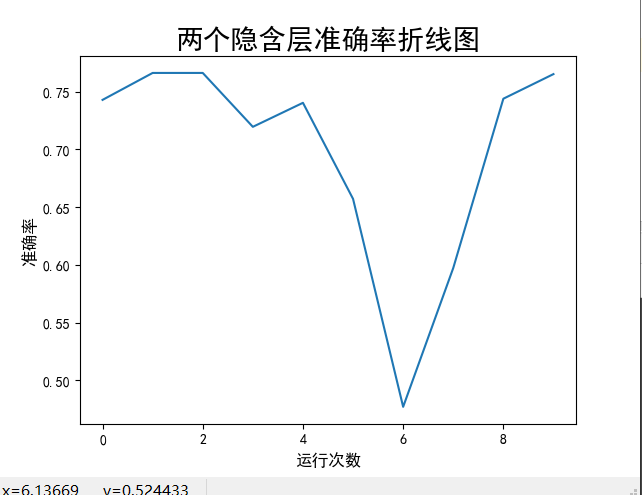
不同隐含层准确率折线图（2.2）如图所示；由图可知，当隐含层层数为2时，准确率最高为'0.825，此时隐含层神经节点数分别为6、5，即layers=[6, 6, 5, 3]；当隐含层层数为5时，模型准确率的波动较小，取值范围在[0.53125,0.5625]之间，可以认为模型比较稳定。



图（2.2）

2.3模型分析

设置layers=[6,6,5,3]，多次执行程序，得到准确率曲线，如图（2.3）所示；



图（2.3）

从图（2.3）可知，多次运行后模型准确率为['0.7430167597765364', '0.766378872524124', '0.766378872524124', '0.7196546470289487', '0.7404773996952768', '0.6571863890299644', '0.47689182326053836', '0.5972574911122397', '0.7440325038090401', '0.7653631284916201']除了有两个值较低以外，其余所有值均保持在0.7左右水平，故可以近似的认为适合处理该分类问题的神经网络模型为layers=[6,6,5,3]。

2.4同一模型针对不同数据集的准确率比较

|  |  |  |  |
| --- | --- | --- | --- |
| 隐含层层数 | 神经节点数 | 数据1准确率 | 数据2准确率 |
| 2 | （11,10)-(2,1) | ['0.7321428571428571', '0.5133928571428571', '0.5178571428571429', '0.5178571428571429', '0.5178571428571429', '0.8258928571428571', '0.4017857142857143', '0.6160714285714286', '0.4017857142857143', '0.5803571428571429'] | ['0.7623762376237624', '0.7458745874587459', '0.7458745874587459', '0.735973597359736', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459'] |
| 3 | (12,11,10)-(3,2,1) | ['0.7366071428571429', '0.53125', '0.5178571428571429', '0.5535714285714286', '0.5178571428571429', '0.7276785714285714', '0.5133928571428571', '0.4017857142857143', '0.5223214285714286', '0.5267857142857143'] | ['0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.759075907590759', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459'] |
| 4 | (13,12,11,10)-(4,3,2,1) | ['0.5178571428571429', '0.53125', '0.53125', '0.5267857142857143', '0.4017857142857143', '0.5267857142857143', '0.6294642857142857', '0.6383928571428571', '0.5178571428571429', '0.5133928571428571'] | ['0.7458745874587459', '0.7458745874587459', '0.7722772277227723', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7392739273927392', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459'] |
| 5 | (14,13,12,11,10)-(5,4,3,2,1) | ['0.5401785714285714', '0.5625', '0.5357142857142857', '0.5401785714285714', '0.53125', '0.5357142857142857', '0.5357142857142857', '0.5535714285714286', '0.5357142857142857',0.5223214285714286] | ['0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7458745874587459', '0.7425742574257426'] |

对比同一模型在处理不同分类问题时的准确率，可以发现在处理数据2这一分类问题上，所有模型的准确率均0.74左右，且波动范围极小。分析发现，出现以上情况的原因可能有：（1）数据2的特征数为20，远远超过数据1的特征数；（2）MLPC算法的隐含层使用sigmoid作为激活函数，而sigmoid函数通常用于二分类问题，数据1为一个三分类问题，数据2为一个二分类问题。