# 机器学习算法实现】logistic回归\_\_基于Python和Numpy函数库

2016/05/10 · [实践项目](http://python.jobbole.com/category/project/) · [1 评论](http://python.jobbole.com/85041/" \l "article-comment)· [回归](http://python.jobbole.com/tag/%e5%9b%9e%e5%bd%92/), [机器学习](http://python.jobbole.com/tag/machinelearning/), [算法](http://python.jobbole.com/tag/%e7%ae%97%e6%b3%95/)

分享到：**2**

原文出处： [wphh](http://blog.csdn.net/u012162613/article/details/41844495" \t "http://python.jobbole.com/85041/_blank)

【机器学习算法实现】系列文章将记录个人阅读机器学习论文、书籍过程中所碰到的算法，每篇文章描述一个具体的算法、算法的编程实现、算法的具体应用实例。争取每个算法都用多种语言编程实现。所有代码共享至github：[https://github.com/wepe/MachineLearning-Demo](https://github.com/wepe/MachineLearning-Demo" \t "http://python.jobbole.com/85041/_blank)     欢迎交流指正！

## **1、算法简介**

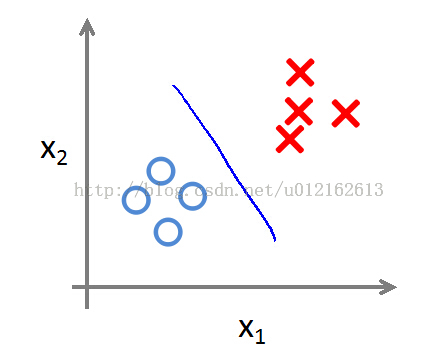
本文的重点放在算法的工程实现上，关于算法的原理不具体展开，logistic回归算法很简单，可以看看Andrew Ng的视频：[https://class.coursera.org/ml-007](https://class.coursera.org/ml-007" \t "http://python.jobbole.com/85041/_blank)，也可以看看一些写得比较好的博文：[洞庭之子的博文](http://blog.csdn.net/dongtingzhizi/article/details/15962797" \t "http://python.jobbole.com/85041/_blank)。下面我只列出一些个人认为重要的点。

****回归的概念****：假设有一些数据点，我们用一条直线对这些点进行拟合，这个拟合过程就称作回归。

logistic回归算法之所以称作“logistic”，是因为它运用了****logistic函数****，即[sigmoid函数](http://science.scileaf.com/library/214" \t "http://python.jobbole.com/85041/_blank)。

logistic回归算法一般用于****二分类问题****（当然也可以多类别，后面会讲）。

****logistic回归的算法思想：****

********

用上面的图来分析，每个O或X代表一个特征向量，这里是二维的，可以写成x=(x1,x2)。

用logistic回归进行分类的****主要思想****就是根据现有数据集，对分类边界建立回归公式，拿上面这个图来说，就是根据这些OOXX，找出那条直线的公式：Θ0\*x0+Θ1\*x1+Θ2\*x2=Θ0+Θ1\*x1+Θ2\*x2=0  （x0=1）。

因为上图是二维的，所以参数Θ=(Θ0，Θ1, Θ2)，分类边界就由这个(Θ0，Θ1, Θ2)确定，对于更高维的情况也是一样的，所以无论二维三维更高维，分类边界可以统一表示成****f(x)=ΘT\*x****  （ΘT表示Θ的转置）。

对于分类边界上的点，代入分类边界函数就得到f(x)=0,同样地，对于分类边界之上的点，代入得到f(x)>0，对于分类边界之下的点，代入得到f(x)f(x)大于0或者小于0来分类了。

logistic回归的最后一步就是将f(x)作为输入，代入Sigmoid函数，当f(x)>0时，sigmoid函数的输出就大0.5，且随着f(x)趋于正无穷，sigmoid函数的输出趋于1。当f(x)0时，sigmoid函数的输出就小于0.5，且随着f(x)趋于负无穷，sigmoid函数的输出趋于0。

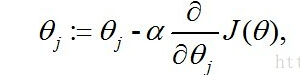
所以我们要寻找出****最佳参数Θ****，使得对于1类别的点x，f(x)趋于正无穷，对于0类别的点x，f(x)趋于负无穷（实际编程中不可能正/负无穷，只要足够大/小即可）。

总结一下思绪，****logistic回归的任务就是要找到最佳的拟合参数****。下图的g(z)即sigmoid函数，跟我上面讲的一样，将f(x)=****ΘT\*x****作为g(z)的输入。

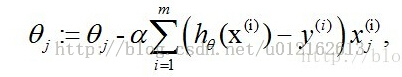


以上就是logistic回归的思想，重点在于怎么根据训练数据求得最佳拟合参数Θ？这可以用****最优化算法****来求解，比如常用的****梯度上升算法****，关于梯度上升算法这里也不展开，同样可以参考上面推荐的博文。

所谓的梯度，就是函数变化最快的方向，我们一开始先将参数Θ设为全1，然后在算法迭代的每一步里计算梯度，沿着梯度的方向移动，以此来改变参数Θ，直到Θ的拟合效果达到要求值或者迭代步数达到设定值。Θ的更新公式：



alpha是步长，一系列推导后：



这个公式也是下面写代码所用到的。

后话：理解logistic回归之后可以发现，其实它的****本质是线性回归****，得到****ΘT\*x****的过程跟线性回归是一样的，只不过后面又将****ΘT\*x****作为logistic函数的输入，然后再判断类别。

## **2、工程实例**

 logistic回归一般用于二分类问题，比如判断一封邮件是否为垃圾邮件，判断照片中的人是男是女，预测一场比赛输还是赢……当然也可以用于多分类问题，比如k类别，就进行k次logistic回归。

我的前一篇文章：[kNN算法\_\_手写识别](http://blog.csdn.net/u012162613/article/details/41768407" \l "t2" \t "http://python.jobbole.com/85041/_blank) 讲到用kNN算法识别数字0～9，这是个十类别问题，如果要用logistic回归，

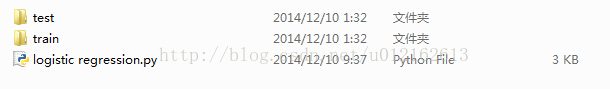
得做10次logistic回归，第一次将0作为一个类别，1～9作为另外一个类别，这样就可以识别出0或非0。同样地可以将1作为一个类别，0、2～9作为一个类别，这样就可以识别出1或非1……..

本文的实例同样是识别数字，但为了简化，我只选出0和1的样本，这是个二分类问题。下面开始介绍实现过程：

### **（1）工程文件说明**

在我的工程文件目录下，有训练样本集train和测试样本集test，源代码文件命名为logistic regression.py

训练样本集train和测试样本集test里面只有0和1样本：



 logistic regression.py实现的功能：从train里面读取训练数据，然后用梯度上升算法训练出参数Θ，接着用参数Θ来预测test里面的测试样本，同时计算错误率。

### **（2）源代码解释**

****loadData(direction)函数****

实现的功能就是从文件夹里面读取所有训练样本，每个样本（比如0\_175.txt）里有32\*32的数据，程序将32\*32的数据整理成1\*1024的向量，这样从每个txt文件可以得到一个1\*1024的特征向量X，而其类别可以从文件名“0\_175.txt”里截取0出来。因此，从train文件夹我们可以获得一个训练矩阵m\*1024和一个类别向量m\*1，m是样本个数。



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | def loadData(direction):      trainfileList=listdir(direction)      m=len(trainfileList)      dataArray= zeros((m,1024))      labelArray= zeros((m,1))  **for** i **in** range(m):          returnArray=zeros((1,1024))  #每个txt文件形成的特征向量          filename=trainfileList[i]          fr=open('%s/%s' %(direction,filename))  **for** j **in** range(32):              lineStr=fr.readline()  **for** k **in** range(32):                  returnArray[0,32\*j+k]=**int**(lineStr[k])          dataArray[i,:]=returnArray   #存储特征向量            filename0=filename.split('.')[0]          label=filename0.split('\_')[0]          labelArray[i]=**int**(label)     #存储类别  **return** dataArray,labelArray |

代码里面用到python os模块里的listdir()，用于从文件夹里读取所有文件，返回的是列表。

python里的open()函数用于打开文件，之后用readline()一行行读取

* ****sigmoid(inX)函数****



|  |  |
| --- | --- |
| 1  2 | def sigmoid(inX):  **return** 1.0/(1+exp(-inX)) |

* ****gradAscent(dataArray,labelArray,alpha,maxCycles)函数****

用梯度下降法计算得到回归系数，alpha是步长，maxCycles是迭代步数。



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | def gradAscent(dataArray,labelArray,alpha,maxCycles):      dataMat=mat(dataArray)    #size:m\*n      labelMat=mat(labelArray)      #size:m\*1      m,n=shape(dataMat)      weigh=ones((n,1))  **for** i **in** range(maxCycles):          h=sigmoid(dataMat\*weigh)          error=labelMat-h    #size:m\*1          weigh=weigh+alpha\*dataMat.transpose()\*error  **return** weigh |

用到numpy里面的mat，矩阵类型。shape()用于获取矩阵的大小。

这个函数返回参数向量Θ，即权重weigh

* ****classfy(testdir,weigh)函数****

分类函数，根据参数weigh对测试样本进行预测，同时计算错误率



|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | def classfy(testdir,weigh):      dataArray,labelArray=loadData(testdir)      dataMat=mat(dataArray)      labelMat=mat(labelArray)      h=sigmoid(dataMat\*weigh)  #size:m\*1      m=len(h)      error=0.0  **for** i **in** range(m):  **if** **int**(h[i])>0.5:              print **int**(labelMat[i]),'is classfied as: 1'  **if** **int**(labelMat[i])!=1:                  error+=1                  print 'error'  **else**:              print **int**(labelMat[i]),'is classfied as: 0'  **if** **int**(labelMat[i])!=0:                  error+=1                  print 'error'      print 'error rate is:','%.4f' %(error/m) |

* ****digitRecognition(trainDir,testDir,alpha=0.07,maxCycles=10)函数****

整合上面的所有函数，调用这个函数进行数字识别，alpha和maxCycles有默认形参，这个可以根据实际情况更改。



|  |  |
| --- | --- |
| 1  2  3  4 | def digitRecognition(trainDir,testDir,alpha=0.07,maxCycles=10):      data,label=loadData(trainDir)      weigh=gradAscent(data,label,alpha,maxCycles)      classfy(testDir,weigh) |

用loadData函数从train里面读取训练数据，接着根据这些数据，用gradAscent函数得出参数weigh，最后就可以用拟合参数weigh来分类了。

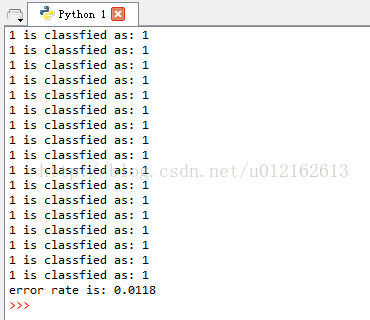
## **3、试验结果**

工程文件可以到这里下载：[github地址](https://github.com/wepe/MachineLearning-Demo/tree/master/logistic regression" \t "http://python.jobbole.com/85041/_blank)

运行logistic regression.py，采用默认形参：alpha=0.07,maxCycles=10，看下效果，错误率0.0118



|  |  |
| --- | --- |
| 1 | >>> digitRecognition('train','test') |



改变形参，alpah=0.01，maxCycles=50，看下效果，错误率0.0471



|  |  |
| --- | --- |
| 1 | >>> digitRecognition('train','test',0.01,50) |

