**回归**

# 1 概述

　Logistic回归的目的是寻找一个非线性函数Sigmoid的最佳拟合参数，求解过程可以由最优化算法来完成。在最优化算法中，最常用的就是梯度上升算法，而梯度上升算法又可以简化为随机梯度上升算法。

随机梯度上升算法与梯度上升算法的效果相当，但是占用更少的计算资源。此外，随机梯度上升是一个在线算法，它可以在新数据到来时就完成参数更新，不需要重新读取整个数据集来进行批处理运算。

# 2 步骤

## 2.1 Logistic回归的一般过程

（1）收集数据：采用任意方法收集数据

（2）准备数据：由于需要进行距离计算，因此要求数据类型为数值型。

（3）分析数据：采用任意方法对数据进行分析。

（4）训练算法：大部分时间将用于训练，训练的目的是为了找到最佳的分类回归系数。

（5）使用算法：首先输入一些数据，并将其转换成对应的结构化数值；接着，训练好的回归系数就可以对这些数值进行简单的回归计算，判定他们属于哪个类别；然后就可以在输出的类别上做一些其他分析工作。

## **2.2 基于Logistic回归和Sigmoid函数的分类**

一般函数应该是：能接受所有的输入，然后预测出类型。例如，在两个类的情况下，上述函数输出0或1。该函数称为海维赛德阶跃函数（Heaviside step function），或者直接称为单位阶跃函数。然而，海维赛德阶跃函数的问题在于：该函数在跳跃点上从0瞬间跳跃到1，这个瞬间跳跃过程有时很难处理。幸好，另一个函数也有类似的性质（可以输出0或者1），且数学上更易处理，这就是Sigmoid函数。Sigmoid函数具体的计算公式如下：



自变量取值为任意实数，值域[0, 1]

图1给出了Sigmoid函数在不同坐标尺度下的两条曲线图。当x为0时，Sigmoid函数值为0.5。随着x的增大，对应的Sigmoid值将逼近于1；而随着x的减少，Sigmoid值将逼近于0.如果横坐标刻度足够大，Sigmoid函数看起来很像一个阶跃函数。

解释Sigmoid函数：将任意的输入映射到了 [0, 1]区间，我们在线性回归中可以得到一个预测值，再将该值映射到 Sigmoid函数中这样就完成了由值到概率的转换，也就是分类任务。

因此，为了实现Logistic回归分类器，我们可以在每个特征上都乘以一个回归系数，然后把所有的结果值相加，将这个总和带入Sigmoid函数中，进而得到一个范围在0~1之间的数值。任何大于0.5的数据被分入1类，小于0.5即被归入0类，所以，Logistic回归也可以被看成是一种概率估计。

确定了分类器的函数形式之后，现在的问题变成了：最佳回归系数是多少？如何确定其大小。

如下图1，两种坐标尺度下的Sigmoid函数图。上图的横坐标为-5到5，这时的曲线变化比较为平滑；下图横坐标的尺度足够大，可以看到，在x=0点处Sigmoid函数看起来很像阶跃函数。

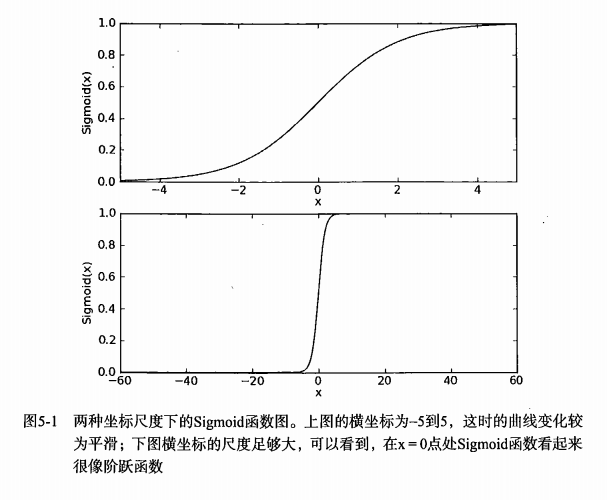


图1

## 2.3基于最优化方法的最佳回归系数确定

Sigmoid函数的输入记为z，由下面公式得到：



如果采用向量的写法，上述公式可以写成 ，它表示将这两个数值向量对应元素相乘，然后全部加起来即得到z值。



其中的向量x是分类器的输入数据，向量w也就是需要求出的最佳参数（系数），从而使得分类器尽可能的准确。需要用到Logistic回归模型的公式：



假设 W>0，Y与X各维度叠加的图形关系，如下图2所示（x为了方便取1维）：

梯度上升的最优化方法，求得数据集的最佳参数，接下来，展示如何绘制梯度上升法产生的决策边界图，该图将梯度上升法的分类效果可视化的呈现出来，最后将学习随机梯度上升算法，以及如何对其进行修改以获得很好地结果。梯度上升算法用来求函数的最大值，梯度下降算法是用来求函数的最小值。

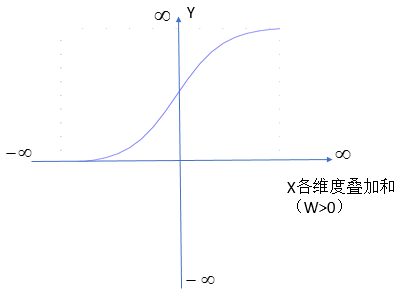


图2

## 2.4 梯度上升法

梯度上升法基于的思想是：要找到某函数的最大值，最好的方法是沿着该函数的梯度方向探寻，如果梯度记为，则函数的梯度由下面式子表示：



这个梯度意味着要沿着x的方向移动，沿着y方向移动，其中函数必须要在待计算的点上有定义并且可微，一个具体的函数例子如下图3，梯度上升算法到达每个点后都会重新计算移动的方向。从P0开始，计算完该点的梯度，函数就根据梯度移动到下一点P1。在P1点，梯度再次被重新计算，并沿新的梯度方向移到到P2。如此循环迭代，知道满足停止条件。迭代的过程中，梯度算子总是保证我们能选取到最佳的移动方向：

上图中的梯度上升算法沿梯度方向移动了一步，可以看出，梯度算子总是指向函数值增长最快的方向。这里所说的移动方向，而未提到移动量的大小。该量值称为步长，记为。用向量来表示的话，梯度算法的迭代公式如下：

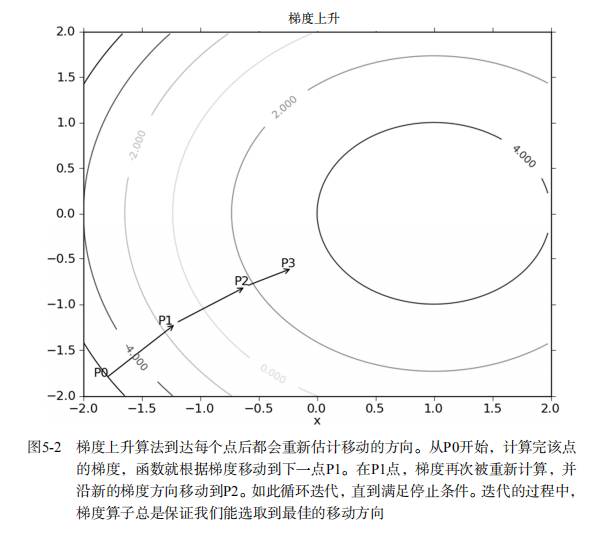


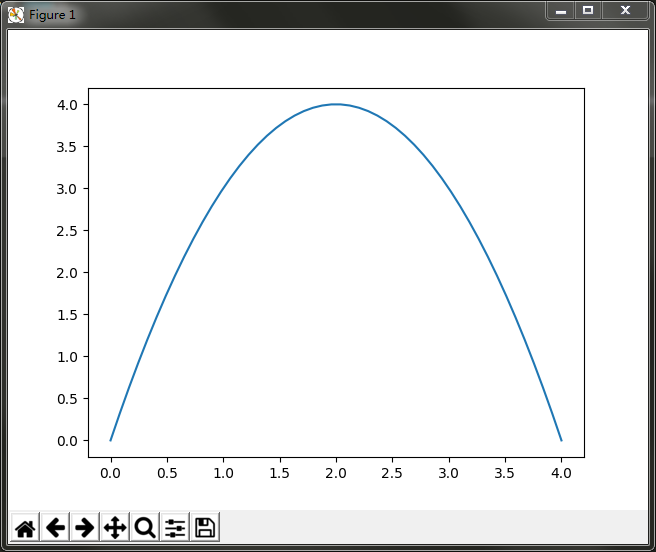
图3



该公式将一直被迭代执行，直至达到某个停止条件为止，比如迭代次数达到某个指定值或算法达到某个可以允许的误差范围。

# 3 梯度上升案例

函数求极值？这个函数开口向下，存在极大值，它的函数图像为：

求极值，先求函数的导数：



令导数为0，可求出x=2即取得函数f(x)的极大值。极大值等于f(2)=4。

但是真实环境中的函数不会像上面这么简单，就算求出了函数的导数，也很难精确计算出函数的极值。此时我们就可以用迭代的方法来做。就像爬坡一样，一点一点逼近极值。这种寻找最佳拟合参数的方法，就是最优化算法。爬坡这个动作用数学公式表达即为：

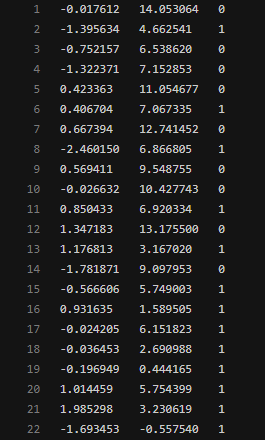


其中，α为步长，也就是学习速率，控制更新的幅度。效果如下图所示：

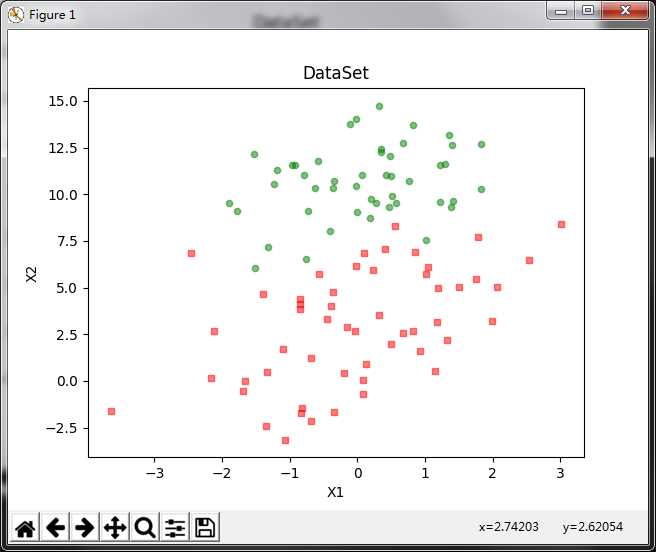
比如从(0,0)开始，迭代路径就是1->2->3->4->...->n，直到求出的x为函数极大值的近似值，停止迭代。

# IMG_2564 线性回归案例

下图为数据集：

[](https://cuijiahua.com/wp-content/uploads/2017/11/ml_6_27.png)

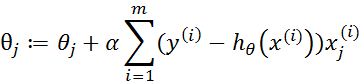
这个数据有两维特征，因此可以将数据在一个二维平面上展示出来。将第一列数据(X1)看作x轴上的值，第二列数据(X2)看作y轴上的值。而最后一列数据即为分类标签。根据标签的不同，对这些点进行分类。

[](https://cuijiahua.com/wp-content/uploads/2017/11/ml_6_28.png)运行结果如下：

从上图可以看出数据的分布情况。假设Sigmoid函数的输入记为z，那么z=w0x0 + w1x1 + w2x2，即可将数据分割开。其中，x0为全是1的向量，x1为数据集的第一列数据，x2为数据集的第二列数据。另z=0，则0=w0 + w1x1 + w2x2。横坐标为x1，纵坐标为x2。这个方程未知的参数为w0，w1，w2，就是需要求的回归系数(最优参数)。

（1）训练算法

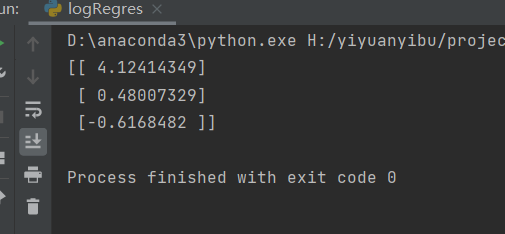
梯度上升迭代公式：



将上述公式矢量化：

[IMG_258](https://cuijiahua.com/wp-content/uploads/2017/11/ml_6_30.png)（2）根据矢量化的公式，编写代码：

运行结果如图所示：

可以看出，已经求解出回归系数[w0,w1,w2]，0=w0x0 + w1x1 + w2x2，X0数据处理时全部补成1，x1相当于x，x2相当于y求得的直线函数应为：

y = (-weights[0]-weights[1] \* x) / weights[2]

通过求解出的参数，确定不同类别数据之间的分隔线，画出决策边界。

（3）绘制决策边界

已经解出了一组回归系数，它确定了不同类别数据之间的分隔线。现在开始绘制这个分隔线：

运行结果如下：

