# 《机器学习与深度学习》课程

# 实 验 报 告



**姓 名： 金家耀**

**专 业：**  人工智能

**学 号： 1193210320**

**江南大学人工智能与计算机学院**

# 逻辑回归

**1实验目的**

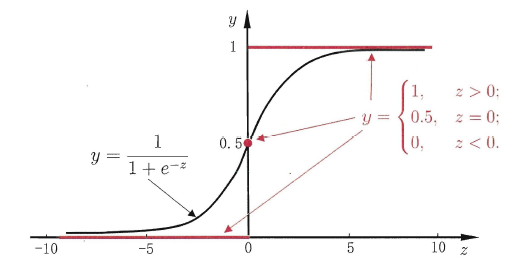
1）理解逻辑回归（对数几率回归）算法原理，掌握逻辑回归算法框架和实现过程；  
2）理解逻辑回归的sigmoid函数；  
3）理解逻辑回归的损失函数；  
4）针对特定应用场景及数据，能应用逻辑回归算法解决实际分类问题。

**2实验原理**

现在我们利用逻辑回归解决二分类问题。我们希望输出概率P大于某个值时，则判别为类别A，输出概率P小于某个值的时候判为类别B。

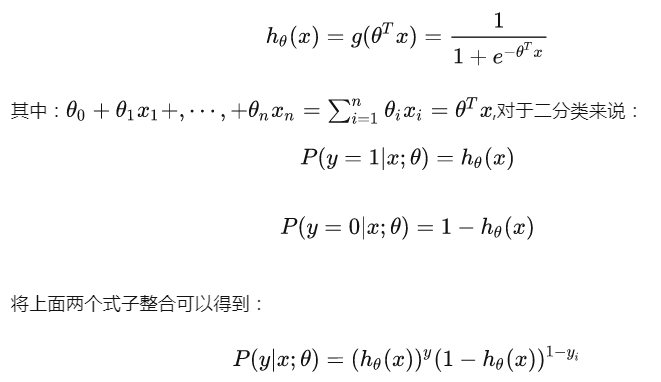
Sigmod函数特性：对于任意实数x∈(−∞,+∞)，对应的y的取值范围为：(0,1)，即将输入的任意实数映射到(0,1)之间，实现了数值到概率的转换。具体函数为：





如上图所示，可以看出当x<0的时候，0<y<0.5，当x>0的时候，0.5<y<1。

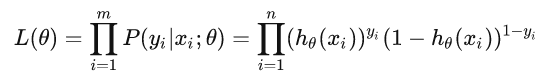
逻辑回归的预测函数：



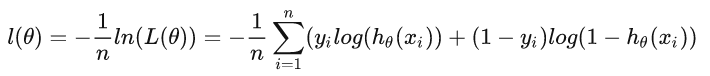


逻辑回归参数求解：

上面得到了预测函数，那么我们如何求解参数θ呢？这里采用了最大似然估计。其中极大似然估计是数理统计中参数估计的一种重要方法。其思想就是一个事件发生了，那么发生这个事件的概率就是最大的。这里我们希望寻找一组参数θ使得每组数据发生得概率最大(最好每个数据x都和其分类一一对应)。即：

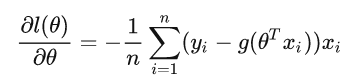


两边同取对数，因为是求式子的最大值，可以转换成式子乘以负1,之后求最小值。同时对于n个数据，累加后值会很大，之后如果用梯度下降容易导致梯度爆炸。所以可以除以样本总数n。





类似于两个贝努力分布得交叉熵（用来评估两个分布的接近程度）。 对θ求偏导得到梯度，公式如下：



通过梯度下降法即可求得参数θ

**3实验内容和要求**

数据集ex2data1.txt是入学申请人的历史数据集，每条记录都有申请人的两次考试成绩和录取决定标记（0代表不录取，1代表录取）。

实验内容：建立一个逻辑回归模型来预测一个学生是否会被大学录取。假设您是大学部门的管理员，您想根据申请人的两次考试成绩来确定他们的入学机会。您有来自以前申请人的历史数据（ex2data1.txt），可以用作逻辑回归的训练集。实验的任务是编写python代码完成逻辑回归算法程序，建立一个分类模型，根据这两门考试的分数估计申请人被录取的概率。

算法步骤与要求：

(1)读取数据；

(2)绘制数据观察数据分布情况；

(3)编写sigmoid函数代码；

(4)编写逻辑回归代价函数代码；

(5)编写梯度函数代码；

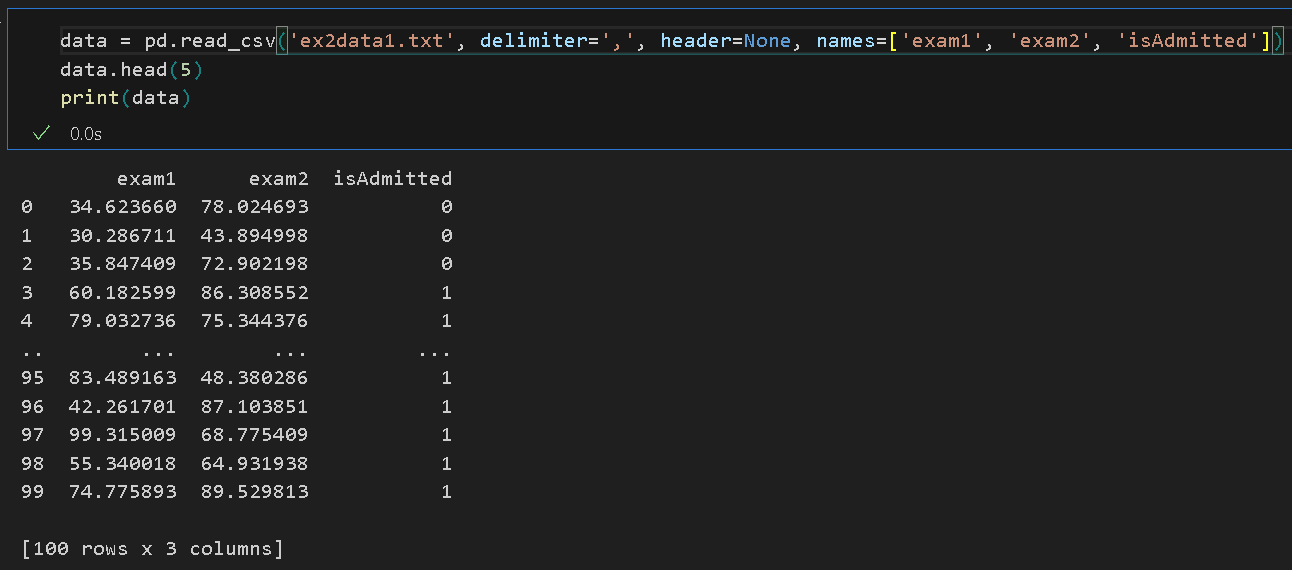
(6)编写寻找最优化参数代码（可使用scipy.opt.fmin\_tnc()函数）；

(7)编写模型评估（预测）代码，输出预测准确率；

(8)寻找决策边界，画出决策边界直线图。

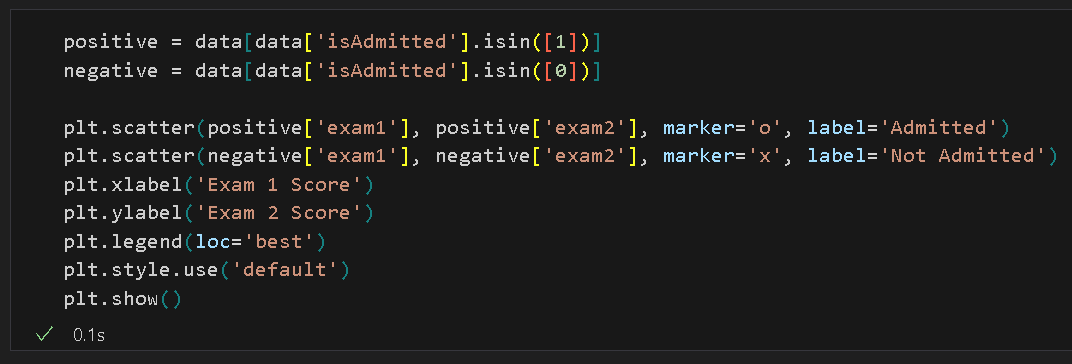
**4实验代码和结果**

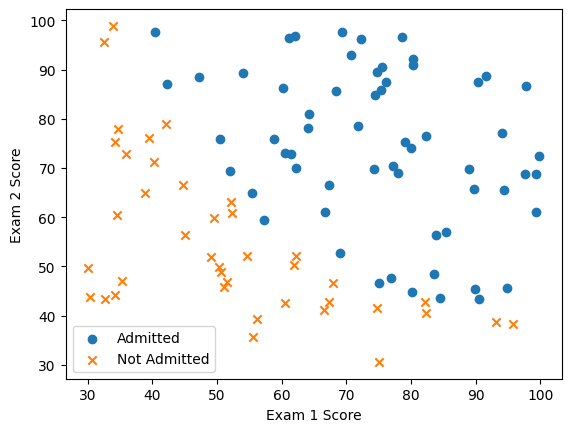
1. **读取数据；**



利用pandas库从txt文件中读入数据，exam1和exam2字段分别表示两次考试的成绩，isAdmitted表示该学生是否通过被录取。

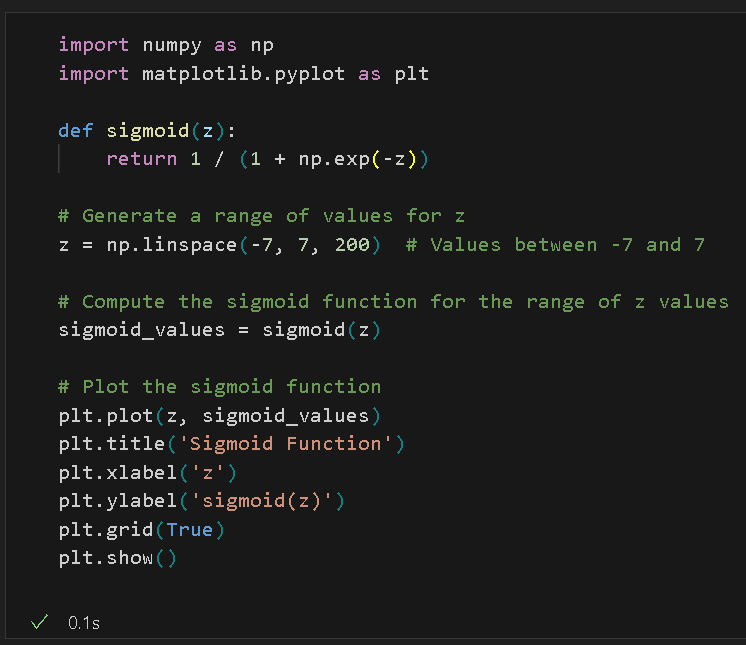
1. **绘制数据观察数据分布情况；**

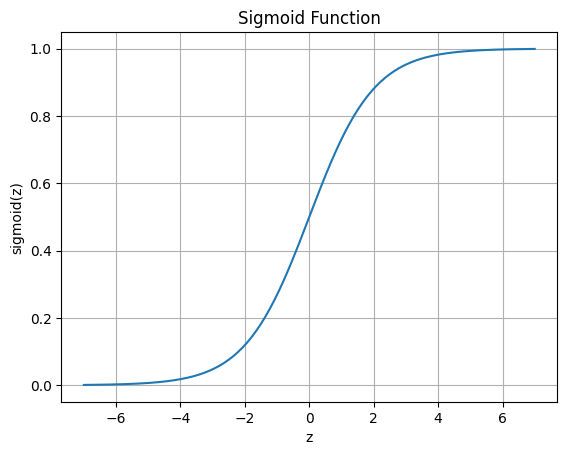




我使用了Matplotlib库中的plt函数来创建一个散点图，以可视化学生的两次考试成绩与录取情况之间的关系。在图中，横轴代表第一次考试成绩，纵轴代表第二次考试成绩，而学生是否被录取则通过不同的符号和颜色来表示。通过观察图表，我发现这些数据点大致可以被一条线分开，这满足了适用于简单逻辑回归的线性可分条件。这个图表有助于我直观地理解数据的分布和潜在的分类趋势。

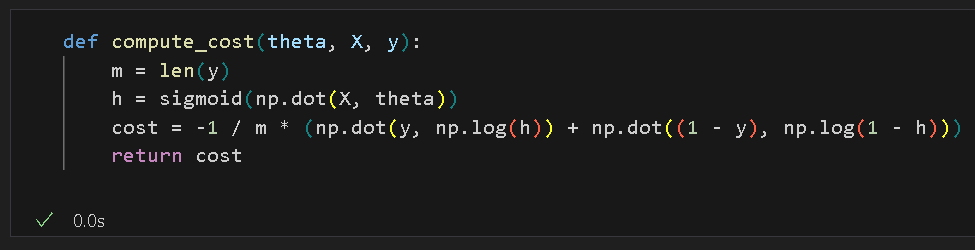
1. **编写sigmoid函数代码；**

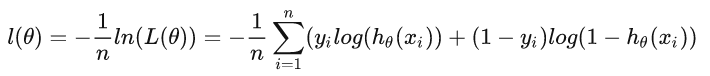




在实验中，我绘制了Sigmoid激活函数的图像。Sigmoid函数通常用于二元分类问题，它具有S形曲线的形状，将输入值映射到0和1之间。图中的横轴代表输入值，纵轴代表Sigmoid函数的输出值。这个函数的特点是在输入接近0时，输出值接近0.5，因此通常被用作二元分类问题的阈值函数。这个图表有助于理解Sigmoid函数的形状和其在机器学习中的应用。

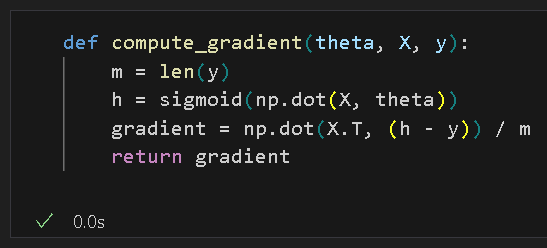
1. **编写逻辑回归代价函数代码；**

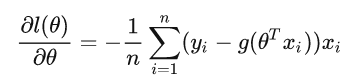




在本实验中，我使用了交叉熵损失函数，这是一种常用于分类问题的损失函数。其定义如下：给定真实类别标签和模型的预测概率分布，交叉熵损失衡量了模型预测与实际标签之间的差异。这个损失函数通常用于多类别分类问题，其中真实类别标签通常采用one-hot编码的向量表示，而模型的预测概率分布是一个包含各个类别概率的向量。

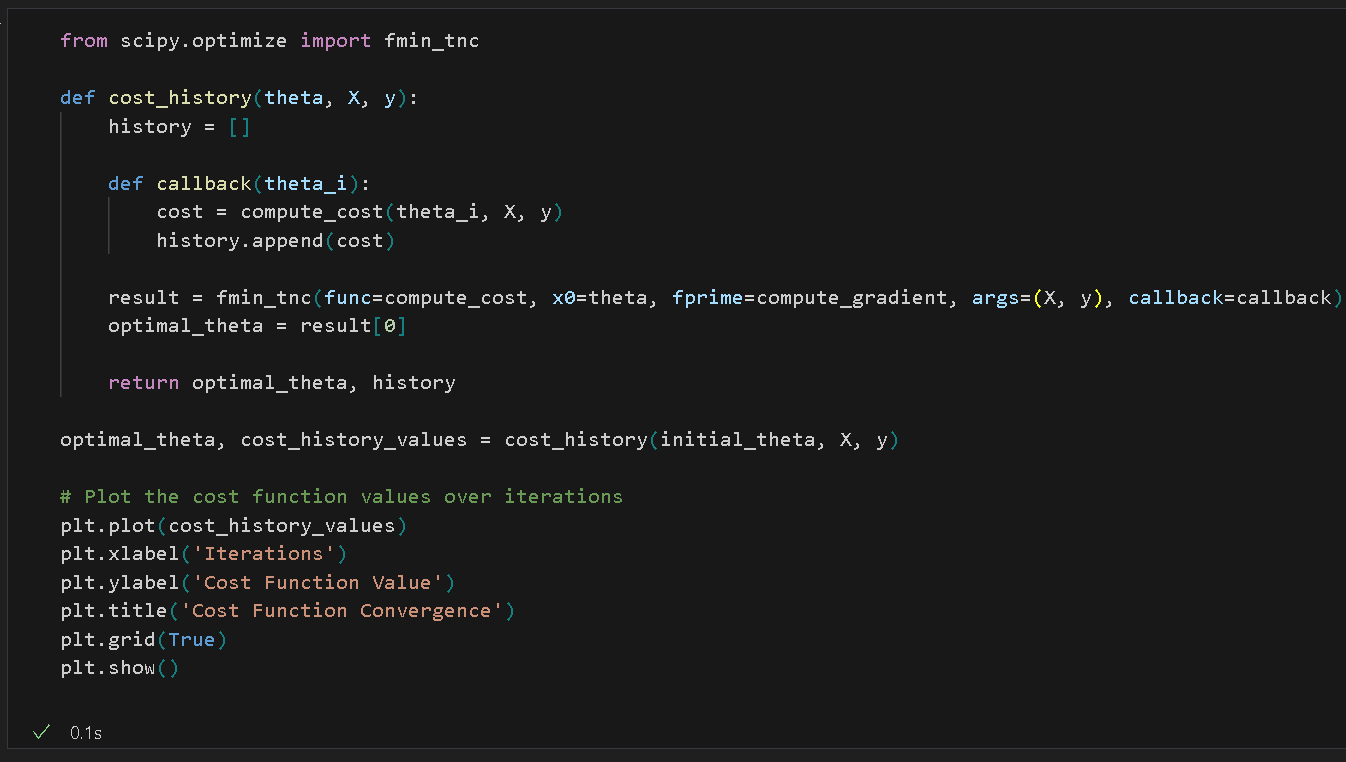
1. **编写梯度函数代码；**

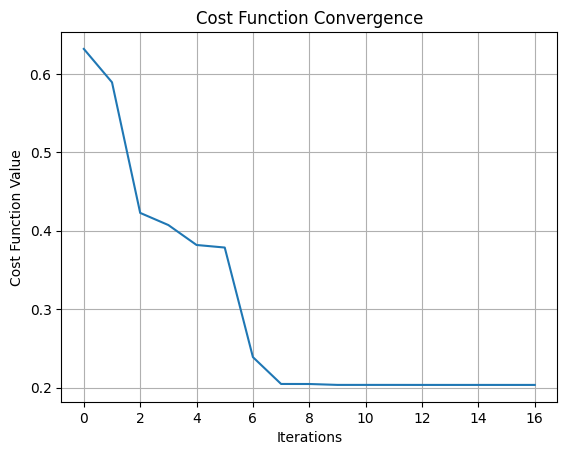




梯度下降是用于最小化代价函数的优化算法。通过计算代价函数的梯度，我们可以不断调整模型参数以降低代价函数的值。

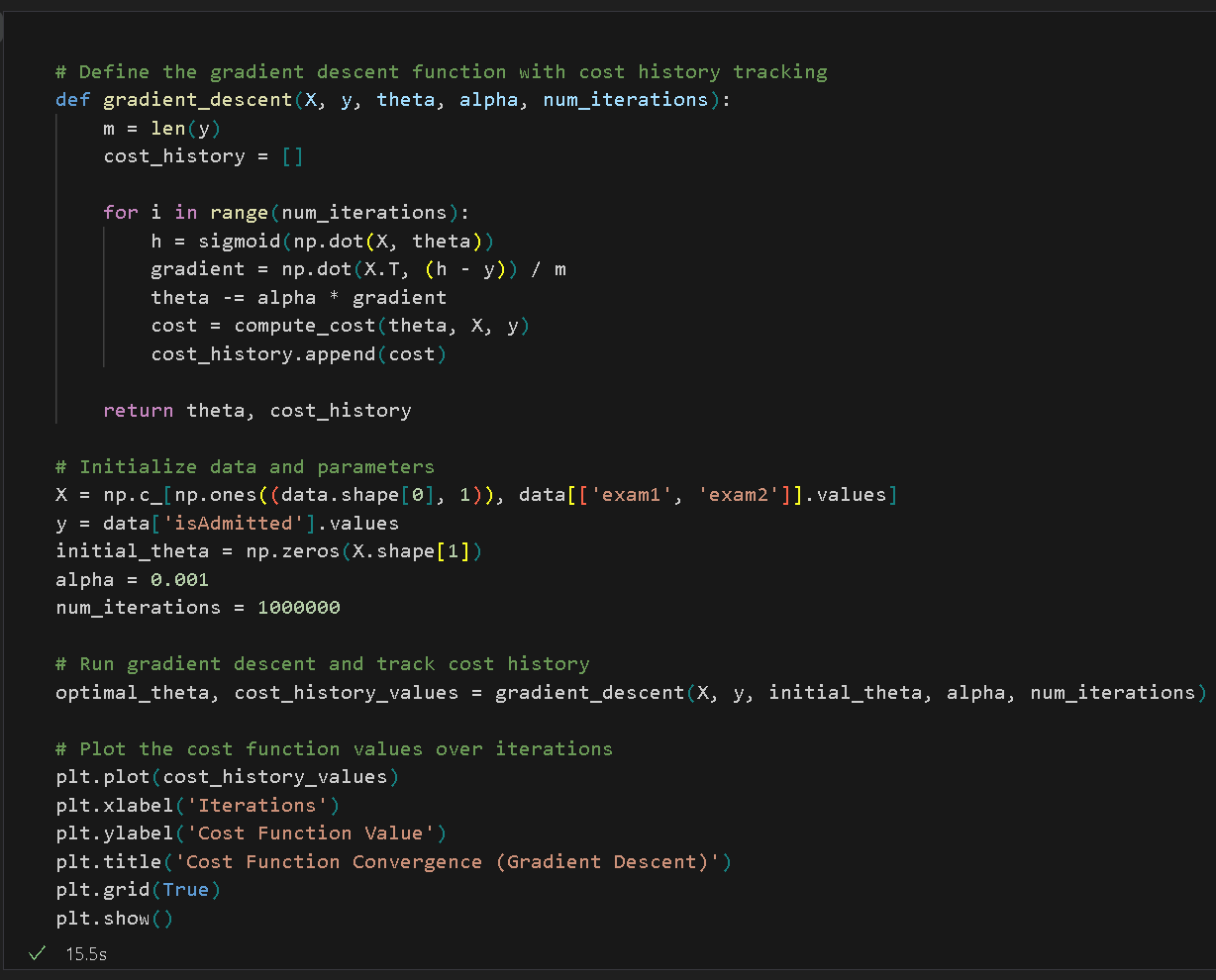
1. **编写寻找最优化参数代码（可使用scipy.opt.fmin\_tnc()函数）；**
2. **使用scipy.opt.fmin\_tnc()**

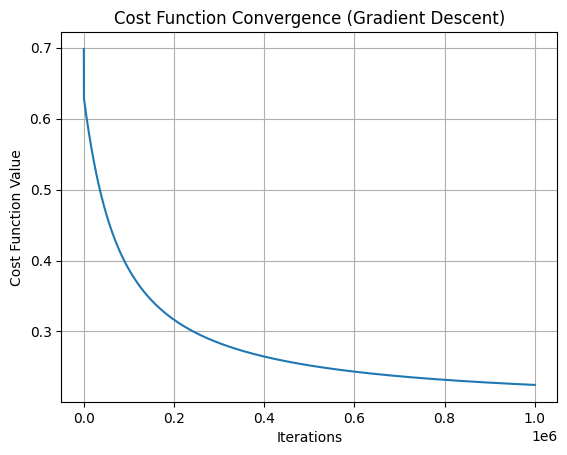




先使用了scipy.opt.fmin\_tnc()函数，每次迭代的梯度图如上图所示，最后损失值接近0.2。

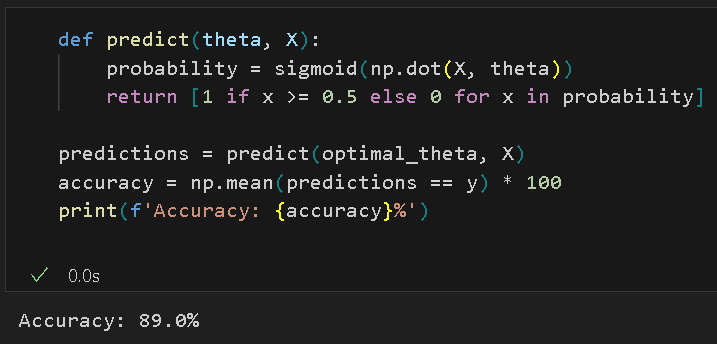
1. **自己编写迭代函数**





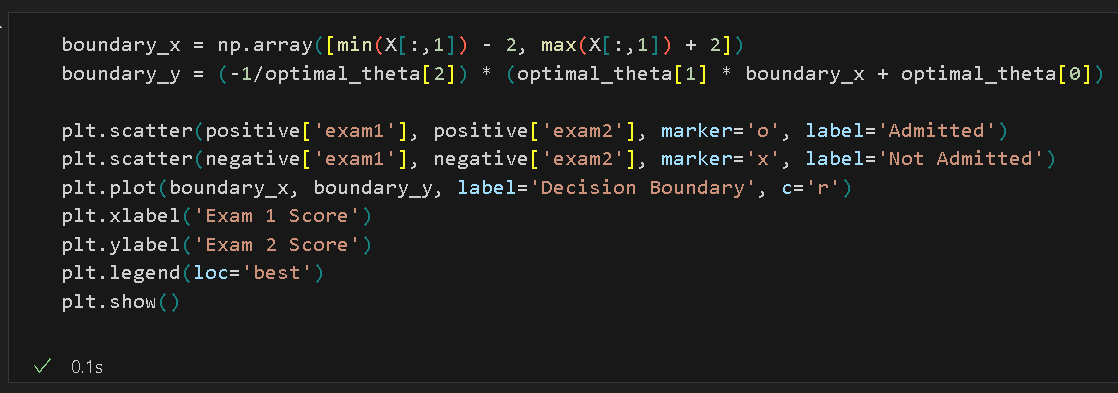
这里是我自己手动实现的一个梯度下降过程，我不断调整learning rate（这里指的是alpha）并将其指定为1e-3，最后迭代1e6次，花费时间15.5s，最后损失函数同样也接近0.2。

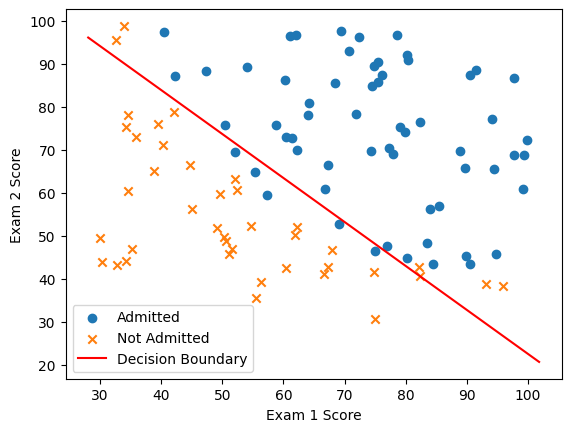
1. **编写模型评估（预测）代码，输出预测准确率；**



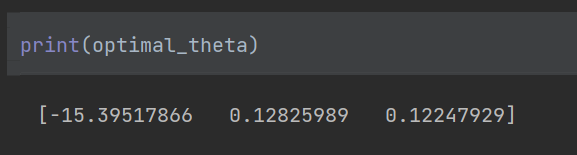
最后将0.5作为分界值，如果最后函数的结果大于等于0.5，那其为第1类，反之为第2类。使用scipy库和自己手动实现最后的准确率都为89.0%

1. **寻找决策边界，画出决策边界直线图。**

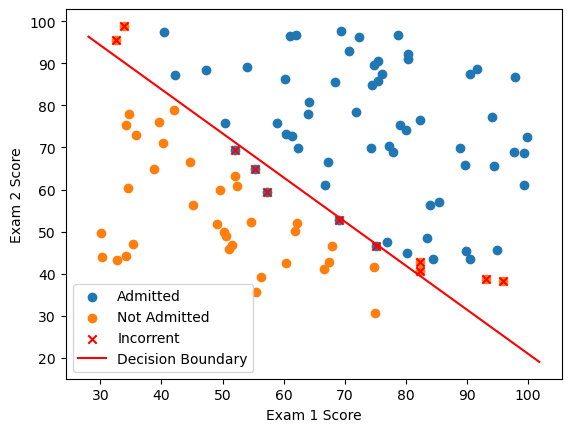




使用优化后的theta参数计算决策边界直线的斜率和截距并绘制图像。



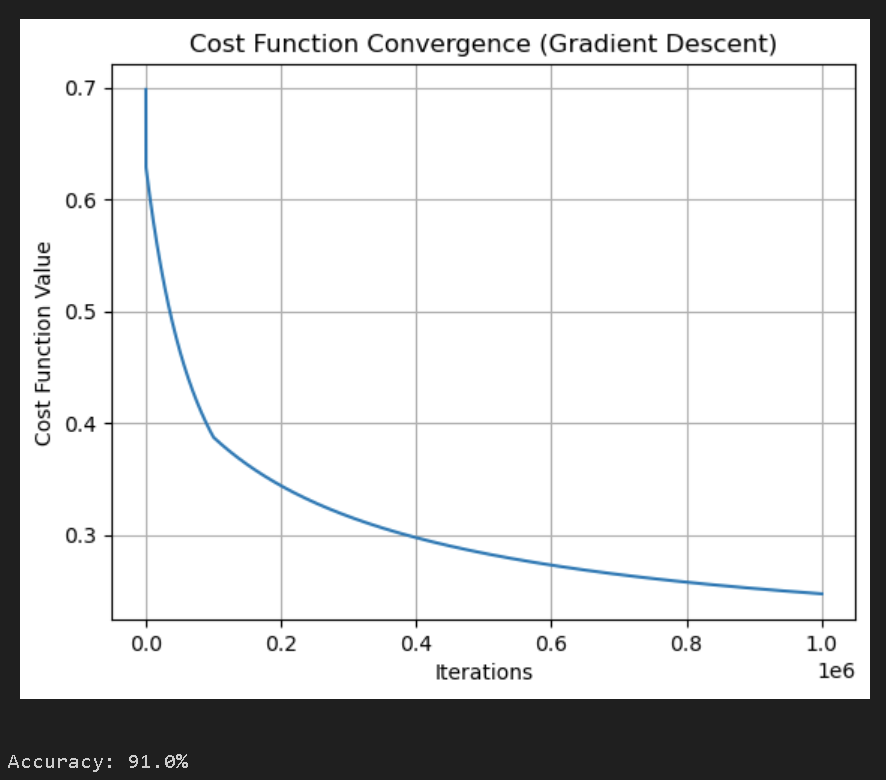
则优化后的optimal\_theta分别是-15.39517866、0.12825989、0.12247929，则判定函数为y = 0.12825989 \* x1 + 0.12247929 \* x2 - 15.39517866，当y大于0时，可以被录取，当y小于0时，不能被录取，从中亦可以看出两次成绩的重要性其实差不多。



如图，我在图中用不同颜色绘制出了预测结果与原始结果，用红色的‘x’代表预测错误的样本。预测的原因是数据集其实并不严格满足线性可分，错误的预测是无法避免的，但后续可以使用非线性的模型拟合方法来进行分类预测。

1. **自己实现简单的学习率调度器（\*）**





由于学习率不变或是太大的原因，模型可能会陷入局部最优解。在这里实验中，我实现了一个简单的学习率调度器，即当使用大学习率进行训练几代后，马上切换为比原先学习率小的学习率，使模型能够在解空间中找到更为好的解。

**虽然这一内容在实验报告中并不要求做，但如何选择优化器同样与模型一样重要，不同的优化器有不同的梯度下降策略。同时学习率也是这些优化器至关重要的点，在实验中我简单的做了一个学习率衰减操作，使得模型表现更佳，同时，这也加深了我对于学习率、优化器重要性的印象。**