# Planar data classification with one hidden layer

# 1、作业简介

本次作业将建立含有一个隐藏层的神经网络，将会看到这个模型和之前使用logistic回归实现的模型之间的巨大差异。

* 使用含有一个隐藏层的神经网络实现二分类
* 使用一个非线性的激活函数(比如tanh)
* 计算交叉熵损失（损失函数）
* 实现前向传播和反向传播

## 2、工具包

在这个过程中，我们将会用到如下库：

numpy：Python中常用的科学计算库

sklearn ：提供简单有效的数据挖掘和数据分析

matplotlib：用于数据可视化

testCases：用以测试相对应函数的测试例子

planar\_utils ：本文实践中需要用到其他一些函数

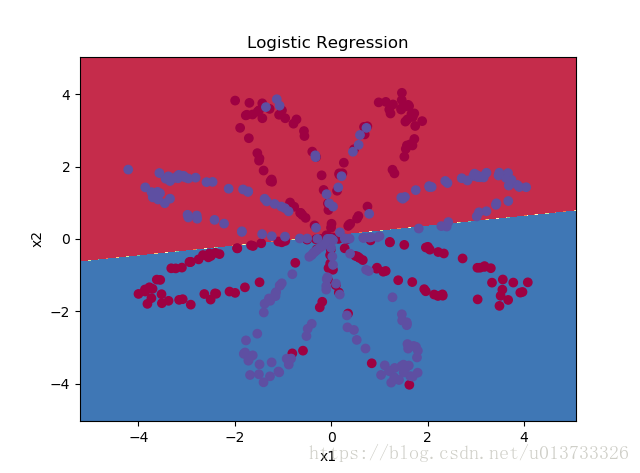
## 3、数据集

加载数据，使用matplotlib将数据可视化。数据集类似一朵花，有红色(label y = 0)和蓝色(label y = 1)的点构成，我们的目的就是去拟合这个数据。

**4、简单的Logistic回归**

在构建完整的神经网络之前，先让我们看看逻辑回归在这个问题上的表现如何，我们可以使用sklearn的内置函数实现。

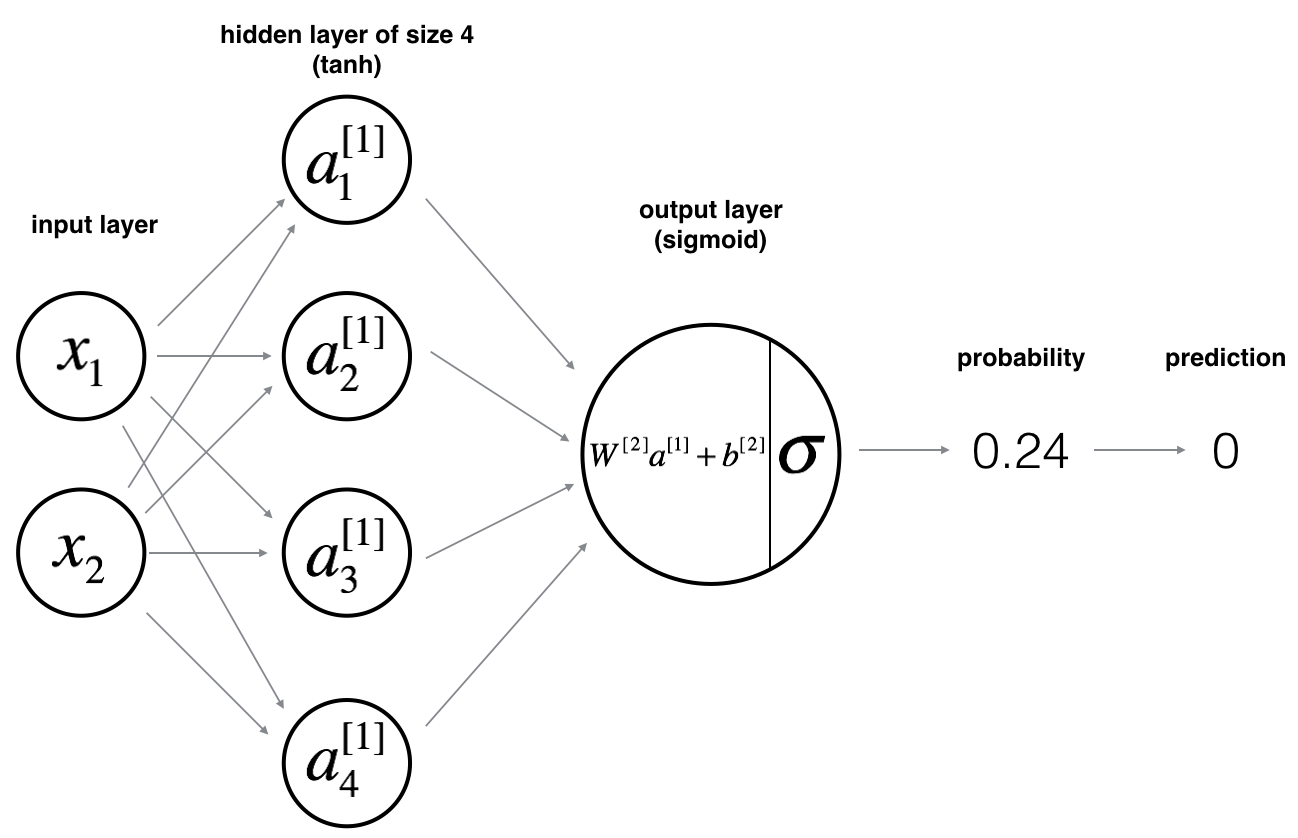
可以得到逻辑回归的准确性为47 %，准确性只有47%的原因是数据集不是线性可分的，所以逻辑回归表现不佳，现在我们正式开始构建神经网络。



**5、神经网络模型**

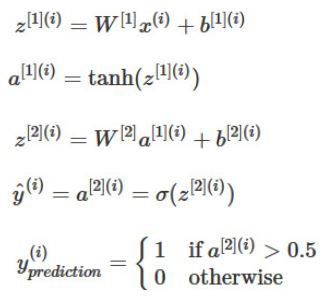
由上面可以看出logistic模型对于解决“flower dataset”效果并不好，这里我们创建一个隐藏层的神经网络，下图是我们使用的网络模型：

从上述的逻辑回归结果我们可以看出该模型对于花形数据表现不佳，为此我们采用单隐藏层的神经网络作为尝试，模型如下：

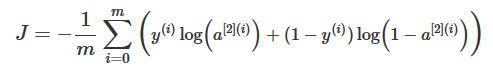


**数学表达式：**

对于输入:



对于所有样本，损失函数J的计算：



**构建神经网络的基本步骤：**

1.定义神经网络的结构(输入单元，隐藏单元等)

2.初始化模型参数

3.循环

1)执行前向传播

2)计算损失

3)执行反向传播

4)更新参数(梯度下降)

通常我们将1-3步骤创建为功能函数(独立的函数)，然后再将它们合并为一个函数，我们称为nn\_model()，创建了nn\_model()函数之后我们就可以进行预测。

**5.1定义神经网络结构**

定义三个变量：

n\_x：输入层单元数目

n\_h：隐藏层单元数目

n\_y：输出层单元数目

**5.2初始化模型参数**

随机初始化权重参数，偏置参数初始化为0。

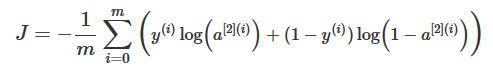
**5.3循环**

**5.3.1前向传播**

为前向传播定义一个函数，在隐藏层的激活函数是tanh，在输出层的激活函数是sigmoid。利用初始化的参数计算Z[1]，A[1]，Z[2] 和A[2]，同时注意保留值到cache，因为在后续的后向传播需要用到。

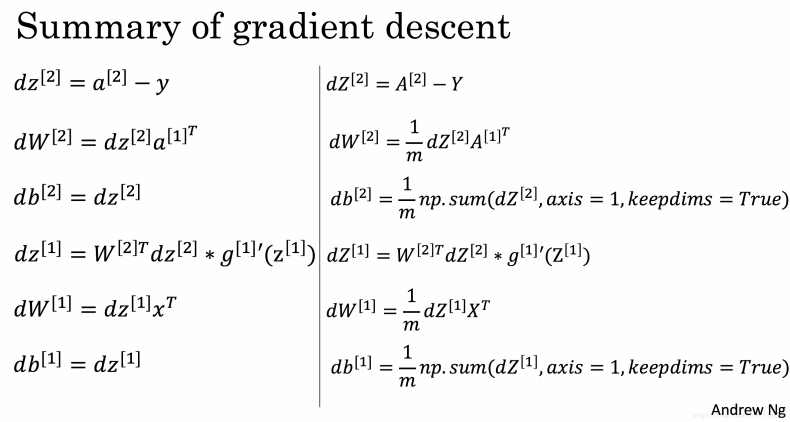
**5.3.2代价函数**

在输出的cache中，我们是记录了 （在代码中用A2表示），该结果包含了的所有值，即整个样本。为此，我们可以计算整个样本的代价函数：



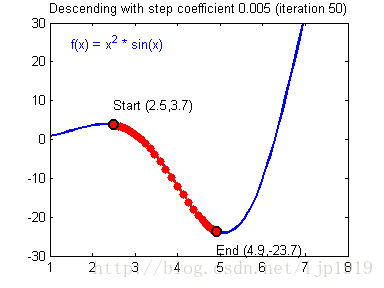
**5.3.3后向传播**

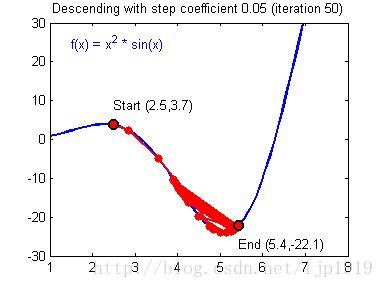
基于前向传播过程中的cache，我们开始计算后向传播。后向传播的公式如下：



**5.3.4更新参数**

上述的结果我们已经可以计算出反向传播的梯度了，那么我们就可以通过梯度下降法对参数进行更新， 其中 α 是学习率， θ 则是代表待更新的参数。选择好的学习率，迭代才会收敛，否则迭代过程不断振荡，呈发散状态。



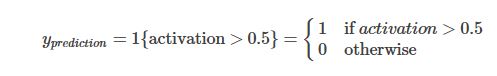


**5.4模型融合，组合成nn\_model()**

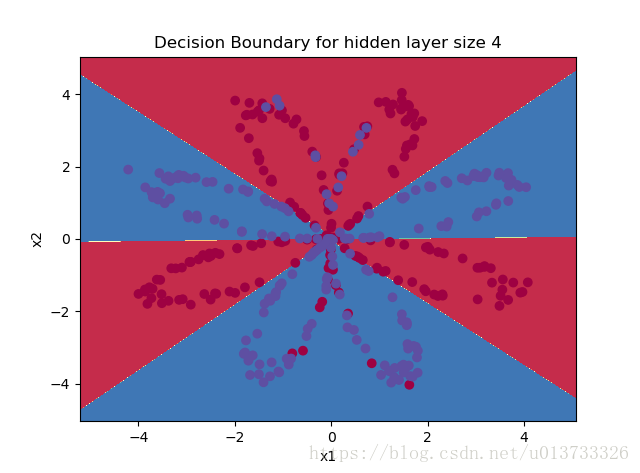
将上述5.1~5.3的各个模块进行整合成一个完整的神经网络模型。

**5.5预测**

至此，我们已经训练获得了最优的参数，那么我们可以基于该模型对新数据进行预测：

****

**5.5.1边界绘制**

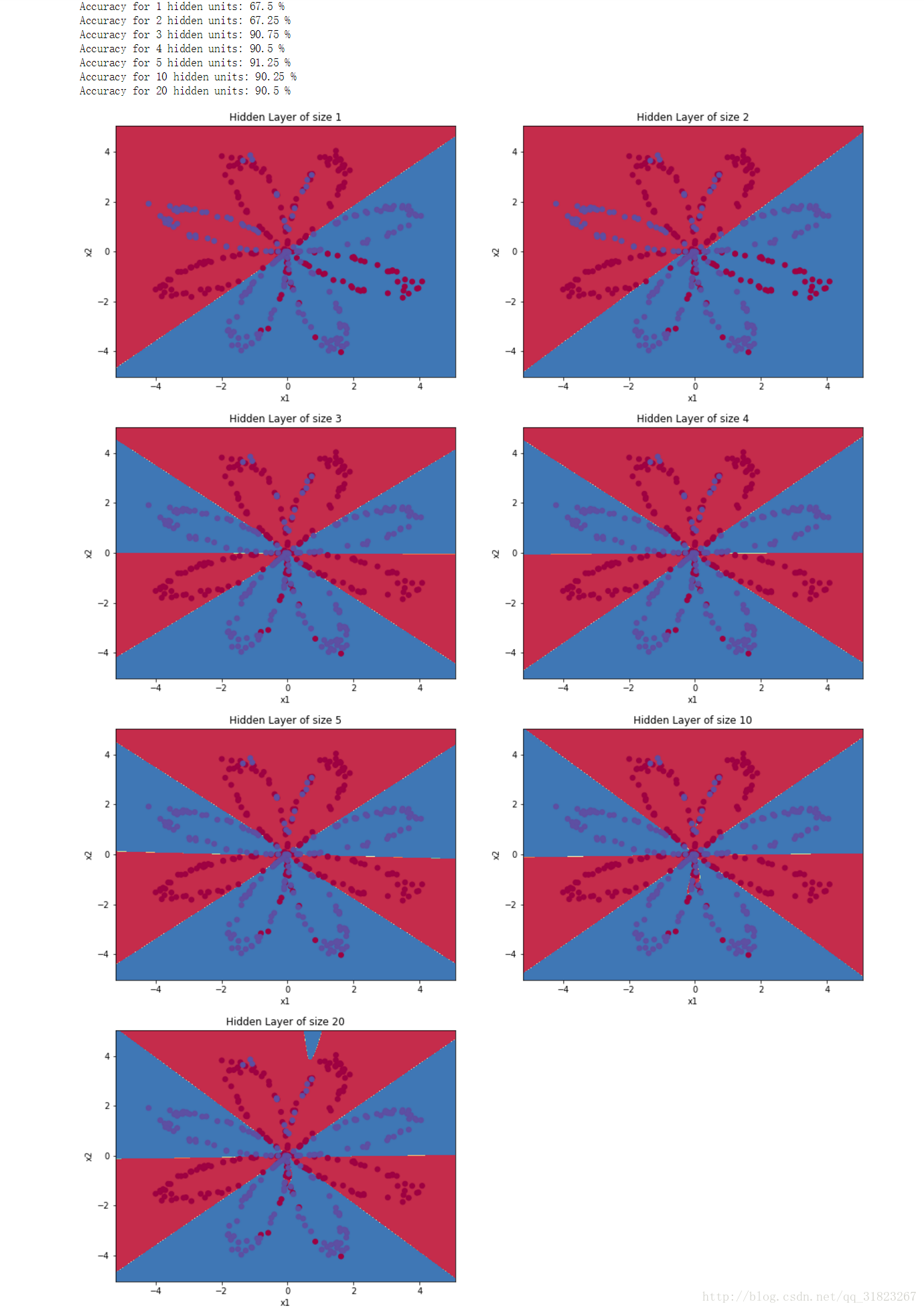


**5.5.2准确率**

统计真实矩阵和预测矩阵中相同位置，值相同的个数。相比47%的逻辑回归预测率，使用含有一个隐藏层的神经网络预测的准确度可以达到90%。

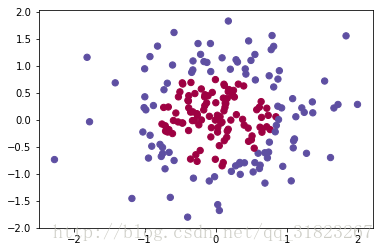
**5.6隐藏层神经元数量的影响（调整隐藏层大小）**

我们在隐藏层的神经元数量分别取1, 2, 3, 4, 5, 10, 20做如下观察：



从上图的对比，我们可以看出，隐藏层的神经元数量越多，则对训练数据集的拟合效果越好，直到最后出现过拟合。本文这里隐藏层的神经元数量，最适值是n\_h=5，即能够较好地拟合训练数据集，也不会出现过拟合现象。另外，对于n\_h过大而产生的过拟合是可以通过正则化来消除的。

**6、其它数据集上的表现**



再次以不同的隐层数量来重新训练模型，查看效果：

