**1 - Packages**

First of all we have to import all the packages that you will need during this assignment.

* [numpy](https://www.numpy.org/) is the fundamental package for scientific computing with Python.
* [sklearn](http://scikit-learn.org/stable/) provides simple and efficient tools for data mining and data analysis.
* [matplotlib](http://matplotlib.org/) is a library for plotting graphs in Python.
* testCases provides some test examples to assess the correctness of your functions
* planar\_utils provide various useful functions used in this assignment

Visualize the dataset using matplotlib. The data looks like a "flower" with some red (label y=0) and some blue (y=1) points. Our goal is to build a model to fit this data. In other words, we want the classifier to define regions as either red or blue.

We have run a logistic regression on the data and it is found out that it is not fit for the data. It reached a accuracy of 47%. So moving on to deep learning.

z[1](i)=W[1]x(i)+b[1](1)(1)z[1](i)=W[1]x(i)+b[1]

a[1](i)=tanh(z[1](i))(2)(2)a[1](i)=tanh⁡(z[1](i))

z[2](i)=W[2]a[1](i)+b[2](3)(3)z[2](i)=W[2]a[1](i)+b[2]

ŷ (i)=a[2](i)=σ(z[2](i))(4)(4)y^(i)=a[2](i)=σ(z[2](i))

y(i)prediction={10if a[2](i)>0.5otherwise (5)(5)yprediction(i)={1if a[2](i)>0.50otherwise

Given the predictions on all the examples, you can also compute the cost JJ as follows:

J=−1m∑i=0m(y(i)log(a[2](i))+(1−y(i))log(1−a[2](i)))(6)(6)J=−1m∑i=0m(y(i)log⁡(a[2](i))+(1−y(i))log⁡(1−a[2](i)))

**Reminder**: The general methodology to build a Neural Network is to:

1. Define the neural network structure ( # of input units, # of hidden units, etc).

2. Initialize the model's parameters

3. Loop:

- Implement forward propagation

- Compute loss

- Implement backward propagation to get the gradients

- Update parameters (gradient descent)