activation.py

```
import numpy as np

class tanh(object):
    @staticmethod #this static method just means we dont have to initialize or pass self, we can do whatever without it
    def calc(v):
        return np.tanh(v)

    @staticmethod
    def calc_deriv(v):
        #calculate d tanh(v)/dv = 1- tanh^2 (v)
        #this is basically dy/dy' = d f(y') / dy' , where y' is the output before activation
        return 1- np.tanh(v)**2

class RELU(object):
    @staticmethod
    def calc(v):
        return np.maximum(0,v)

    @staticmethod
    def calc_deriv(v):
        if v > 0:
            derivative= 1
        else:
            derivative=0
        return derivative

class logistic(object):
    @staticmethod
    def calc(v):
        return 1/(1+np.exp(- v))

    @staticmethod
    def calc_deriv(v):
        return calc(v) * ( 1 - calc(v) )
```