1. **Run a Tensor Flow Algorithm**

   See the Python notebook solutions in section or online!

2. **Deep Dream**

   a. Loop over all images, run them through your deep learning network, and select the one for which the activation of the final neuron is the largest.

   b. Note that: \( h_2 = \sigma(\sum_{i=0}^{64} \theta_i z_i) \). We are going to use gradient ascent to choose pixel values \( \mathbf{x} \) that maximize the activation of the neuron \( h_2 \)!

   \[
   \arg \max_{\mathbf{x}} h_2 = \arg \max_{\mathbf{x}} \sigma(\sum_{i=0}^{64} \theta_{i,2} x_i)
   \]

   Which requires us to solve for the derivative of \( h_2 \) with respect to each pixel \( x_i \).

   \[
   \frac{\partial h_2}{\partial x_i} = \frac{\partial \sigma(z)}{\partial z} \cdot \frac{\partial z}{\partial x_i}
   \]

   \[
   = \sigma(z)[1 - \sigma(z)] \cdot \frac{\partial z}{\partial x_i}
   \]

   \[
   = h_2[1 - h_2] \cdot \theta_i
   \]

   That’s all folks!

   c. The approach is basically the same as in part (b), but we start with the pixels \( \mathbf{x} \) set to the input picture’s pixel values. We then optimize for the output of \( Y = 1 \), the cat neuron (using the same methodology of derivatives as above, but with respect to the final neuron instead of \( h_2 \)). Finally, we only run a few iterations of gradient descent (so that the image is only slightly more catlike, as requested).