Forward Propagation

In this part of the lab, we will implement a neural net's capability to propagate a given input all the way to its last layer. We will refer to this procedure as a forward pass inside a neural network. At every fully connected layer l, we will need to do two things:

1. Perform a matrix multiplication between the activation nodes $a^{(l)}$ in layer l and the fully connected layer parameters $W^{(l)}$ to get the new activation units $z^{(l+1)}$. This can be done by computing the following: $z^{(l+1)} = W^{(l)}a^{(l)}$. The computed result $z^{(l+1)}$ should be stored into variable `nn.z{l+1}` for every layer of the network.
2. Apply a non-linear sigmoid function $a^{(l+1)} = \frac{1}{1 + \exp\left(-z^{(l+1)}\right)}$ on a given input $z^{(l+1)}$ to get activation units $a^{(l+1)}$. Note that your implementation of a sigmoid function must be able to handle multi-dimensional vector inputs (i.e. use '.*' operator in matlab instead of '*'). The computed result $a^{(l+1)}$ should be stored into variable `nn.a{l+1}` for every layer of the network.

Your Function

Save Reset MATLAB Documentation (https://www.mathworks.com/help/)

```
function nn = ForwardPass(nn, x)
    % perform the forward pass inside the network
    %
    % Input:
    % - nn: a structure storing the parameters of the network
    % - x: a feature matrix where every row depicts a data observation, and every column represents a particula
    % Output:
    % - nn: a new neural network variable where the values nn.a{l} are updated for every layer of the network.

    %setting the input to the network
    nn.a{1} = x;
    n_layers=numel(nn.W)+1;


    nn.a{1} = nn.a{1}';

    %% feedforward pass
    for i = 2 : n_layers
       % 1) Performing a matrix multiplication to compute the activation nodes for the current layer
       % use z_{i+1} = W(i)a_i
       nn.z{i} = nn.W{i-1}*nn.a{i-1};

       % 2) Applying a non-linear sigmoid function on every activation node z
       % use a_i = 1/(1+e^{-z_i})
        %octave e.^(...)
       exp_zi = exp(-nn.z{i});
       nn.a{i} = 1./(1+exp_zi);
    end
end

```

Code to call your function

Reset

```
architecture=[336 100 20];
nn = InitializeNetwork(architecture);
random_feature=rand(1,336);
nn = ForwardPass(nn, random_feature);
```

▶ Run Function ?