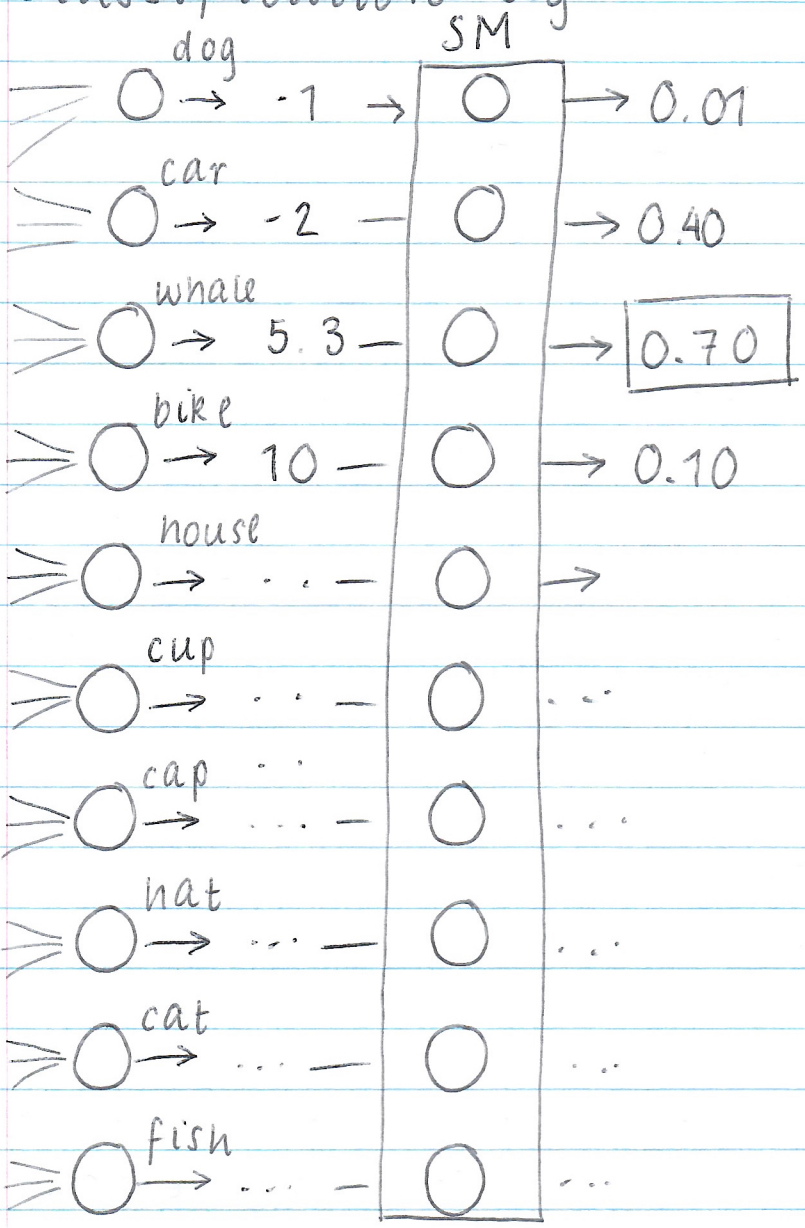
Worksheet 2. Model Selection for Multilayer Perceptrons

- Implement model in 4.3 and evaluate it in 3.6
- Implement model using ReLU, sigmoid, tan

```
net = nn.Sequential(nn.Flatten(),
                    nn.Linear(784, 256),
                    nn.ReLU(),
                    #nn.Sigmoid(),
                    #nn.TanH,
                    #nn.pReLU(),
                    nn.Linear(256,10))
```

- n-dimensional tensor is flattened to 1-dimensional tensor (e.g. image flattened)
- 784 pixels are mapped to 784 neurons
- Those neurons map (each) to 256 outputs
- Here, we introduce a hidden layer with an activation function to decide whether or not neuron should be activated
- Those neurons map (each) to 10 outputs using some activation function:
  - ReLU
  - Sigmoid()
  - ...
- In 3.6, we evaluate the performance of the model by running softmax Regression where each of 10 outputs get turned into a val from 0 to 1 ~~(probability)~~ (so ~~probability)~~ ~~and~~
- When these outputs from Softmax Regression are added → equal 1 (so treat as probability)
- Once softmax is done pick neuron which is most activated (~~closest to~~ highest val)
- The neuron from the model which corresponds to ~~the~~ the most activated

neuron in Softmax Reg. is the classification e.g.



∴ output from model corresponding with ~~neuron 0.70~~ output neuron with val 0.7 ~~in~~ from SM is classification (e.g. whale)