1. Why is it generally preferable to use a Logistic Regression classifier rather than a classical

Perceptron (i.e., a single layer of linear threshold units trained using the Perceptron training

algorithm)? How can you tweak a Perceptron to make it equivalent to a Logistic Regression

classifier?

**Logistic Regression is generally preferable over a classical Perceptron because it not only provides binary classification but also produces probabilities as outputs, which allows for more nuanced predictions. To make a Perceptron equivalent to a Logistic Regression classifier, you can:**

**Replace the step function (used in Perceptron) with the logistic (sigmoid) activation function to produce continuous outputs between 0 and 1.**

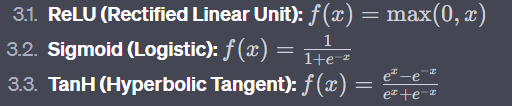
**Modify the training algorithm to use gradient descent and minimize a logistic loss function (cross-entropy) rather than the Perceptron learning rule.**

2. Why was the logistic activation function a key ingredient in training the first MLPs?

**The logistic (sigmoid) activation function was a key ingredient in training the first MLPs because it introduced non-linearity into the model. Without non-linear activation functions, the entire neural network would collapse into a linear model, making it incapable of learning complex patterns.**

3. Name three popular activation functions. Can you draw them?

**Three popular activation functions are:**

****

**Drawing these functions is challenging in text, but you can easily find visual representations online.**

4. Suppose you have an MLP composed of one input layer with 10 passthrough neurons,

followed by one hidden layer with 50 artificial neurons, and finally one output layer with 3

artificial neurons. All artificial neurons use the ReLU activation function.

 What is the shape of the input matrix X?

 What about the shape of the hidden layer’s weight vector Wh, and the shape of its

bias vector bh?

 What is the shape of the output layer’s weight vector Wo, and its bias vector bo?

 What is the shape of the network’s output matrix Y?

 Write the equation that computes the network’s output matrix Y as a function

of X, Wh, bh, Wo and bo.

5. How many neurons do you need in the output layer if you want to classify email into spam

or ham? What activation function should you use in the output layer? If instead you want to

tackle MNIST, how many neurons do you need in the output layer, using what activation

function?

**For binary classification like email spam or ham, you need 1 neuron in the output layer with the sigmoid activation function (to produce a probability).**

**For MNIST with 10 classes, you need 10 neurons in the output layer with the softmax activation function (for multi-class classification).**

6. What is backpropagation and how does it work? What is the difference between

backpropagation and reverse-mode autodiff?

**Backpropagation is a specific algorithm used for training neural networks by computing gradients of the loss with respect to model parameters layer by layer. It is a manual algorithm that requires specifying the derivatives at each layer.**

**Reverse-mode autodiff, also known as automatic differentiation, is a general technique for computing gradients automatically. It leverages the chain rule of calculus to compute gradients efficiently. While backpropagation is a form of autodiff, reverse-mode autodiff can be used for a broader range of mathematical expressions, not limited to neural networks.**

7. Can you list all the hyperparameters you can tweak in an MLP? If the MLP overfits the

training data, how could you tweak these hyperparameters to try to solve the problem?

**Some hyperparameters in an MLP include:**

**Number of hidden layers and neurons in each layer.**

**Learning rate.**

**Activation functions for hidden layers.**

**Batch size.**

**Number of training epochs.**

**Regularization techniques (e.g., dropout, L2 regularization).**

**To address overfitting, you can:**

**Reduce the number of neurons or layers.**

**Increase dropout rates.**

**Apply L2 regularization.**

**Use early stopping.**

8. Train a deep MLP on the MNIST dataset and see if you can get over 98% precision. Try

adding all the bells and whistles (i.e., save checkpoints, restore the last checkpoint in case of

an interruption, add summaries, plot learning curves using TensorBoard, and so on).

**Training a deep MLP on MNIST with over 98% precision involves building a deep neural network, selecting appropriate hyperparameters, and training it. This task is typically done in code with multiple steps and may require substantial code. You can refer to deep learning tutorials and examples for code implementations specific to your platform and framework.**