Chapter 10

Textbook Exercises

Q1 (a): Why is it generally preferable to use a Logistic Regression classifier rather than a classical Perceptron?

Q1 (a): Why is it generally preferable to use a Logistic Regression classifier rather than a classical Perceptron?

A classical Perceptron will converge only if the dataset is linearly separable, and it won't be able to estimate class probabilities. In contrast, a Logistic Regression classifier will generally converge to a reasonably good solution even if the dataset is not linearly separable, and it will output class probabilities.

Q1 (b): How can you tweak a Perceptron to make it equivalent to a Logistic Regression classifier?

Q1 (b): How can you tweak a Perceptron to make it equivalent to a Logistic Regression classifier?

If you change the Perceptron's activation function to the logistic activation function, and if you train it using Gradient Descent, then it becomes equivalent to a Logistic Regression classifier.

Q2: Why was the logistic activation function a key ingredient in training the first MLPs?

Q2: Why was the logistic activation function a key ingredient in training the first MLPs?

The logistic activation function was a key ingredient in training the first MLPs because its derivative is always nonzero, so Gradient Descent can always roll down the slope. When the activation function is a step function, Gradient Descent cannot move, as there is no slope at all.

Q3: Name three popular activation functions. Can you draw them?

Q3: Name three popular activation functions. Can you draw them?

Popular activation functions include the step function, the logistic (sigmoid) function, the hyperbolic tangent (tanh) function, and the Rectified Linear Unit (ReLU) function.

Q4: 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.

Q4 (a): What is the shape of the input matrix X?

Q4 (a): What is the shape of the input matrix $\boldsymbol{X}$?

The shape of the input matrix $\boldsymbol{X}$ is $m \times 10$, where m represents the training batch size.

Q4 (b): What are the shapes of the hidden layer's weight vector W_h and its bias vector b_h?

Q4 (b): What are the shapes of the hidden layer's weight vector W_h and its bias vector b_h?

The shape of the hidden layer's weight vector $\boldsymbol{W}_h$ is 10×50, and the length of its bias vector $\boldsymbol{b}_h$ is 50.

Q4 (c): What are the shapes of the output layer's weight vector W_o and its bias vector b_o?

Q4 (c): What are the shapes of the output layer's weight vector W_o and its bias vector b_o?

The shape of the output layer's weight vector $\boldsymbol{W}_o$ is 50×3, and the length of its bias vector $\boldsymbol{b}_o$ is 3.

Q4 (d): What is the shape of the network's output matrix Y?

Q4 (d): What is the shape of the network's output matrix $\boldsymbol{Y}$?

The shape of the network's output matrix $\boldsymbol{Y}$ is $m \times 3$.

Q4 (e): Write the equation that computes the network's output matrix Y as a function of X, W_h, b_h, W_o, and b_o.

Q4 (e): Write the equation that computes the network's output matrix $\boldsymbol{Y}$ as a function of $\boldsymbol{X}$, $\boldsymbol{W}_h$, $\boldsymbol{b}_h$, $\boldsymbol{W}_o$, and $\boldsymbol{b}_o$.

$\boldsymbol{Y} = \text{ReLU}(\,\text{ReLU}(\boldsymbol{X}\boldsymbol{W}_h + \boldsymbol{b}_h)\,\boldsymbol{W}_o + \boldsymbol{b}_o\,)$. Recall that the ReLU function just sets every negative number in the matrix to zero. Also note that when you are adding a bias vector to a matrix, it is added to every single row in the matrix, which is called broadcasting.

Q5 (a): How many neurons do you need in the output layer if you want to classify email into spam or ham?

Q5 (a): How many neurons do you need in the output layer if you want to classify email into spam or ham?

You just need one neuron in the output layer of a neural network—for example, indicating the probability that the email is spam. You would typically use the logistic activation function in the output layer when estimating a probability.

Q5 (b): If instead, you want to tackle MNIST, how many neurons do you need in the output layer, and which activation function should you use?

Q5 (b): If instead, you want to tackle MNIST, how many neurons do you need in the output layer, and which activation function should you use?

You need 10 neurons in the output layer, and you must replace the logistic function with the softmax activation function, which can handle multiple classes, outputting one probability per class.

Q5 (c): What about for getting your network to predict housing prices, as in Chapter 2?

Q5 (c): What about for getting your network to predict housing prices, as in Chapter 2?

You need one output neuron, using no activation function at all in the output layer.

Q6: What is backpropagation and how does it work?

Q6: What is backpropagation and how does it work?

Backpropagation is a technique used to train artificial neural networks. It first computes the gradients of the cost function with regard to every model parameter (all the weights and biases), then it performs a Gradient Descent step using these gradients. This backpropagation step is typically performed thousands or millions of times, using many training batches, until the model parameters converge to values that (hopefully) minimize the cost function.

Q7 (a): Can you list all the hyperparameters you can tweak in a basic MLP?

Q7 (a): Can you list all the hyperparameters you can tweak in a basic MLP?

Here is a list of all the hyperparameters you can tweak in a basic MLP: the number of hidden layers, the number of neurons in each hidden layer, and the activation function used in each hidden layer and in the output layer. In general, the ReLU activation function is a good default for the hidden layers. For the output layer, in general you will want the logistic activation function for binary classification, the softmax activation function for multiclass classification, or no activation function for regression.

Q7 (b): If the MLP overfits the training data, how could you tweak these hyperparameters to try to solve the problem?

Q7 (b): If the MLP overfits the training data, how could you tweak these hyperparameters to try to solve the problem?

You can try reducing the number of hidden layers and reducing the number of neurons per hidden layer.