16ML Perceptron Quiz Answer Key

1. (a) We assume that the data is linearly separable.
2. The learning rate is a hyper parameter that allows us to control how much each misclassified training point affects the old weight. It is used during the update stage by scaling how much we add to the old weight. Decreasing the learning rate will decrease the amount we change during every update step, and could lead to slower convergence.
3. There is one weight for every input (and possible an extra weight if we include the bias term). We put the weights into a vector and take the dot product of this weight vector with the input. This is our weighted sum. By using the activation function on this weighted sum, we can get the predicted output of our input.
4. There is no guarantee on the optimality of the solution. The only thing that is guaranteed is convergence of the Learning Algorithm if we have linearly separable data. This means that although the Learning Algorithm may converge, the weights that it generates may not be the best (for example, outliers may affect it a lot) for the data set. Soon you will learn about SVM's that take into account optimality.
5. We append a constant term to every input and add an extra weight variable to the weight vector to account for bias. This way when we take the dot product of the weight vector and the input vector, we can learn an extra scalar which in 2-d represents where the decision boundary crosses the y-axis.
6. The activation function allows us to classify the weighted sum by sorting it into one of two classes. We can use various activation functions such as the step function, but in the note we use the sign function in our derivations. Thus it is not unique.
7. In order for the Perceptron Learning Algorithm to stop iterating, there needs to be one round where every training point is correctly classified. Only when we have linearly separable data is the convergence guaranteed.
8. We can view our problem as an optimization problem where we are trying to minimize the amount of misclassified points. Our update step is traveling in the negative gradient of the cost function of this optimization problem.

9. Taking the dot product of $\mathbf{x}_i$ and $\mathbf{w}$ we get a value of -0.426. Because this is less than 0, we classify our point as label -1.

10. To extend the algorithm, we add a weight vector for each class. Then, in order to classify some input, we take the argmax of the dot product of every weight vector and the input. We also change the update algorithm slightly by adding to the correct weight vector when we misclassify and subtracting from the incorrect ones.

11. (a) $\mathbf{x}_i^T\mathbf{w}_1 = 0.696$, $\mathbf{x}_i^T\mathbf{w}_2 = 1.347$, $\mathbf{x}_i^T\mathbf{w}_3 = -1.095$, $\mathbf{x}_i^T\mathbf{w}_4 = 1.848$

    (b) The current weight scheme classifies $\mathbf{x}_i$ as class 4. This is incorrect because the correct class should be class 3.

    (c) Because we incorrectly classified, we need to update the weights. Remember that for the correct weight we add the input vector scaled by the learning rate, and for the rest of the weights we subtract this value. The new weights are

$$\mathbf{w}_1 = \begin{bmatrix} -1.8 \\ -0.08 \end{bmatrix}, \mathbf{w}_2 = \begin{bmatrix} -1.39 \\ -0.78 \end{bmatrix}, \mathbf{w}_3 = \begin{bmatrix} 1.56 \\ 0.43 \end{bmatrix}, \mathbf{w}_4 = \begin{bmatrix} -1.27 \\ 0.05 \end{bmatrix}$$

    (d) $\mathbf{x}_i^T\mathbf{w}_1 = 0.7$, $\mathbf{x}_i^T\mathbf{w}_2 = 1.977$, $\mathbf{x}_i^T\mathbf{w}_3 = -1.32$, $\mathbf{x}_i^T\mathbf{w}_4 = 0.281$

    (e) The weight scheme classifies the point as class 2. This is the correct label.

    (f) Because the point was correctly classified we don't need to update the labels. Thus our weights stay the same as

$$\mathbf{w}_1 = \begin{bmatrix} -1.8 \\ -0.08 \end{bmatrix}, \mathbf{w}_2 = \begin{bmatrix} -1.39 \\ -0.78 \end{bmatrix}, \mathbf{w}_3 = \begin{bmatrix} 1.56 \\ 0.43 \end{bmatrix}, \mathbf{w}_4 = \begin{bmatrix} -1.27 \\ 0.05 \end{bmatrix}$$