Lecture 1 Notes

Deep Learning AI335

1 Perceptron

1.1 The Basic Perceptron

The basic perceptron is a single-layer model that takes in multiple input features $\mathbf{x} \in \mathbb{R}^d$ and produces a binary output $y \in \{-1, 1\}$:

$$y = \operatorname{sign}\left(\left(\sum_{i=1}^{d} w_i \cdot x_i\right) + b\right), \tag{1}$$

or:

$$y = \operatorname{sign}(w^\top \cdot x + b), \tag{2}$$

where b is a bias term added to shift the decision boundary.

The perceptron uses a linear boundary to separate classes. For example, if you have a two-dimensional input, the decision boundary would be a line; in three dimensions, it would be a plane, and so on.

The perceptron learns using the Perceptron Learning Algorithm (PLA), which iteratively adjusts the weights and bias to minimize classification error. PLA starts with random (or 0) values for w and b, for each example (x_i, y_i) in the training set $\mathcal{D}$ we calculate the predicted class $\hat{y}$

$$\hat{y} = \operatorname{sign}(w^\top \cdot x + b) \tag{3}$$

For any misclassified points $\hat{y} \neq y$ we update the weight and threshold as follows

$$w \leftarrow w + \eta \cdot y_i \cdot x_i \tag{4}$$

$$b \leftarrow b + \eta \cdot y_i \tag{5}$$

Where η (eta) is the learning rate.

1.2 Limitations of Perceptrons

Since a single-layer perceptron can only learn patterns that are linearly separable, if the data classes are not linearly separable (like in the XOR problem), the perceptron cannot classify them correctly regardless of how it's trained.

The problem of non-linearly separable data seems to be solved by using more layers—a multilayer perceptron (MLP)—since multiple layers would be able to capture a non-linear decision boundary. However the PLA as described is not capable of training such a model, since it has no *ground truth* to check against for the intermediate layers.