Logistic Regression

Logistic regression is a probabilistic model that classifies the instances in terms of probabilities. Because the classification is probabilistic, a natural approach for optimizing the parameters is to ensure that the predicted probability of the observed class for each training instance is as large as possible. This goal is achieved by using the notion of maximum likelihood estimation in order to learn the parameters of the model. The likelihood of the training data is defined as the product of the probabilities of the observed labels of each training instance. Clearly, larger values of this objective function are better. By using the negative logarithm of this value, one obtains a loss function in minimization form. Therefore, the output node uses the negative log-likelihood as a loss function. This loss function replaces the squared error used in the Widrow-Hoff method. The output layer can be formulated with the sigmoid activation function, which is very common in neural network design.

Let $(\overline{X_1}, y_1) \ldots (\overline{X_n}, y_n)$ be a set of n training pairs in which $\overline{X_i}$ contains the d-dimensional features and $y_i \in \{-1, +1\}$ is a binary class variable. As in the case of a perceptron, a single-layer architecture with weights $\overline{W} = (w_1 \ldots w_d)$ is used. Instead of using the hard sign activation on $\overline{W} \cdot \overline{X_i}$ to predict y_i, logistic regression applies the soft sigmoid function to $\overline{W} \cdot \overline{X_i}$ in order to estimate the probability that y_i is 1,

$$\hat{y}_i = P(y_i = 1) = \frac{1}{1+exp(-\overline{W} \cdot \overline{X_i})}$$

For a test instance, it can be predicted to the class whose predicted probability is greater than 0.5. Note that $P(y_i = 1)$ is 0.5 when $\overline{W} \cdot \overline{X_i} = 0$, and $\overline{X_i}$ lies on the separating hiperplane. Moving $\overline{X_i}$ in either direction from the hyperplane results in different signs of $\overline{W} \cdot \overline{X_i}$ and corresponding movements in the probability values. Therefore, the sign of $\overline{W} \cdot \overline{X_i}$ also yields the same prediction as picking the class with probability larger than 0.5.

We will now describe how the loss function corresponding to likelihood estimation is set up. This methodology is important because it is used widely in many neural models. For positive samples in the training data, we want to maximize $P(y_i = 1)$ and for negative samples, we want to maximize $P(y_i = -1)$. For positive samples satisfying $y_i = 1$, one wants to maximize $\hat{y}_i$ and for negative samples satisfying $\hat{y}_i = -1$, one wants to maximize $1 - \hat{y}_i$. One can write this casewise maximization in the form of a consolidated expression ofalways maximizing $|\frac{y_i}{2} - \frac{1}{2} + \hat{y}_i|$.

The products of these probabilities must be maximized over all training instances to maximize the likelihood $\mathcal{L}$,

- $\mathcal{L} = \prod_{i=1}^{n} |\frac{y_i}{2} - \frac{1}{2} + \hat{y}_i| \Rightarrow -\log(\mathcal{L}) = \sum_{i=1}^{n} -\log(|\frac{y_i}{2} - \frac{1}{2} + \hat{y}_i|) = \sum_{i=1}^{n} L_i$

Additive forms fo the objective function are particularly convenient for the types of stochastic gradient updates that are common in neural networks. For each training instance, the predicted probability $\hat{y}_i$ is computed by passing it through the neural network, and the loss is used to determine the gradient for each training instance.

The gradient of L_i with respect to the weights in $\overline{W}$ can be computed as follows:

$$\frac{\partial L_i}{\partial \overline{W}} = -\frac{sign(\frac{y_i}{2} - \frac{1}{2} + \hat{y}_i)}{|\frac{y_i}{2} - \frac{1}{2} + \hat{y}_i|} \cdot \frac{\partial \hat{y}_i}{\partial \overline{W}}$$