Assignement -11

Dhanush V Nayak
ee23btech11015@iith.ac.in

1 Regularization for Linear Regression

With univariate linear regression, we did not have to worry about overfitting. However, when moving to multivariable linear regression in high-dimensional spaces, there is a risk that some irrelevant dimensions may appear useful by chance, leading to overfitting. To prevent this, we use regularization techniques.

$$\text{Cost}(h) = \text{EmpLoss}(h) + \lambda \cdot \text{Complexity}(h)$$

In the case of linear functions, the complexity is a function of the weights. A family of regularization functions can be defined as:

$$\text{Complexity}(h_w) = L_q(w) = \sum_i |w_i|^q$$

There are two main types of regularization

- L1 regularization (with $q = 1$): Minimizes the sum of the absolute values of the weights.
- L2 regularization (with $q = 2$): Minimizes the sum of the squared values of the weights.

L1 regularization has an important advantage it often leads to a sparse model by setting many weights to zero, thus identifying irrelevant attributes. This sparsity makes models easier to interpret and less prone to overfitting.

Why L1 regularization often results in zero weights, is explained using a picture in the book: In two-dimensional weight space, L1 complexity forms a diamond-shaped constraint, while L2 complexity forms a circular one. The solution must lie within the boundary when minimizing the loss function subject to a complexity constraint. With L1, the solution often falls at the corner of the diamond, where some weights are zero. On the other hand, L2 does not tend to produce zero weights because its circular shape lacks such "corners."

Another distinction is that L1 regularization is not rotationally invariant, meaning the result would change if the axes are rotated. L2 regularization, being spherical, is rotationally invariant.

2 Linear classifiers with a hard threshold

Linear functions can be used for both regression and classification. For example, we can classify seismic events like earthquakes and explosions using two inputs, x_1 and x_2, which represent different wave magnitudes. A hypothesis $h(x)$ will output 0 for earthquakes and 1 for explosions.The key idea in classification is the decision boundary, which separates the two classes. In the case of linear classification, this boundary is a line. For the given example, the line can be written as:

$$x_2 = 1.7x_1 - 4.9 \quad \text{or} \quad -4.9 + 1.7x_1 - x_2 = 0$$

$$h(x) = \begin{cases} 1 & \text{if} \quad -4.9 + 1.7x_1 - x_2 > 0 \quad \text{(explosion)} \\ 0 & \text{if} \quad -4.9 + 1.7x_1 - x_2 \leq 0 \quad \text{(earthquake)} \end{cases}$$

This equation can also be expressed in a vector form. By introducing $x_0 = 1$, we rewrite the equation as:

$$-4.9x_0 + 1.7x_1 - x_2 = 0$$

The weights vector $w = [-4.9, 1.7, -1]$ allows us to express the hypothesis as:

$$h_w(x) = \begin{cases} 1 & \text{if} \quad w \cdot x \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

This is essentially passing the linear function $w \cdot x$ through a threshold function, where the output is 1 if the result is positive, and 0 otherwise.

To learn the correct weights for classification, we use the perceptron learning rule. For a given example (x, y), the weight update is:

$$w_i \leftarrow w_i + \alpha(y - h_w(x))x_i$$

This training rule is applied by choosing examples at random just as we did in stochastic gradient descent.Then they are plotted on a training curve. A training curve measures the classifier performance on a fixed training set and updates on that. In general, the perceptron rule may not converge to a stable solution for fixed learning rate α, but if α decays as $O(t^{-1})$ where t is the iteration number, then the rule can be shown to converge to a minimum-error solution when examples are presented in a random sequence.

3 Linear classification with logistic regression

The perceptron model faces several limitations, such as non-differentiability due to the hard threshold function, making learning unpredictable. To address this, the threshold can be softened using a continuous and differentiable function like the logistic function:

$$\text{Logistic}(z) = \frac{1}{1 + e^{-z}}$$

The logistic function offers smoother behavior and is mathematically more convenient. With this soft threshold, the hypothesis for logistic regression becomes:

$$h_w(x) = \text{Logistic}(\mathbf{w} \cdot \mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w} \cdot \mathbf{x}}}$$

The output of this function, which lies between 0 and 1, can be interpreted as the probability of belonging to a particular class, forming a soft boundary in the input space. The process of fitting the weights to minimize loss is known as logistic regression. Unlike linear regression, there is no closed-form solution, so gradient descent is used to minimize the loss. The loss function used is L_2 loss, and the derivative of the logistic function, satisfies:

$$g'(z) = g(z)(1 - g(z)) = h_w(x)(1 - h_w(x))$$

Thus, the weight update rule for logistic regression, using gradient descent, becomes:

$$w_i \leftarrow w_i + \alpha(y - h_w(x))h_w(x)(1 - h_w(x))x_i$$

In experiments comparing logistic regression with a linear classifier, logistic regression converges more slowly in the linearly separable case but does so more predictably. For noisy, non-separable data, logistic regression demonstrates faster and more reliable convergence.