AI1001 - Assignment 11

Shivram S
ai24btech11031@iith.ac.in

1 Regularization in multivariate linear regression

With univariate linear regression, we don't have to worry about overfitting, but with multivariate linear regression it is possible that some dimension that is irrelevant appears to be useful, leading to overfitting.

We can specify the complexity as a function of the weights:

$$Complexity(h_{\mathbf{w}}) = L_q(\mathbf{w}) = \sum_i |w_i|^q$$

With $q = 1$, we have L_1 regularization; with $q = 2$ we have L_2 regularization, and so on. The choice of regularization function depends on the specific problem, but L_1 regularization has the advantage of producing a **sparse model** where many weights are set to zero. This makes the model less likely to overfit, and makes it easier for a human to understand.

The cost function can be equivalent to minimizing $Loss(\mathbf{w})$ subject to the constraint that $Complexity(\mathbf{w}) \leq c$ for some constant c. The set of points that have L_1 complexity less than c is diamond-shaped, and the corners, where some values are zero, have a tendency to be closer to the minimum. This explains why the L_1 complexity measure produces a spare model.

The L_2 complexity measure is spherical, which makes it rotationally invariant. This is appropriate when the choice of axes is arbitrary. The L_1 function is not rotationally invariant and is appropriate when the axes are not interchangeable.

2 Linear classifiers with a hard threshold

Linear functions can be used to do classification. The task of classification is to learn a hypothesis h that takes some input values and returns the class to which the input belongs. A **decision boundary** is a surface that separate two classes. A linear decision boundary is called a **linear separator** and data that can be separated by such a boundary are called **linearly separable**.

We can define the vector of weights $\mathbf{w} = \langle w_1, w_2 \rangle$ and write the classification hypothesis, $h_{\mathbf{w}}$ as;

$$h_{\mathbf{w}}(\mathbf{x}) = 1 \text{ if } \mathbf{w} \cdot \mathbf{x} \geq 0 \text{ and 0 otherwise}$$

We can also think about it in terms of a threshold function

$$h_{\mathbf{w}}(\mathbf{x}) = Threshold(\mathbf{w} \cdot \mathbf{x}) \text{ where } Threshold(z) = 1 \text{ if } z \geq 0 \text{ and 0 otherwise}$$

We now need to choose **w** to minimize the loss. Provided that the data is linearly separable, we can use the **perceptron learning rule** to converge to a solution.

$$w_i \leftarrow w_i + \alpha(y - h_{\mathbf{w}}(\mathbf{x})) \times x_i$$

Preprint. Under review.

The working of this rule can be explained as:

- If the output is correct ($y = h_{\mathbf{w}}(\mathbf{x})$), then the weights are not changed.
- If y is 1 but $h_{\mathbf{w}}(\mathbf{x})$ is 0, then we want to increase w_i so that $\mathbf{w} \cdot \mathbf{x}$ becomes bigger and $h_{\mathbf{w}}(\mathbf{x})$ outputs 1.
- If y is 0 but $h_{\mathbf{w}}(\mathbf{x})$ is 1, then we want to decrease w_i so that $\mathbf{w} \cdot \mathbf{x}$ becomes smaller and $h_{\mathbf{w}}(\mathbf{x})$ outputs 0.

We can visualize we learning rule using a **learning curve** which plots the proportion of correct classifications against the number of weight updates. If the data points are not linearly separable then the perceptron rule fails to converge. We can reach a minimum error solution if the learning rate α decays with the iteration number as $\mathcal{O}(1/t)$, such as $\alpha(t) = 1000/(1000 + t)$.

3 Linear classification with logistic regression

The hard nature of the threshold makes learning with the perceptron rule very unpredictable. It would be better if we could classify some examples as unclear borderline cases. This can be done by softening the threshold function. The logistic function is commonly used due to its convenient mathematical properties

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

The derivative $g'(x)$ satisfies $g'(z) = g(z)(1 - g(z))$, so we can write our update rule as

$$w_i = w_i + \alpha(y - h_{\mathbf{w}}(\mathbf{x})) \times h_{\mathbf{w}}(\mathbf{x})(1 - h_{\mathbf{w}}(\mathbf{x})) \times x_i$$

Logistic regression is slower to converge, but behaves more predictably. When the data is non-separable, logistic regression converges far more quickly. This has led to logistic regression becoming one of the most popular classification techniques in medicine, marketing, survey analysis, etc.