Regularisation

Shuvrajeet Das

June 2021

Abstract

In mathematics, statistics, finance, computer science, particularly in machine learning and inverse problems, regularization is the process of adding information in order to solve an ill-posed problem or to prevent overfitting.

1 Introduction

Regularization can be applied to objective functions in ill-posed optimization problems. The regularization term, or penalty, imposes a cost on the optimization function to make the optimal solution unique.

Independent of the problem or model, there is always a data term, that corresponds to a likelihood of the measurement and a regularization term that corresponds to a prior. By combining both using Bayesian statistics, one can compute a posterior, that includes both information sources and therefore stabilizes the estimation process. By trading off both objectives, one chooses to be more addictive to the data or to enforce generalization (to prevent over fitting)

2 Mathematics behind the scenes

Assumptions: Logistic Regression makes certain key assumptions before starting its modeling process:

- The labels are almost linearly separable.
- The observations have to be independent of each other.
- There is minimal or no multicollinearity among the independent variables.
- The independent variables are linearly related to the log odds.

Hypothesis: We want our model to predict the probability of an observation belonging to a certain class or label. As such, we want a hypothesis h that satisfies the following condition $0 \leq h(x) \leq 1$, where x is an observation.

We define $h(x) = g(w^T * x + b)$, where g is a sigmoid function and w are the trainable parameters. As such, we have:

$$h(x) = \frac{1}{1 + e^{-(w^T x + b)}} \tag{1}$$

The cost for an observation:

Now that we can predict the probability for an observation, we want the result to have the minimum error. If the class label is y, the cost (error) associated with an observation x is given by:

$$Cost(h(x), y) = \begin{cases} -log(h(x) & \text{if } y = 1 \\ -log(1 - h(x)) & \text{if } y = 0 \end{cases}$$

Cost Function: Thus, the total cost for all the m observations in a dataset

$$\begin{aligned} l(b, W) &= \log L(b, w) \\ &= -\sum_{i=0}^{n} y_i \log h(x_i) + (1 - y_i) \log(1 - h(x_i)) \\ &= -\sum_{i=0}^{n} \log(1 - h(x_i)) - \sum_{i=0}^{n} y_i \log \frac{h(x_i)}{1 - h(x_i)} \end{aligned} \tag{2}$$

Gradient Descent:

Gradient descent is an optimization algorithm used to minimize some function by iteratively moving in the direction of steepest descent as defined by the negative of the gradient. We will update each of the params w_i using the following template:

$$\frac{\partial}{\partial w_i} = \frac{1}{m} \sum_{j=1}^{m} (h(x^j) - y^j) x_i^j \tag{3}$$

But there is also an undesirable outcome associated with the above gradient descent steps. In an attempt to find the best h(x), the following things happen:

CASE I: For class label = 0
h(x) will try to produce results as close 0 as possible
As such, $w^T.x + b$ will be as small as possible
$\implies$ W_i will tend to -∞

CASE II: For class label = 1
h(x) will try to produce results as close 1 as possible
As such, $w^T.x + b$ will be as large as possible
$\implies$ W_i will tend to +∞

This leads to a problem called overfitting, which means, the model will not be able to generalize well, i.e. it won't be able to correctly predict the class label for an unseen observation. So, to avoid this we need to control the growth of the params w_i.

3 Regularisation:

Regularization is a technique to solve the problem of overfitting in a machine learning algorithm by penalizing the cost function. It does so by using an additional penalty term in the cost function.

There are two types of regularization techniques:

- Lasso or L1 Regularization
- Ridge or L2 Regularization (we will discuss only this in this article)

So, how can L2 Regularization help to prevent overfitting? Let's first look at our new cost function:

$$Cost = -\frac{1}{m}\sum_{j=1}^{m} y_i \log h(x_i) + (1-y_i)\log(1-h(x_i)) + \frac{\lambda}{2m}\sum_{j=1}^{m} w_j^2 \qquad (4)$$

λ is called the regularization parameter. It controls the trade-off between two goals: fitting the training data well vs keeping the params small to avoid overfitting.

Hence, the gradient of J(w) becomes:

$$\frac{\partial}{\partial w_i} = \frac{1}{m}\sum_{j=1}^{m}(h(x^j) - y^j)x_i^j + \lambda w_i \qquad (5)$$

The regularization term will heavily penalize large w_i. The effect will be less on smaller w_i. As such, the growth of w is controlled. The h(x) we obtain with these controlled params w will be more generalized.

NOTE: λ is a hyper-parameter value. We have to find it using cross-validation.

Larger value λ of will make w_i shrink closer to 0, which might lead to under fitting.

$\lambda = 0$, will have no regularization effect.

When choosing λ, we have to take proper care of bias vs variance trade-off.

4 Conclusion

Hence a major problem of bias vs variance trade-off can be solved.