Random Thoughts on Machine Learning

A Layman's Perspective

Yu Liu[1]

May 6, 2016

[1]`https://github.com/ohliumliu`

Contents

List of Figures

List of Tables

Preface

This is a collection of ongoing thoughts I gathered on my journey to better understand machine learning. I have been trying to relate with my past training in math and physics and these thoughts, naive as it may appear to professionals, might be helpful to get my feet into the door of this field.

1

Regularization: the “hidden" facts?

1.1 Motivation of regularization

It is well known that introducing too many features in a model will almost always lead to perfect score with the training and validation set, but miserable failure with the test set. In another word, a feature-rich model has a very small bias (toward a particular model), but a very high variance (when used for prediction). The high variance probably comes from the observation that the confidence region of the fitted parameters is big enough to overwhelm the average value.

If there is no way or rationale to reduce the number of features, regularization is often applied to solve the problem of over-fitting. I have a feeling that this amounts to the introduction of extra data points to the modeling problem, and this chapter is a summary of my attempt to formulate this feeling.

1.2 Linear model

I will start with the simple case of a a linear model with a feature space $\vec{x} \equiv (x_1, \cdots, x_n)^T$ and the corresponding set of coefficient vector $\vec{\theta} \equiv (\theta_1, \cdots, \theta_n)^T$. The prediction of the model is simply linear with a constant offset θ_0:

$$h(\vec{x}; \vec{\theta}) = \theta_0 + \vec{\theta} \cdot \vec{x} \tag{1.1}$$

Given m data points $(\vec{x}^{(m)}, y^{(m)})$ with $\vec{x}^{(m)} = (\vec{x}_1^{(m)}, \cdots, \vec{x}_n^{(m)})^T$, the cost function without regularization reads

$$J(\vec{\theta}) = \sum_{i=1}^{m} \left[y^{(i)} - h(\vec{x}^{(i)}; \vec{\theta}) \right]^2 . \tag{1.2}$$

With regularization, an extra term is added to penalize non-zero values of θ_i,

$$J(\vec{\theta}) = \sum_{i=1}^{m} \left[y^{(i)} - h(\vec{x}^{(i)}; \vec{\theta}) \right]^2 + \lambda \sum_{i=0}^{n} \theta_i^2, \tag{1.3}$$

As an aside, we can vectorize the cost function by first defining an error vector $(m \times 1)$,

$$\vec{\epsilon} = \vec{y} - \left[\vec{\theta_0} + \vec{\theta}^T \cdot \mathbf{X} \right], \tag{1.4}$$

where $\vec{\theta_0} \equiv \theta_0 \times (1, \cdots, 1)$ and $\mathbf{X}$ is an $n \times m$ matrix which each column as $\vec{x}^{(m)}$. In this way, the cost function is

$$J(\vec{\theta}) = \vec{\epsilon} \cdot \vec{\epsilon} + \lambda(\theta_0^2 + \vec{\theta} \cdot \vec{\theta}). \tag{1.5}$$

It's also worth noting that the penalty term could take other forms. For example, in Lasso regression, the L_1 norm (absolute values) of θ_i is used.

Going back to the regularizaton process, the factor λ is a tunable parameter to control the biasness of the regression. I will try to show that, in this context of simple linear regression, the introduction of regularization term seems to be equivalent to the “artificial" addition of extra data points.

In order to demonstrate this point, we need to find n data points that can lead to the extra penalty term $\lambda(\theta_0^2 + \vec{\theta} \cdot \vec{\theta})$.

For example, we want to find $(\vec{u}^{(1)}, v^{(1)})$ that satisfies

$$\left[v^{(1)} - \left(\theta_0 + \vec{\theta} \cdot \vec{v}^{(1)}\right)\right]^2 = \lambda\theta_1^2. \tag{1.6}$$

An obvious choice that is independent θ_i is $\vec{u}^{(1)} = (\sqrt{\lambda}, 0, \cdots)$ and $v^{(1)} = \theta_0$. In general, we would like to have $\vec{u}^{(i)} = \{\sqrt{\lambda}\delta_{ij}\}_j$ and $v^{(i)} = \theta_0$.

With this choice of points, the regularization cost can be compared with adding some extra points to the training set. In the space spanned by the features, these points share the same predicted value or θ_0 and distributed on each of the independent axis. It also includes the origin. The adoption of regularization essentially claims that there is no dependence of the prediction on any of the features. The hypothesis, of course, is just the opposite, hence the hypothesis is associated with some “bias". The combination of agnostic regularization and the biased hypothesis reminds me of the central idea of Bayes statistics. In Bayes, the prior knowledge has to be “unbiased", and the training set makes the posteori knowlege more “biased". Here, the two parts are mixed together in the hope to achieve a balance.

The strength of the agnostic regularization is adjusted by λ. In the interpretation above, it translates to the distance of the extra points from the origin. In the hyper-plane spanned by the features and the prediction, these extra points, $(\vec{u}^{(i)}, v^{(i)})$ $(i = 0, \cdots, n)$, define a hyperplane which is “flat" (agnostic). The size of the plane is $\sqrt{\lambda}$. If λ is bigger, it will be more important in the regression because it covers a larger range of feature values. When it is smaller, it will not be very signficant compared with the original training points. In the extreme case that $\lambda \sim 0$, these extra points gather around $(0, \theta_0)$, no longer serving as extra constraints.

An immediate application of this interpretation is the choice of λ. It should be based on the distribution of feature values. Because there is only one lambda available, it is obviously beneficial to perform feature scaling. Or else, there is no way to choose a universal λ that is appropriate for all features. Of course, we could use a different λ for different θ, but this is adding to the complexity of the model furthermore.

In practice, it may be helpful to study the response of θ_i with respect to the change in λ. As λ increase, some coefficients may be more resilent than others, and these must be more important features. For those θ_i that approaches zero quickly, we might remove them just as well.

Another potential development is to dig deeper into the Bayes interpretation. Is there a way to adjust λ based on training set in iterations: starting from a large λ, as more training data are fed to the program, λ is reduced. I am not sure if this makes sense.

1.3 Polynomial model

Without increasing feature set, one can increase the complexity of the model, for example, by increasing the polynomial order.

2

Principle component analysis: Moment of inertia and rotational spectroscopy

3

Entropy cost: Boltzmann or Shannon?

4

Feature scaling: the Π theorem

5

k-mean: the Harmonic mean