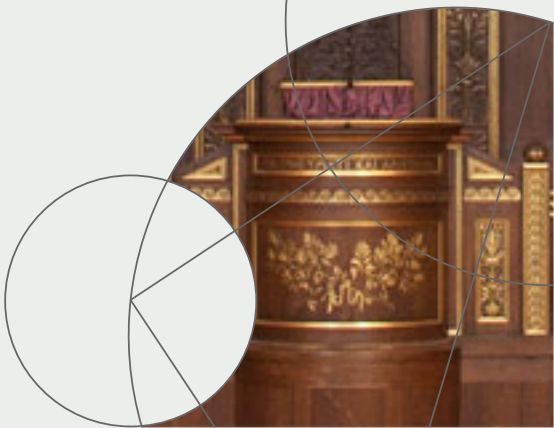
STEVENS INSTITUTE OF TECHNOLOGY

Regularization: the Path to Bias-Variance Tradeoff

Rensheng Wang,
https://sit.instructure.com/courses/34886



Underfitting Example

- Suppose we want to predict the salary of someone based on an independent variable, age.
- If we fit a curve of age against salary, can we accurately use this curve to predict the age of people in the data taken?
- We know this model will fail because it is too simple. This is what is referred to as model having a high bias or underfitting.

Overfitting Example

- Lets say we now sample from 10 different cities asking respondents their age, sex, level of education, profession and years of experience, profession of parents, weight, height, best food among others.
- It becomes more complex. And our model can now generalize and make good prediction of our training data as long as we collect more data measurements.
- But can it make good prediction of other test set it has not seen. No, because it will always stick to the data as it has memorized the data. This is what we call overfitting.

  When a learner outputs a model that is 100% accurate on training data and 50% accurate on test data, it has overfitted.
- Therefore, we say our model has a high variance and thats what regularization wants to solve.

Regularization

- This is a linear regression problem and we know that our independent variables (age, level of education etc) will combine differently in different weights to make a prediction of the salary.
- Here now comes our regularization technique. We want to penalize or adjust each weights of the independent variables so that it makes a good prediction on test set that it has not seen before.
- In other words , we say regularization performs feature selection by shrinking the contribution of each features.
- Regularization will help select a midpoint between the first scenario of high bias and the later scenario of high variance. This ideal goal of generalization in terms of bias and variance is a low bias and a low variance which is near impossible or difficult to achieve. Hence, the need of the trade-off.
- We might have to reduce accuracy on training data from 100% to 80% and increase accuracy on unseen data from 50% to 80%. This is so-called trade-off between bias and variance.

Regularization

- Regularization adds stability to the learning algorithm by making it less sensitive to the training data and processes.
- Since we dont know and have no access to the true function that we can use to compare our estimated function with it, the best strategy would be to build a very complex model that fits the training data really well (overfitting) and regularize it so that it would have a good generalization (test) error.
- When using regularization, we try to reduce the generalization error and that may lead to increase the training error in the process which is okay because what we care about is how well the model generalizes.
- With regularization, we try to bring back the very complex model that suffers from overfitting to a good model by increasing bias and reducing variance. This builds on the assumption that complex model has large parameters and simple model has small parameters.

Regularization

Below are some methods used for regularization:

- L2 Parameter Regularization: Its also known as weight decay. This method adds L2 norm penalty to the objective function to drive the weights towards the origin. Even though this method shrinks all weights by the same proportion towards zero; however, it will never make any weight to be exactly zero.
- L1 Parameter Regularization (Lasso): It can be seen as a feature selection method because; in contrast to L2 regularization, some weights will be actually zero. It shrinks all weights by the same amount by adding L1 norm penalty to the objective function.

L2 Parameter Regularization

- The objective function (binary cross-entropy) would then change from:

$$J = -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}\log\left(a^{[L](i)}\right) + (1-y^{(i)})\log\left(1-a^{[L](i)}\right)\right)$$

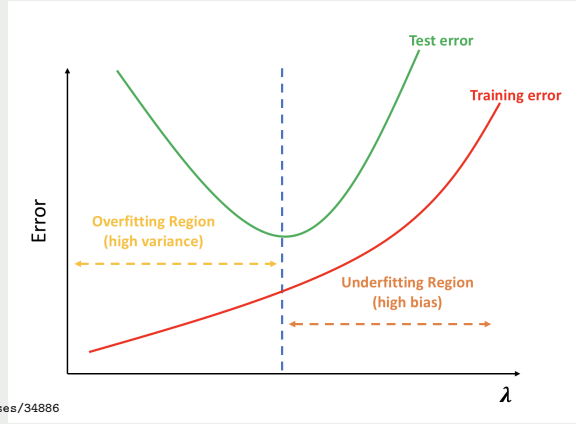
- To:

$$\begin{aligned} &J_{\text{regularized}} \\ =&J + \frac{1}{m}\frac{\lambda}{2}\sum_{l=1}^{L}\sum_{i=1}^{n^l}\sum_{j=1}^{n^{l-1}}(W_{j,i}^{[l]})^2 \end{aligned}$$

- Overall we cost function equals to the sum of the binary cross-entropy cost and the L2 regularization cost.

L2 Parameter Regularization

- λ is the regularized hyperparameter. As λ increases, the bias increases (and the model becomes less flexible) with the following extreme cases
- $\lambda = 0$, no regularization.
- $\lambda \to \infty$, model becomes very simple where all weights are essentially zero. In the case of regression, we would end-up with the intercept only equal to average of the target variable.



Why Regularization Reduce Over-fitting?

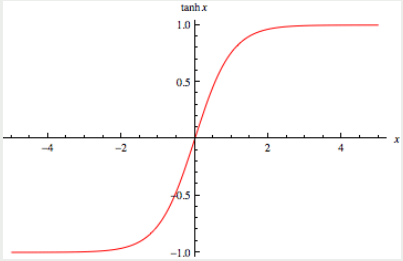
- Let us look at the cost function with regularization:

$$J_{\text{regularized}} = J + \frac{1}{m}\frac{\lambda}{2}\sum_{l=1}^{L}\sum_{i=1}^{n^l}\sum_{j=1}^{n^{l-1}}(W_{j,i}^{[l]})^2$$

- When λ is relatively large, we want to minimize all the weighting factors $\left\{\left(W_{j,i}^{[l]}\right)^2\right\}$ as much as we can. This will enforce a lot of weighting factors close to zeros.
- A lot of zero weighting factors means the developing neural network will be much simpler than the original design since some neurons are reduced to zeros. A simplified model can avoid the over-fitting case in a sense.

Why Regularization Reduce Over-fitting?

- Let us look at the activation function with each neuron, for example:



- The input of the activation function is the summation of weighted previous neurons. We know the regularization will enforce all the weighting factors small. Small $W_{j,i}^{[l]}$ will make the input close to zero.
- From above plot, we know, when input is small, the output vs. input is almost linear relationship, i.e., the activation is close to a linear transform.
- A lot of close to linear transform will make the neural networks close an under-fitting linear model. This is why regularization can reduce over-fitting.