Overfitting Problem in Machine Learning and Some Methods

Authors: Harry Hu

Hello, this blog post serves a supplement to our course material in CS514. In class, we talked about the machine learning, and some learning models like support vector machine (SVM). In this case, we want to talk about one of the biggest problems in machine learning, which is overfitting. Also, we include 2 different but relevant solutions to the overfitting problem in linear regression, which are ridge regression and LASSO regression.

Overfitting

Overfitting refers to a model that models the training data too well. Overfitting happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the model's ability to generalize.

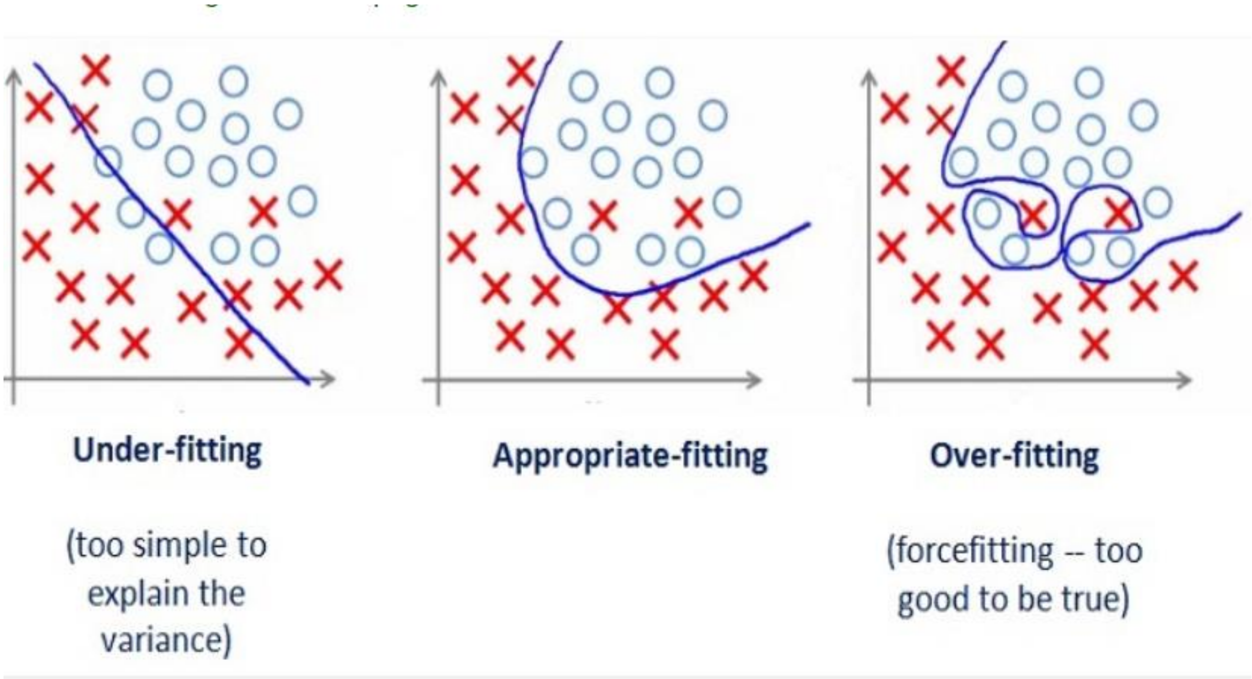
Under-fitting
(too simple to explain the variance)
Appropriate-fitting
Over-fitting
(forcefitting -- too good to be true)

Fig.1

Fig.1 shows a good example of overfitting. The model is too precise for the training set. If we have a circle in the test set near the top-right corner, it will be predicted as a cross which influence the accuracy of prediction.

How to diminish overfitting

One of the ways of avoiding overfitting is using cross validation, that helps in estimating the error over test set, and in deciding what parameters work best for your model.
The other way is using regularization. This is a form of regression, that constrains/ regularizes or shrinks the coefficient estimates towards zero. In other words, this technique discourages learning a more complex or flexible model, so as to avoid the risk of overfitting.

$$\mathrm{RSS} = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 .$$

Fig.2

The fitting procedure involves a loss function, known as residual sum of squares or RSS. The coefficients are chosen, such that they minimize this loss function. The formula in Fig.2 will adjust the coefficients based on your training data. If there is noise in the training data, then the estimated coefficients won't generalize well to the future data. This is where regularization comes in and shrinks or regularizes these learned estimates towards zero.

Ridge Regression and L2 Penalty

ridge regression performs 'L2 regularization', i.e. it adds a factor of sum of squares of coefficients in the optimization objective. Thus, ridge regression optimizes the following:

$$\sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = \mathrm{RSS} + \lambda \sum_{j=1}^{p} \beta_j^2$$

Objective = RSS + lambda * (sum of square of coefficients)
Here, lambda is the parameter which balances the amount of emphasis given to minimizing RSS vs minimizing sum of square of coefficients. lambda can take various values:

lambda = 0:

The objective becomes same as simple linear regression.
We'll get the same coefficients as simple linear regression.
lambda = ∞:
The coefficients will be zero. Why? Because of infinite weightage on square of coefficients, anything less than zero will make the objective infinite.
0 < lambda < ∞:
The magnitude of α will decide the weightage given to different parts of objective.
The coefficients will be somewhere between 0 and ones for simple linear regression.

LASSO Regression and L1 Penalty

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| \;= \mathrm{RSS} + \lambda\sum_{j=1}^{p}|\beta_j|.$$

Lasso is another variation, in which the above function is minimized. Its clear that this variation differs from ridge regression only in penalizing the high coefficients. It uses |βj|(modulus)instead of squares of β, as its penalty. In statistics, this is known as the L1 norm.

Comparison and Conclusion

The key difference between these techniques is that Lasso shrinks the less important feature's coefficient to zero thus, removing some feature altogether. So, this works well for feature selection in case we have a huge number of features.

Traditional methods like cross-validation, stepwise regression to handle overfitting and perform feature selection work well with a small set of features but these techniques are a great alternative when we are dealing with a large set of features.