Solving Non-Linear Problem in Linear Way – A Data Science Perspective

Ankur Tutlani
Microsoft India (R & D) Pvt. Ltd., Hyderabad, Telangana, India

Abstract— In this paper, we discuss about the tradeoff between using non-linear regression algorithms and linear regression algorithm for non-linear problems. This tradeoff arises because non-linear regression algorithms are difficult to explain and implement in the production environment but provide better accuracy. On the other hand, linear regression algorithm is easy to explain to the business and implement in the production environment but has relatively less accuracy in case of non-linear regression problems. The paper starts with defining non-linear regression problem and proceeds with explaining how to solve non-linear regression problem using linear regression by transforming the feature space. The paper concludes with illustrating some of the methods which can help achieve better accuracy from applying linear regression algorithm to the non-linear regression problem.

Key words: Data Mining, Machine Learning, Predictive Analytics, Linear Regression, Feature Engineering

I. INTRODUCTION

We start with explaining what constitutes a regression problem. In a regression problem, dependent variable is continuous and independent variable can be continuous/ categorical. Examples include, predicting movie theater attendance, demand for air tickets, price of a house etc. A regression problem can be linear or non-linear. It is determined by the complexity of its feature space and its interactions with the dependent variable. Feature space include all the independent variables that are being created to explain the pattern in the dependent variable. Here it needs to be noted that non-linearity implies non-linearity in the regression parameter and not in the predictor variable. For example,

$$Y = \alpha + \beta^2 X + \epsilon \qquad (1.1)$$

$$Y = \alpha + \beta X^2 + \epsilon \qquad (1.2)$$

The first equation is an example of non-linear problem, while the second equation shows linearity. In simple terms, to judge if a regression problem is linear or nonlinear we can look at the scatterplot pattern between the dependent variable and independent variable. If the plot shows linear or curvilinear pattern as in Fig. 1 below, it means it's still a linear problem. The left most part of the figure plots y with respect to x, the second plot is with respect to x^2 and the right most is with respect to x^3. All the three plots depict linear regression problems even though it includes non-linear predictor variables like x^2, x^3.

In technical terms, y is linearly related to x if the rate of change of y with respect to x (i.e. dy/dx) is independent of the value of x. For example, equation (1) above is non-linear in the parameter. By solving for α and β values using the least squares method, we get three values of β that satisfy the first order condition of optimization, namely β = 0 or Cov(x,y)/ Var(x) or - Cov(x,y)/Var(x). Here Cov(x,y) is covariance between x and y and Var(x) is variance of x. This shows non-linearity behavior. Another example of non-linear pattern in parameters are following:

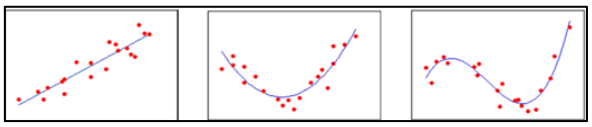

Fig. 1: Linear, Quadratic and Cubic pattern

$$Y = x / (\alpha + \beta x)$$

$$Y = \sqrt{\alpha + \beta x}$$

There are different algorithms to solve regression problem. In case of non-linear problems, usually random forest, gradient boosting algorithms give better accuracy. Here accuracy can be measured by adjusted R-squared value, Mean Squared Error (MSE) or Mean Absolute Error (MAE). The output of these algorithms is difficult to explain to business and is sometimes difficult to implement in the production environment. For example, gradient boosting generates an ensemble tree output which is hard to explain in simple terms. Fig. 2 below shows an illustrative gradient boosting output snapshot by using GBT Regression Model module in Apache Spark. As can be seen from the figure, this GB output is an ensemble of 10 individual decision tree models. The value of dependent variable 'Predict' varies depending upon feature's own value and other features' values. And the final output value will be a weighted average of individual decision tree models.

On the other hand, linear regression output is easy to implement and easily interpretable to business. For example, in the below equation predicted Y (Y*) is given by:

$$Y^* = 12 + 2X_1 - 3X_2$$

Y* is the predicted Y which comes from applying linear regression algorithm. Here it is easy to explain that with a one unit change in X_1, Y* increases by 2 units and with a one unit change in X_2, Y* decreases by 3 units.

A regression problem being a linear or non-linear is usually determined by the data availability and the feature space created. Sometimes, a problem is non-linear to begin with but with addition of new features or transformation of existing features or a combination of both will make the regression problem linear. The question arises why it is required to convert or transform non-linear problem into linear problem?

```
GBTRegressionModel (uid=GBTRegressor_4d8b901d8165b91b1e98) with 10 trees
 Tree 0 (weight 1.0):
   If (feature 4 <= 326.0)
    If (feature 1 <= 31.0)
     Predict: 182.0
    Else (feature 1 > 31.0)
     If (feature 3 <= 4.0)
      If (feature 0 <= 8.0)
       If (feature 0 <= 0.0)
        Predict: 80.0
       Else (feature 0 > 0.0)
        Predict: 87.5
      Else (feature 0 > 8.0)
       If (feature 2 <= 186.0)
        Predict: 81.0
       Else (feature 2 > 186.0)
        Predict: 41.0
     Else (feature 3 > 4.0)
      If (feature 1 <= 35.0)
       Predict: 55.0
      Else (feature 1 > 35.0)
       If (feature 0 <= 15.0)
        Predict: 157.0
       Else (feature 0 > 15.0)
        Predict: 87.0
```

Fig. 2: Illustrative Gradient Boosting Output

All rights reserved by www.ijsrd.com

This is because algorithms like gradient boosting give better performance on non-linear regression problems but are difficult to explain while algorithms like linear regression are easy to explain but give relatively poor performance.

The next section explains some of the techniques or methods which can be deployed to make the transition from non-linear problem to a possibly linear problem. To begin with, we need to consider if we have enough features capturing all possible distinct dimensions of a problem. For example, if we are dealing with the problem of movie theater attendance we should have features on movie titles, genre, cast, theater type, competitor theatres in the vicinity and its type, seat type having recliners or not, demographics etc. Once, we have enough diversity in the feature space, we can perform different transformation on these features to make the regression problem linear in the transformed feature space.

II. TRANSITION FROM NON-LINEARITY TO LINEARITY

There are different ways through which it can be achieved. We have listed out four methods below from the data science perspective which can help in making this transition.

A. Transforming Dependent/Independent variables

The different transformations are shown below in Table 1. First column Method is a name given to the transformation. The next column tells what transformations are applied on dependent/ or independent variable. Regression equation column shows the transformed equation. And last column shows how to calculate the predicted value from the regression equation.

The regression equations given in Table 1 are to be compared with the standard linear regression equation of the form $y = \beta_0 + \beta_1 x$. In some of the transformations shown in the table, the interpretation of β_1 coefficient also changes e.g. in log-log model it shows the percentage change in y with respect to 1 percent change in x. The log transformations shown above assume common log with the base 10 but it can be easily generalizable to natural log as well. The box-cox transformation is particularly helpful when there is heteroscedasticity present in the data. It requires additional parameter, λ which require to be tuned to attain normality in the data.

Method	Transformation	Regression Equation	Predicted Value(y*)
Exponential model	Dep var = log(y)	$\log(y) = \beta_0 + \beta_1 x$	$y^* = 10^{(\beta 0 + \beta 1 x)}$
Quadratic model	Dep var = sqrt(y)	$\text{sqrt}(y) = \beta_0 + \beta_1 x$	$y^* = (\beta_0 + \beta_1 x)^2$
Reciprocal model	Dep var = 1/y	$1/y = \beta_0 + \beta_1 x$	$y^* = 1/(\beta_0 + \beta_1 x)$
Log model	Ind var =log(x)	$y = \beta_0 + \beta_1 \log(x)$	$y^* = \beta_0 + \beta_1 \log(x)$
Log-log model	Dep var = log(y) Ind var =log(x)	$\log(y) = \beta_0 + \beta_1 \log(x)$	$y^* = 10^{(\beta 0 + \beta 1 \log(x))}$
Box-Cox model	Dep var = $(y^\lambda - 1)/\lambda$	$(y^\lambda - 1)/\lambda = \beta_0 + \beta_1 x$	$y^* = (\lambda(\beta_0 + \beta_1 x) + 1)^{1/\lambda}$

Table 1: Transformation of Dependent/ Independent Variables

The question arises which one of these transformations should be used in which contexts. Following steps are useful to determine this:

1) First perform standard linear regression of the form $y = \beta_0 + \beta_1 x$ and check R-squared value. We are assuming R-squared as a measure of accuracy. We can use other measures like MAE or MSE also.
2) From the regression performed in step 1, check the residuals pattern versus predicted value of y (y*), also called residual plot.
3) If residual pattern is random, transformation of either dependent or independent variable may not improve accuracy.
4) If the pattern is non-random, start with any one of the transformation from above table.
5) Check the residuals pattern again on the regression performed on transformed variables. If it is random, proceed with this.
6) If the residual pattern is still non-random, try with some other transformation.

In Fig. 3 below, different residual patterns are shown. The first plot (extreme left) does not have any pattern and hence it can be called a random residual pattern while the other plots have non- random patterns. It needs to be noted that one should also check for the assumptions on which linear regression algorithm is based. The non-random pattern observed in the residual plots may be attributed to the violation of some of the assumptions. For example, the non-random residual pattern shown in Fig. 3 may be indicative of high bias wherein an important predictor or feature variable is not being incorporated in the feature space.

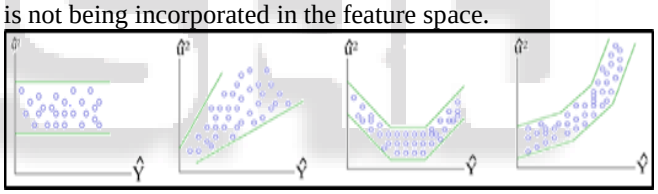

Fig. 3: Residual pattern between squared residuals values ($u\hat{}2$) and predicted value of Y ($Y\hat{}$)

In this case, the better approach is to add more features into the feature space in addition to trying out different transformations on existing features. Some of the transformations mentioned above actually help in avoiding the violation of few of these assumptions. The case in point is the box-cox transformation which is used to make the data homoscedastic when there is a presence of heteroscedasticity in the data.

B. Binning

The other method to try is to bin continuous variables. In simple terms, it means converting a continuous variable into a categorical variable. There are different ways through which this can be achieved. One option is to perform quantile distribution of the variable and create a categorical variable having values as the cut-off points from the quantile distribution. For example, variable x has the following quantile distribution:

Min	Q1	Q2	Q3	Max
-45	12	23	34	58

Table 2: Binning

In above distribution, Q1, Q2 and Q3 are the first, second and third quartile respectively. With this distribution, we can have another column x* which is categorical in nature.

All rights reserved by *www.ijsrd.com*

x is the original raw variable (continuous) and x* is the categorical variable having 4 levels.

x	x*
7	<12
13	≥ 12, <23
30	≥ 23, < 34
39	≥ 34

Table 3: Binning

In another way, values can be bucked based upon business understanding of the variable, for instance variable like Age of a customer can be bucked into 0-20, 20-30, 30-50, and > 50 to indicate child, teen, adult and senior categorization which is subjective and varies depending upon the context.

C. Splines

The idea behind regression spline is to do a scatterplot and identify kink in the graph of dependent variable and independent variable. In Fig. 4 below there is a kink observed at x = 13. The key to introduce this type of information in the predictive modeling exercise is to introduce two separate variables max (0, x-13) and max (0, 13-x) instead of standalone x. 'max' is indicating max function.

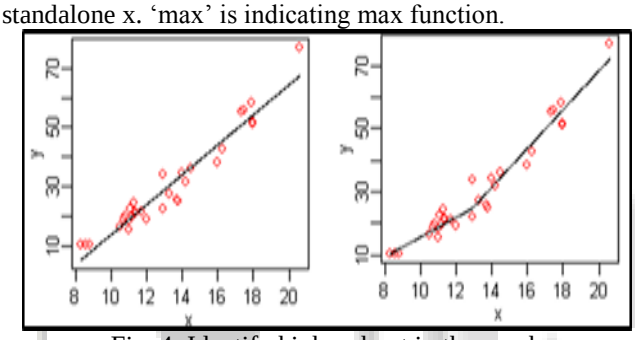

Fig. 4: Identify kink or knot in the graph

By doing this, we should get two slope coefficients while performing linear regression of y on x as in Fig. 4 (right) as against just one slope coefficient (Fig. 4 left).

D. Ablation Features

The intent of ablation features is to understand the trend at the individual record level when compared with the overall data. In ablation features, for the existing set of continuous variables, we create another set of these variables which contain values as deviation from the total. This is illustrated below with a small example:

x	x* = 100 - x
10	90
15	85
20	80
25	75
30	70

Table 4: Ablation Features

Suppose x is a feature representing number of alpha characters in a string. x* is another feature showing the deviation from N where N is the number of alpha characters in all the five records/ strings taken together. For simplicity, we assume there are total 5 records in the training data. This means N = 10+15+20+25+30 = 100. By replicating this to all continuous variables, we can expand the feature space and include x* variables in addition to the original x variables.

III. CONCLUSION

The paper illustrated some of the methods or transformation techniques that can be used when we are dealing with non-linear regression problem and at the same time we want a simple and easy to explain solution. The four methods illustrated may help to increase accuracy of linear regression algorithm either by augmenting the feature space or modifying the existing feature space and/or transforming the dependent variable. It needs to be noted that one or a combination of different methods may work in a problem. For some of the features, log transformation may work but for some of the features incorporating splines variables produce better accuracy in the same problem. So, there is some trial and error involved in this process. However, this can be reduced to a certain extent if it is supplemented with background knowledge and business understanding.

All rights reserved by *www.ijsrd.com*