Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

Nearest-Neighborhood Linear Regression in an Application with Software Effort Estimation

Luciana Q. Leal, Roberta A.A. Fagundes,
Renata M.C.R. de Souza and Hermano P. Moura
Centro de Informática
Universidade Federal de Pernambuco
Recife - PE, Brazil
{lql,raaf,rmcrs,hermano}@cin.ufpe.br

Cristine M.G. Gusmão
Departamento de Sistemas de Computação,
Escola Politécnica de Engenharia
Universidade Estadual de Pernambuco
Recife - PE, Brazil
cristine@dsc.upe.br

***Abstract*—This paper discusses nearest-neighborhood linear regression methods in a statistical view of learning and present an application of these models to software project effort estimation. The usefulness of the models is highlighted through experiments with a well-known NASA software project data set. A comparative study with global regression methods such as bagging predictors, support vector regression, radial basis functions neural networks is also introduced.**

***Index Terms*—local linear regression, kernel function, LOESS, LOWESS, software effort estimation.**

I. Introduction

In most learning methods [1] a single global model is used to fit all of the training data. Local learning methods fit the training data set only in region around the location of the query (test point). Examples of types of local linear models included nearest neighborhood and weighted nearest neighborhood. Nearest neighborhood local linear models fit a surface to nearby points using linear regression. Weighted nearest neighborhood models fits a surface to nearby points using a distance weighted regression.

Linear regression [2] is widely used in statistical estimation. The benefits of a linear model are its simplicity and ease of use, while its major drawback is its high model bias: if the underlying function is not well approximated by an linear function, then linear regression produces poor results. Local linear regression exploits the fact that, over a small enough subset of the domain, any sufficiently nice function can be well approximated by a linear function.

In software engineering tools and methodologies are primarily based on data from past projects. Researchers and engineers have studied project data and determined equations and formulas that best matched with the existing data points. Some of the formulas are extremely simple, others are complex. None are totally accurate with all of the past projects, nor should you expect them to be with their predictions for future projects.

To estimate software effort some authors have fitted regression models using as input variables a number of features related to software development, such as the software size and the methodology [3], [4], [5], [6] and [7]. The output is the total effort in man-months considering both programming and management activities. The regression model is trained with a number of past projects and is subsequently employed to predict the effort of novel projects. Examples of software effort estimation using global learning methods include the radial basis function (RBF) neural networks [5], support vector regression [6] and bagging predictions [7].

This paper discusses local models (nearest neighborhood and weighted nearest neighborhood) over a statistical view of learning in an application with software effort estimation. A NASA software project data base is considered. The idea is to show the usefulness of the local regression methods as one alternative to the problem of software engineering estimation models, particulary effort estimation. In addition, the local methods are compared with global methods that used the NASA data base to software effort estimation. The prediction accuracy furnished by the local and global regression methods was assessed by the mean magnitude of relative errors (MMRE) and prediction rate PRED(25).

The structure of the paper is as follows: Section 2 present local linear regression methods. Section 3 describes the importance of estimation in software engineering and it shows the NASA data base. Section 4 presents a experimental evaluation of the local regression methods using the NASA data set. The local methods are compared with support vector regression, radial basis function neural networks and bagging regression methods. Finally, Section 5 gives the concluding remarks.

II. Local Linear Regression Methods

This section introduces two local linear regression methods. These methods construct an local approximation of a linear function that model the relation between a set of predictor variables and response variable at a query point. Non-weighted and weighted error criterions by obtaining this approximation are considered. Here the goal is to apply local regression models using a data set to predict the total effort in future software projects.

Let $\Omega = \{(\mathbf{x}_i), y_i\}$ $(i = 1, \ldots, n)$ be a training data set. Each pattern i is described by a vector $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ representing quantitative values of p predictor variables $X_1, \ldots, X_p$ and a quantitative response value y_i representing the value of a dependent quantitative variable Y.

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

A. Nearest-Neighborhood Linear Regression (NNLR)

Let $\mathbf{x}_q = (x_{q1}, \ldots, x_{qp})$ be the description of a query point q, K be a size of neighborhood surrounding $\mathbf{x}_q$ and D a set of K nearest neighbors of q. Consider a matrix $\mathbf{X}_q = \{\mathbf{x}_k\}$ representing K nearest neighbors to the point $\mathbf{x}_q$ and the corresponding vector $\mathbf{y}_q = (y_1, \ldots, y_K)$ of $\mathbf{X}_q$ representing K response values.

An linear function that model the relation between a subset of predictor variables and response variable over D is given of the form

$$y = f(\mathbf{x}) + \epsilon \tag{1}$$

where $f(\mathbf{x}) = w_0 + w_1 x_1 + \ldots + w_p x_p$ is unknown and e is a error term, representing random errors in the observations or variability from sources not included in $\mathbf{x}_q$.

Assuming that the errors are independent and identically distributed with mean 0 and constant variance $var(\epsilon) = \sigma^2$. An approximation of the function $f(\mathbf{x})$ is of the form

$$\hat{f}(\mathbf{x}) = \hat{w}_0 + \hat{w}_1 x_1 + \ldots + \hat{w}_p x_p \tag{2}$$

The parameters $w_1, \ldots, w_p$ are estimated minimizing the squared error over D

$$E = \sum_{x \in D} (f(\mathbf{x}) - \hat{f}(\mathbf{x}))^2 \tag{3}$$

by the least square method that does not require any probabilistic hypothesis on the variable Y.

Note that an approximation of the function $f(\mathbf{x})$ at point $\mathbf{x}_q$ is of the form

$$\hat{f}(\mathbf{x}_q) = \hat{w}_0 + \hat{w}_1 x_{q1} + \ldots + \hat{w}_p x_{qp} \tag{4}$$

B. Weighted Nearest-Neighborhood Regression (WNNLR)

The essential idea of this method is that in estimating $f(\mathbf{x}_q)$ it is desirable to give greater weight to observations that are close to the focal $\mathbf{x}_q$.

Considering $d(\mathbf{x}, \mathbf{x}_q)$ the distance between a observation $\mathbf{x}$ of the data set D and the query $\mathbf{x}_q$. To weight the neighborhood of $\mathbf{x}_q$ it is need a kernel function $K(d(\mathbf{x}, \mathbf{x}_q))$ that attaches greatest weight to observations that are close to the focal $\mathbf{x}_q$, and then falls off symmetrically and smoothly as $|d|$ grows. Given these characteristics, the specific choice of a kernel function is not critical.

A popular kernel function is the gaussian function given by

$$K(d(\mathbf{x}, \mathbf{x}_q)) = \frac{1}{\sqrt{2\pi}} e^{\frac{-d(\mathbf{x}, \mathbf{x}_q)}{2h}} \tag{5}$$

where $d(\mathbf{x}, \mathbf{x}_q)$ is square Euclidian distance between $\mathbf{x}$ and the location of interest $\mathbf{x}_q$. Here, the bandwidth h is the standard deviation of a normal distribution centered at $\mathbf{x}_q$.

In this method, an approximation of the linear function $f(\mathbf{x})$ at point $\mathbf{x}_q$ is of the also form

$$\hat{f}(\mathbf{x}_q) = \hat{w}_q^0 + \hat{w}_1 x_q^1 + \ldots + \hat{w}_p x_q^p \tag{6}$$

The parameters $w_1, \ldots, w_p$ are estimated minimizing the squared error over D

$$E = \sum_{x \in D} (f(\mathbf{x}) - \hat{f}(\mathbf{x}))^2 K(d(\mathbf{x}, \mathbf{x}_q)) \tag{7}$$

by the weighted least square method.

III. Estimation in Software Engineering

An activity which has been realized in software engineering is the development of prediction models. This activity aims to help the software development process and the software project management. Studies in this area employ experimental data to build these models, and a logical approach for dealing with these kind of data is to develop empirical models which provide an approximate response over a specific range of relevant variables [5].

Estimations are the basis of planning in software engineering, and particulary they are useful in schedule and budget development. It is necessary to estimate the size of the software, the project costs and the time to execute the whole project. Researches in the field have built up models relating project size, which is the factor that affects the effort of the project the most. Size and productivity estimations can be obtained from effort estimation. Managers can derivate cost and time estimations using effort. However, good results only can be obtained when considering any previous projects.

Software effort is probably one of the most analyzed response variables in recent years in the process of project management. The determination of the estimated value for this variable when initiating software projects allows us to plan adequately the activities. Estimating the effort with a high grade of reliability is a problem which has not yet been solved and the project manager has handled with it since the beginning.

In order to perform an application with software effort estimation using nearest-neighbor linear regression, a well known NASA data set presented in [3] is considered in this work. This data base consist 18 projects. Each project is described by two predictor variables: Developed Lines (DL) and Methodology (ME) and a response variable Effort. Table I shows the NASA data set.

The dependent variable effort was defined to be measured from beginning of the design phase through acceptance testing and to include programming, management and support hours [3]. It is given in man-months. Developed Lines (DL) and Methodology (ME) are the predictor variables of the model. DL is the number of developed lines of source code with comments plus 20% of re-used lines [3]. It's given KLOC (thousands of lines of code). ME is the methodology applied in the development of each NASA software project. Some factors used by [3] to define ME were: (i) Formal documentations, (ii) Code reading, (iii) Formal test plans, (iv) Unit development folders, and (v) Formal training.

TABLE I
NASA SOFTWARE PROJECT DATA [6]

Project Number	DL	ME	Effort
1	90.2	30	115.8
2	46.2	20	96.0
3	46.5	19	79.0
4	54.5	20	90.8
5	31.1	35	39.6
6	67.5	29	98.4
7	12.8	26	18.9
8	10.5	34	10.3
9	21.5	31	28.5
10	3.1	26	7.0
11	4.2	19	9.0
12	7.8	31	7.3
13	2.1	28	5.0
14	5.0	29	8.4
15	78.6	35	98.7
16	9.7	27	15.6
17	12.5	27	23.9
18	100.8	34	138.3

IV. EXPERIMENTAL EVALUATION

This section present an experimental evaluation of the discussed models in this work using the NASA data set. It starts describing the prediction accuracy measures used to evaluate the model performance. In the following, an prior data analysis is presented. After that, results of prediction accuracy furnished by the models are showed and a comparison with others methods introduced in the literature is discussed. These measures are estimated by the leave-on-out method.

A. Prediction accuracy

In regression models, an important part of the development model is to know how accurate are the predictions. Here, the the prediction accuracy was measured by mean magnitude of relative errors (MMRE) and PRED(25). The MMRE is defined as:

$$MMRE = \frac{1}{K}\sum_{j=1}^{K}\frac{|y_j - \hat{y_j}|}{y_j} \tag{8}$$

where K is the number of projects, y_j is the actual effort and $\hat{y_j}$ is the predicted effort. Lower values of MMRE result from good regression models, and a common criterion for accepting a model as good is MMRE $\leq$ 0.25 [9].

The PRED(l) measure means prediction rate for level (l). Consider the relative error for the observation j of data set indexed by $1, \ldots, K$ as follow

$$MRE_j = \frac{|y_j - \hat{y_j}|}{y_j} \tag{9}$$

Let r the number of $MMRE_j \leq l$, the PRED(l) is given by

$$PRED(l) = \frac{r}{K} \tag{10}$$

For exemple, PRED(25) is the percentage of predictions that fall within 25% of the actual value. According to [8], a criterion for accepting a model as good is PRED $\geq$ 0.75. With this result at least 75% of the estimates are within the 25% range of the actual values.

MMRE is fairly conservative with bias against overestimates while PRED(25) will identify those prediction systems that are generally accurate but occasionally wildly inaccurate [9]. For this reason, in this paper we considered MMRE as the main performance measure.

B. Prior Data Analysis

We executed a prior analysis of the data to know if the variables are correlated. We have calculated the VIFs for independent variables. We obtained a VIF value equals to 1.0413 for both DL and ME variables. This value means that DL an ME are minimally correlated, representing the good quality of the prediction model. The Figure 1 confirms the minimum correlation between the independent variables.

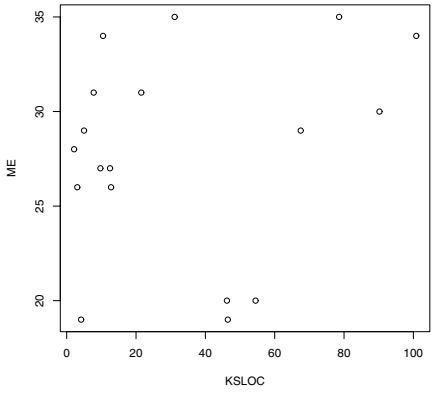

Fig. 1. Scatterplot between variables DL and ME.

Another way to show the variable correlation is calculating Pearson's Correlation (r), the most common measure of correlation. For DL and ME, we have obtained $\rho = 0.1993$. Small values of ρ means low degree of relationship between two variables and the value obtained was expected because the calculated values of VIFs and the scatterplot showed in the Fig.1.

C. Effort Estimation

The NASA data base was applied to the NNLR and WNNLR methods with neighborhood parameter $K = 9$ and this section presents MMRE and PRED(25) values that were estimated by the leave-on-out method. For WNNLR method, it was used the Gaussian kernel with h equal to the standard deviation of the DL variable.

Table II presents the average and standard deviation for the estimated effort considering the NNLR and WNNLR models in which only the DL variable is used as predictor variable. From the results in this table, we can observe that the models furnished the same values for average PRED(25) but the

WNNLR model furnished a standard deviation smaller than that of the NNLR model. With respect to average MMRE, the models furnished similar results.

TABLE II
AVERAGE AND STANDARD DEVIATION FOR MMRE AND PRED(25) (IN%) AND NNLR AND WNNLR MODELS

Measure	NNLR		WNNLR	
	Average	Standard Deviation	Average	Standard Deviation
MMRE	0.1443	0.1363	0.1418	0.1359
PRED(25) (in %)	88.89	0.3234	88.89	0.3234

Figure 2 illustrates the prediction effort values for the WNNLR model.

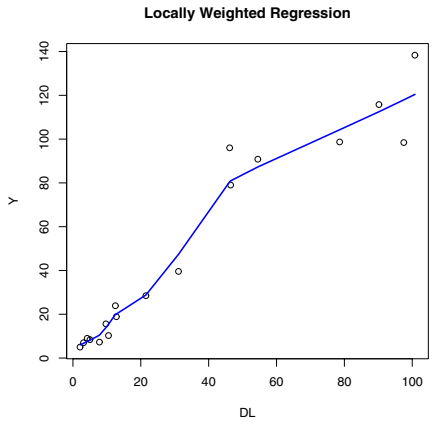

Fig. 2. Estimated effort for the WNNLR model

Table III presents the average and standard deviation for the estimated effort considering the NNLR and WNNLR models in which the DL and ME variables are used as predictor variables. From the results in this table for PRED(25), we can observe that both models furnished 100% as average value. With respect to MMRE, the average values are similar.

TABLE III
AVERAGE AND STANDARD DEVIATION FOR MMRE AND PRED(25) (IN%) AND NNLR AND WNNLR MODELS

Measure	NNLR		WNNLR	
	Average	Standard Deviation	Average	Standard Deviation
MMRE	0.0868	0.0637	0.0845	0.0615
PRED(25) (in %)	100.00	0.000	100.00	0.0000

Table IV shows actual and estimated efforts obtained by the the NNLR and WNNLR models.

TABLE IV
COMPARISON BETWEEN ACTUAL AND ESTIMATED EFFORTS FOR THE NNLR AND WNNLR MODELS

Project	Actual Effort	Predicted Effort	
		NNLR model	WNNLR model
1	115.8	104.101	103.105
2	96.0	83.048	83.598
3	79.0	83.743	83.419
4	90.8	93.321	89.844
5	39.6	38.627	38.873
6	98.4	111.500	110.154
7	18.9	22.282	21.313
8	10.3	9.902	10.106
9	28.5	29.000	28.907
10	7.0	7.390	7.461
11	9.0	8.934	8.397
12	7.3	8.945	8.943
13	5.0	4.357	4.335
14	8.4	7.314	7.235
15	98.7	104.795	105.046
16	15.6	16.121	16.072
17	23.9	20.063	19.955
18	138.3	132.132	134.443

To conclude, we can say that the WNNLR model was as good as the NNLR one in terms of MMRE and PRED(25). Because the WLLNR model furnished MMRE results smaller than those for LLNR model, the WNNLR model is considered as the best option in this work.

D. Comparing with related works

Tables V and VI show estimated values for PRED(25) and MMRE regarding simples and multiple regressions, respectively. The goal is to compare the WNNLR model with the bagging method, support vector regression (SVR), multilayer perceptron neural networks (MLP) and radial basis functions neural networks (RBF) introduced in, [6], [5] and [4] respectively. The values in these tables point out that WNNLR model considered in this paper is very superior to other methods.

TABLE V
COMPARISON OF MODELS USING SIMPLE REGRESSION

Method	MMRE	PRED(25) (in%)
Bagging	0.1776	83.33
SVR Linear	0.1790	88.89
MLP	0.1970	83.33
RBF	0.1870	72.22
WNNLR	0.1418	88.89

TABLE VI
COMPARISON OF MODELS USING MULTIPLE REGRESSION

Method	MMRE	PRED(25) (in%)
Bagging	0.1639	88.89
SVR Linear	0.1650	88.89
MLP	0.1942	83.33
RBF	0.1907	72.22
WNNLR	0.0868	100.00

V. Conclusion

In this paper, we have investigated the use of nearest-neighborhood linear regression for estimation in software engineering. We applied the NASA data set of software projects to theses methods. The accuracy of the results furnished by these regression methods were assessed by the average relative error and the prediction rate that were estimated by the leave-on-out method.

The methods discussed in this work were compared with global methods such as radial basis function neural networks [4], support vector regression [5], bagging predictors [6] according to the average relative error and the prediction rate. Concerning the presented results, the nearest-neighborhood linear regression methods clearly outperformed the methods that use global otimization.

Acknowledgment

The authors would like to thank CNPq (Brazilian Agency) for its financial support.

References

[1] T. Mitchell, *Machine learning*, McGraw-Hill, 1997.

[2] D. Montgomery, E. Peck, and G. Vining. *"Introdution to Linear Regression Analysis"* Fourth Edition, Wiley Series in Probability and Statistics, 2006.

[3] J.W. Bailey, and V.R. Basili, "A meta model for software development resource expenditure", *Proceedings of the Fifth Intenational Conference on Software Engineering*, San Diego, California, USA, 1981, pp. 107-116

[4] M. Shepperd, C. Schofield, "Estimating software project effort using analogies.", *IEEE Trans. Software Eng. 23 (11)*, 1997. 736743.

[5] M. Shin, A. L. Goel, "Empirical data modeling in software engineering using radial basis functions". *IEEE Transactions on Software Engineering*, vol 26, no. 6, June, 2000.

[6] A. L. I. Oliveira, "Estimation of software projects effort with support vector regression". *Neurocomputing*, vol 69, no. 13-15, pp. 1749-1753, August, 2006.

[7] P. L. Braga, A. L. I. Oliveira, G. H. T. Ribeiro, and S. R. L. Meira, "Bagging predictors for estimating of software project effort". *Proceedings of International Joint Conference on Neural Networks*, Orlando, Florida, USA, 2007.

[8] C.F. Kemerer, "An Empirical Validation of Software Cost Estimation Models", *Communications of the ACM*, vol. 30, no. 5, pp. 416-429, May, 1987.

[9] J. J. Dolado. "A validation of the component-based method for software size estimation", *IEEE Transactions on Software Engineering*, vol. 25, no. 10, pp. 1006-1021, October, 2000.