A predictive analysis approach using linear regression to estimate software effort

Antogio G. L. Esteves[1], Leonardo M. Medeiros (Orientador)[1]

[1]Pós-graduação em Gerenciamento e Desenvolvimento Ágil de Software
Instituto Federal de Alagoas (IFAL) – Campus Maceió – Maceió - AL – Brasil

toni.esteves@gmail.com, leonardomelomedeiros@gmail.com

Abstract. *Making decisions with a highly uncertain level is a critical problem in the area of software engineering. Predicting software quality requires high accurate tools and high-level experience. AI-based predictive models, on the other hand, are useful tools with an accurate degree that help to make decisions learning from past data. In this study, we build a software effort estimation model to predict the effort before the project development lifecycle, using a linear regression model and also using non-parametric validation model through a Knn regression algorithm.*

1. INTRODUCTION

Software development involves a number of interrelated factors which affect development effort and productivity . The most significant activity in software engineer is the development of projects within the confined timeframe and budget. So accuracy has a vital role for software development, effort prediction estimation is one of the critical tasks required for developing software. In this work our research focus on analyzing the importance of attributes in estimating software cost as well as its correlation.

In this paper, we set out to answer two research questions related to the dataset:

1. Which the correlation of each metrics in the estimation of software effort ?
2. How accurate is the model of software effort ?

2. EFFORT ESTIMATION

When measurements embrace structure system they become more meaningful indicators called metrics. Metrics are conceived by the user and designed to reveal chosen characteristics in a reliable meaningful manner. Then these metrics are mapped to ongoing measurements, to arrive at a best fit [Pandian 2003]

One of the fundamental issues in a software project is to know, before executing it, how much effort, in working hours, it will be necessary to bring it to term. This area called effort estimation counts on some techniques that have presented interesting results over the last few years[Wazlawick 2013].

One of reasons for failed estimations is an insufficient background of information in the area of software estimation. Unfortunately, human experts are not always as good at estimating as one could hope: estimates of cost and effort in software projects are often inaccurate, with an average overrun of about 30% [Halkjelsvik and Jørgensen 2011].

Deliberate decisions regarding the particular estimation method and knowledgeable use require insight into the principles of effort estimation[Trendowicz and Jeffery 2014].

Learning-oriented models attempt to automate the estimation process by building computerised models that can learn from previous estimation experience [Boehm et al. 2000]. These models do not rely on assumptions and are capable of learning incrementally as new data are provided over time [Lee-Post et al. 1998].

2.1. RELATED WORKS

The research developed by Ayyıldız makes use of Desharnais dataset to finding the necessary attributes that affects the software effort estimation and analyzing the necessity of these attributes [Erçelebi Ayyıldız and Can Terzi 2017]. The Pearson's Correlation correlations between metrics of Desharnais dataset and software effort are analyzed and applicability of the regression analysis is examined.

To show the differences between the actual and estimated values of the dependent variable, prediction performance are evaluated using Magnitude of Relative Error (MRE), Mean Magnitude of Relative Error (MMRE), Median Magnitude of Relative Error (MdMRE), MSE (Mean Square Error) and Prediction Quality (pred(e)).

One of the most complete studies was presented by Kitchenham [Kitchenham et al. 2002]. In her study, was present a data set that enables to investigate the actual accuracy of industrial estimates and to compare those estimates with estimates produced from various function point estimation models. How- ever, the study make it clear that any models derived from the current data set are context-specific. The conclusions drawed from this study are somewhat limited, because the projects studied were undertaken by a single company. Thus, it was not expected any of the models presented in this paper to generalize automatically to other maintenance or development situations.

3. MATERIALS AND METHODS

To perform our study firstly we analyze the correlation between each attributes of Desharnais dataset and effort attribute. We apply linear regression technique to investigate relation between these attributes. After that we apply a regression based on k-nearest neighbors regressor. Lastly we evaluate our prediction performance comparing the squared error value of both algorithms .

3.1. DATASET

To perform this study we used Desharnais dataset [1] which is composed of a total of 81 projects developed by a Canadian software house in 1989. This data set includes nine numerical attributes. The eight independent attribute of this data set, namely "TeamExp", "ManagerExp", "YearEnd", "Length", "Transactions", "Entities", "PointsAdj", and "PointsNonAjust" are all considered for constructing the models. The dependent attribute "Effort" is measured in person hours.

[1] The promise repository of empirical software engineer data.

3.2. FEATURE SELECTION

To address Desharnais dataset the correlations between attributes and software effort are analyzed. The correlation between two variables is a measure of how well the variables are related.A feature is an individual measurable property of the process being observed.

The most common measure of correlation in statistics is the Pearson Correlation Pearson correlation coefficient (PCC), wich is a statistical metric that measures the strength and direction of a linear relationship between two random variables [Rodgers and Nicewander 1988]. Pearson correlation coefficient analysis produces a result between -1 and 1. Results between 0.5 and 1.0 indicate high correlation [Mehedi Hassan Onik et al. 2018]. The Pearson correlation coefficients between attributes and software efforts are given in Figure 1 for Desharnais dataset.

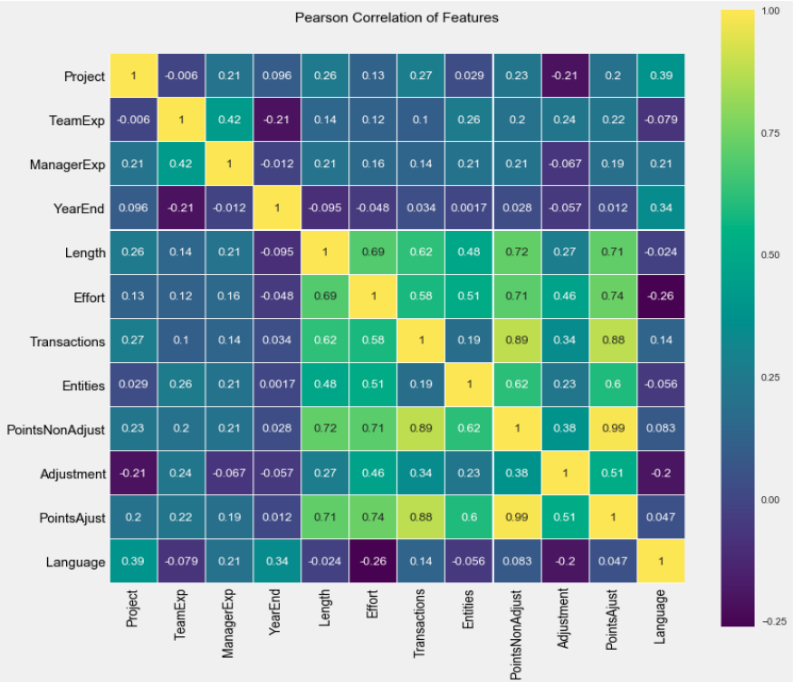


Figure 1. Pearsons Correlation to Desharnais dataset

3.3. MODELS CONSTRUCTION

In this study the following algorithms were used: *Linear Regression* and *K-Nearest Neighbors Regression*. The training of the models was carried out in Python language, along with the following libraries: Numpy, Pandas, Scikit-learn, Seaborn and Matplotlib. During the training it was necessary to estimate the values of the random state parameter, since they are not previously known.

The regression analysis aims to verify the existence of a functional relationship between a variable with one or more variables, obtaining an equation that explains the variation of the dependent variable *Y*, by the variation of the levels of the independent variables. The training of the *Linear Regression* model consists of generating a regression for the target variable *Y*. Thus a linear regression line has an equation of the form $Y = a + bX$, where *X* is the explanatory variable and *Y* is the dependent variable. The slope of the line is *b*, and *a* is the intercept (the value of *y* when $x = 0$).

Likewise the *K-Nearest Neighbor Regression* is a simple algorithm that stores all available cases and predict the numerical target based on a similarity measure and it's been used in a statistical estimation and pattern recognition as non-parametric technique classifying correctly unknown cases calculating euclidean distance between data points. In fact our choice by *K-Nearest Neighbor Regression* was motivated by the absence of a detailed explanation about how effort attribute value is calculated on Desharnais dataset.

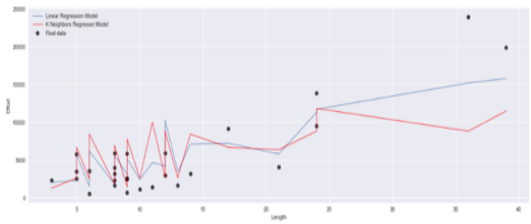
4. RESULTS

Both models generated from the training with data from the previous section will be applied to the remaining 33% of the base, previously isolated, and their performances will be evaluated in order to demonstrate how accurate the linear regression model can predict software effort estimation. Thus, we calculate respective R^2 values. Table 3 shows the coefficients reached.

Algorithm	**R^2 Score**
Linear Model Regression	0.7680074954440712
K-Nearest Neighbor Regressor	0.7379861869550943

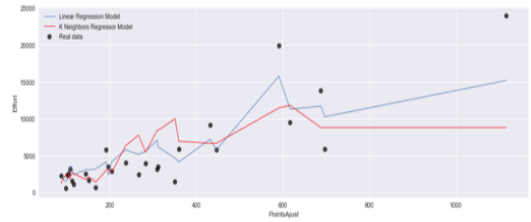
Table 1: Algorithms model results

In Figure 2 plots of the best correlated variables applied to both models are displayed.

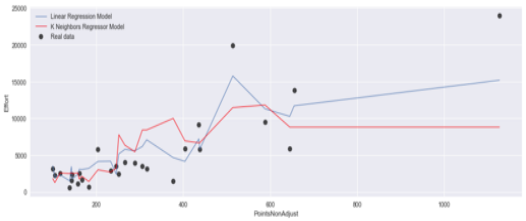


(a) *Knn* x *LR* on Length feature

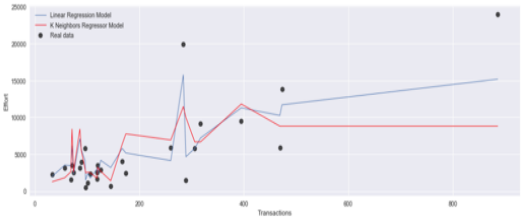
(b) *Knn* x *LR* on Entities feature



(c) *Knn* x *LR* on PointsAdjust feature



(d) *Knn* x *LR* on PointsNonAdjust feature



(e) *Knn* x *LR* on Transactions feature

Figure 2. Comparative R^2 scores from K-neighbors Regression and Linear Regression

Each feature from more correlated features is illustrated in Figure 2. The figure shows the linear model (blue line) prediction is fairly close to Knn model effort prediction (red line), predicting the numerical target based on a similarity measure.

5. CONCLUSION AND FUTURE WORKS

The contributions of this work are based on the use of two output models that seek to take advantage of the relationships between the target values of the project. These methods, together with linear regression and K-neighbors regression algorithms , resulted in predictive models capable of estimating values for the software effort estimation operations. The results of our empirical study reveal that predictive model of software effort presented by both models, could successfully predict more than 70% with less than 3% difference between them.

Our results obtained obtained a R^2 value of more than 70% and a difference of only 3% among them, indicating the feasibility of using linear regressors to predict software effort. However, to have a more concise and fair result we need to reproduce the same approach with other available algorithms.

Finally, we propose as future works the use of a larger project base in order to diversify and give greater reliability to the method. Another point to consider is to apply these models in order to compare them with the function points.

References

Boehm, B., Abts, C., and Chulani, S. (2000). Software development cost estimation approaches – a survey. *Ann. Softw. Eng.*, 10(1-4):177–205.

Erçelebi Ayyıldız, T. and Can Terzi, H. (2017). Case study on software effort estimation. 7:103–107.

Halkjelsvik, T. and Jørgensen, M. (2011). From origami to software development: A review of studies on judgment-based predictions of performance time. 138:238–71.

Kitchenham, B., Pfleeger, S. L., McColl, B., and Eagan, S. (2002). An empirical study of maintenance and development estimation accuracy. *Journal of Systems and Software*, 64(1):57 – 77.

Lee-Post, A., Cheng, C. H., and Balakrishnan, J. (1998). Software development cost estimation: Integrating neural network with cluster analysis. 34:1–9.

Mehedi Hassan Onik, M., Ahmmed Nobin, S., Ferdous Ashrafi, A., and Mohmud Chowdhury, T. (2018). Prediction of a Gene Regulatory Network from Gene Expression Profiles With Linear Regression and Pearson Correlation Coefficient. *ArXiv e-prints*.

Pandian, C. R. (2003). Software metrics: A guide to planning, analysis, and application.

Rodgers, J. and Nicewander, W. (1988). Thirteen ways to look at the correlation coefficient. *The American Statistician*, 42(1):59–66.

Trendowicz, A. and Jeffery, R. (2014). *Software Project Effort Estimation: Foundations and Best Practice Guidelines for Success*. Springer Publishing Company, Incorporated.

Wazlawick, R. (2013). *ENGENHARIA DE SOFTWARE: CONCEITOS E PRÁTICAS*. Elsevier Editora Ltda.