Evaluating Artificial Neural Networks and Traditional Approaches for Risk Analysis in Software Project Management

A case study with PERIL dataset

Keywords: Software Project, Risk Management, Risk Analysis, Support Vector Machine, MultiLayer Perceptron, Monte Carlo Simulation, Linear Regression Model

Abstract: Many software project management end in failure. Risk analysis is an essential process to support project success. There is a growing need for systematic methods to supplement expert judgment in order to increase the accuracy in the prediction of risk likelihood and impact. In this paper, we evaluated support vector machine (SVM), multilayer perceptron (MLP), a linear regression model and monte carlo simulation to perform risk analysis based on PERIL data. We have conducted a statistical experiment to determine which is a more accurate method in risk impact estimation. Our experimental results showed that artificial neural network methods proposed in this study outperformed both linear regression and monte carlo simulation.

1 INTRODUCTION

How risky are software projects? Several studies about effectiveness of software cost, scope, schedule estimation techniques; surveys from software professionals in industry; and analysis of project portfolios have been done to answer this question (Budzier and Flyvbjerg, 2013). However, there is not a consensus.

Some remarkable work support that software projects involve risky activities. Schmidt et al. (2001) have noticed that many software development projects end in failure. They showed that around twenty five percent of all software projects are canceled outright and as many as eighty percent of all software projects run over their budget, exceeding it by fifty percent in average. Bannerman (2008) states that industry surveys suggest that only a quarter of software projects succeed outright, and billions of dollars are lost annually through project failures or projects that do not deliver promised benefits. Moreover, Bannerman (2008) shows evidences that it's a global issue, impacting private and public sector organizations.

Boehm (1991) defined risk as the possibility of loss or injury. This definition can be expressed by risk exposure formula. This study takes into account Project Management Institute (2008) definition whereupon project risk is a certain event or condition that, if it occurs, has a positive or negative effect on one or more project objectives. A complementary definition proposed by Haimes (2011) is also considered, which express risk as a measure of the probability and severity of adverse effects.

Recent perceptions about risk management and the inherent challenges posed by the nature of software projects contributes to the lack of project stability from majority of software project organizations (Kwak and Stoddard, 2004). Kwak and Ibbs (2000) identified risk management as the least practiced discipline among different project management knowledge areas. The authors mention that, probably, a cause for it is that software developers and project managers perceive managing uncertainty processes and activities as extra work and expense.

A difficult task in risk analysis is to perform accurate estimates of the probability and impact associated with an unexpected outcome (Boehm, 1991). Bannerman (2008) have found a limitation on Boehm's definition - it is very difficult, in practice, to estimate the probability of many risk factors, especially in software projects. Probability and impact can only be meaningfully determined for activities that are repeated many times, under controlled circumstances. The one-off nature of many software project activities mitigates accurate estimates.

There is an increasing need for more systematic methods and tools to supplement individual knowledge, judgment, and experience. These human traits are often sufficient to address less complex and isolated risks. For example, a portion of the most serious issues encountered in system acquisition are the result of risks that are ignored, due to its low likelihood, until they have already created serious consequences (Higuera and Haimes, 1996).

Project Management Institute (2008) presents Monte Carlo Simulation as a good practice method to project risk analysis. However, there are some limitations in the adoption of this approach that makes it unfeasible (Support,). Simulations can lead to misleading result if inappropriate inputs, derived from subjective parametrization, are entered into the model. Commonly, the user should be prepared to make the necessary adjustments if the results that are generated seem out of line. Moreover, Monte Carlo can not model risks correlations. That means the numbers coming out in each draw are random and in consequence, an outcome can vary from its lowest value, in one period, to the highest in the next. Therefore, alternative approaches must be considered to predict risk likelihood and impact.

The main purpose of this paper is to analyze which is a more efficient approach to software project risk analysis: Monte Carlo Simulation (MCS) technique or Artificial Neural Networks (ANN's) alternatives through Multilayer Perceptron (MLP) and Support Vector Machine (SVM) related to improved accuracy and decreased error prone. A Linear Regression Model (LRM) is also considered as baseline to evaluation method.

The methodology adopted in this study is a statistical experiment to evaluate the prediction error of risk impact from PERIL dataset (Kendrick, 2003), a framework to identify risks in software project management. The four selected techniques will estimate the outcome to risk impacts. Mean Absolute Error (MAE) will be calculated thirty times for each approach, and then a hypothesis test may be necessary to achieve the study goals.

Section 2 presents basic concepts to perform the experiment. Section 3 presents the methodology for this study, including dataset characterization. Section 4 presents the result analysis and establishes the best analyzed technique. In the end, Section 5 concludes this work and presents limitations and future works.

2 STATE OF ART

After a short bibliographic revision, we have identified numerous alternative approaches to risk analysis, which includes Bayesian Belief Networks, Artificial Neural Networks (ANN), Decision Tree (DT), Fuzzy Set Theory (FST), Neuro-Fuzzy System (NFS), etc.

Hu et al. (2007) utilized genetic algorithm to improve ANN estimator. Experimental results showed that it achieved higher accuracy when compared to SVM model. Dan (2013) proposed an ANN prediction model that incorporates with Constructive Cost Model (COCOMO) which was improved through particle swarm optimization, to estimate the software development effort accurately. Attarzadeh and Ow (2010) utilized ANN to improve accuracy on effort estimation compared to the traditional COCOMO. (Huang et al., 2004) have presented a general framework for software estimation based on NFS, the authors improved cost estimation for COCOMO'81.

Yu (2011) showed up a model based on fuzzy theory. It overcame the difficulty of qualitative and quantitative assessment compared to traditional methods. Saxena and Singh (2012) explored neuro-fuzzy techniques to design a suitable model to improve estimation of software effort for NASA software projects. Results showed that NFS has the lowest prediction error compared to existing models. Meanwhile, Lazzerini and Mkrtchyan 2011 suggested E-FCMs and extended E-FCMs themselves, introducing a special graphical representation for risk analysis.

Dzega and Pietruszkiewicz (2010) have presented results of risk analysis experiments performed using data mining classifiers: C4.5, RandomTree and classification and regression tree algorithms. They described how boosting and bagging metaclassifiers were applied to improve the outcomes and also analyzed influence of their parameters on generalization and in prediction accuracy. The authors rejected MLP and SVM prematurely.

2.1 Project Risk Management

According to Project Management Institute (2008), Project Risk Management includes planning, identification, analysis, response planning, monitoring and controlling risks. Its purpose is to increase likelihood and impact of positive events and reduce probability and severity of negative events. From management point of view, making informed decisions by consciously assessing what can go wrong, as well as its likelihood and severity of the impact, is at the heart of risk management. This activity involves the evaluation of the trade-offs associated with all policy options for risk mitigation in terms of their costs, benefits, risks and the evaluation of the impact of current decisions on future options.

Project risk management processes are:

- Planning risk management: The process of defining how conduct risk management activities;
- Identifying risks: The process of determining risks that can affect project and documenting its characteristics;
- Performing qualitative risk analysis: The process of prioritizing risks to analyze through assessment and combination of its occurrence probability and impact;

- Performing quantitative risk analysis: The process of analyzing numerically the effect of previous identified risks, in terms of general project objectives;
- Planning risk responses: The process of developing options and actions to increase opportunities and decrease threats to project objetives;
- Monitoring and controlling risks: The process of implementing risk responses planning, tracking identified risks, monitoring residual risks, identifying new risks and assessing the efficacy of risk treatment process during the whole project.

2.1.1 Risk Analysis

Analysis is the conversion of risk data into risk decision-making information. Analysis provides the basis for the project manager to work on the most critical risks. Boehm (1991) defines risk analysis objective as the assessment of the loss probability and magnitude for each identified risk item, and it assesses compound risks in risk-item interactions. Typical techniques include performance and cost models, network analysis, statistical decision analysis and quality-factor (like reliability, availability, security) analysis.

Risk analysis depends on a good mechanism to identify risks. However, most of the methods assume that managers have the required experience to be aware of all pertinent risk factors, but it can not be the situation. Moreover, many of these methods can be time-consuming and thus too costly to use on a regular basis. Therefore, one popular method for identifying risk factors has been the use of checklists. Unfortunately, these checklists are based in small samples or, even worse, flawed in their risk historical data collection methods.

According to Project Management Institute (2008), most used techniques are sensibility analysis, earned monetary value (EMV), modeling and simulation, specialized opinion.

2.1.2 Monte Carlo Simulation

Monte Carlo simulation is a technique that computes or iterates the project cost or schedule many times using input values selected at random from probability distributions of possible costs or durations, to calculate a distribution of possible total project cost or completion dates (Institute, 2008).

A model is developed, and it contains certain input variables. These variables have different possible values, represented by a probability distribution function of the values for each variable. The Monte Carlo method is a detailed simulation approach through intensive computing to determine the likelihood of possible outcomes of a project goal; for example, the completion date. The inputs of the procedure are obtained randomly from specific intervals with probability distribution functions for the durations of schedule activities or items from cost baseline. Those different input values are used to construct a histogram of possible results to the project and its relative probability, but also the cumulative probability to calculate desired contingency reserves for time or cost. Additional results include the relative importance of each input in determining the overall project cost and schedule (Kwak and Ingall, 2007).

2.2 Artificial Neural Networks

An ANN is a massively parallel distributed processor made up of simple processing units, which has a natural propensity (Lin, 1996). It adopts non-parametric regression estimates made up of a number of interconnected processing elements between input and output data.

2.2.1 MultiLayer Perceptron

MLP model is constituted of some neurons organized in at least three layers. The first of them is the input layer, in which input variables are directly connected to a exclusive neuron. The next is the hidden layer that completely connects the neurons from previous layer to the neurons in output layer. Lastly, output layer represents ANN outcome. Each input in a neuron has an associated weight to be adjusted by training algorithm. Common MLP models contain one bias neuron. MLP is a direct graph, in which inputs data are propagated from input layer to hidden layer*s* and from hidden layer*s* to output layer. The data flow in forward way is known as ”forward phase”. The data flow in the opposite way is the ”backward phase”.

One major concern of ANN is the stability-plasticity dilemma. Although continuous learning is desired in ANN, further learning will cause the ANN to lose memory when the weights have reached a steady state (Haykin, 1994). The Backpropagation algorithm is used as training method because it allow us to adjust weights of multilayer networks, towards Generalized Delta Rule (Rumelhart et al., 1985).

2.2.2 Support Vector Machine

Support Vector Machine (SVM) is an elegant tool for solving pattern recognition and regression problems. It has attracted a lot of attention from researchers due to its ability to provide excellent generalization performance. The goal of SVM regression is to estimate a function that is as ”close” as possible to the target outcomes for every input data in training set and at the same time, is as ”flat” as possible for good generalization. More details about SVM can be found in (Shevade et al., 1999).

3 METHODOLOGY

In this paper, we analyzed which is a more efficient approach to risk analysis of software projects: MCS, MLP, SVM or a LRM. A LRM was considered as baseline approach. The analysis was made in terms of prediction accuracy. Accuracy means the degree of closeness of a predicted outcome to the true value. A metric of accuracy is the Mean Absolute Error (MAE) given by

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i|, \qquad (1)$$

where $e_i = f_i - y_i$, f_i is the calculated outcome, y_i is the expected outcome and n is the number of data pairs.

The four selected techniques have predicted the outcome to risk impacts. Mean Absolute Error was calculated thirty times for each method. Nevertheless, a Non-paired Wilcoxon Test (Siegel, 1956) may be necessary to assert which is a more efficient approach to fit PERIL. Non-paired Wilcoxon Test is used because there were no evidence that the samples came from a normally distributed population, either there were no relation between outcomes from different samples.

One important requirement considered in this study is that the same prediction method must be adopted for each approach. Furthermore, cross-validation (Amari et al., 1996a) must be used to avoid the occurrence of overfitting of data training. For instance, *early stopping* training was used to identify the beginning of overfitting because this method has been proved to be capable of improving the generalization performance of the ANN over exhaustive training (Haykin, 1994) (Amari et al., 1996b). Therefore, cross-validation method are used for each alternative, excluding Monte Carlo Simulation, to promote higher generalization performance.

3.1 PERIL Data Set

A better risk management starts identifying potential problems, asserted here as risk factors. The adoption of available methods like: reviewing lessons learned, brainstorming, interviews and specialized judgment are relative efficient alternatives, otherwise in most of situations it involves high costs. A low cost, extensive and accessible proposal is to use PERIL dataset (Kendrick, 2003).

For more than a decade, in Risk Management Workshops, Kendrick have collected six hundred and forty nine anonymous risk registers from hundreds of project leaders dealing with their past project problems. He has compiled this data in the PERIL database, which summarizes both a description of what went wrong and the amount of impact it had on each project. The dataset provides a sobering perspective on what future projects may face and is valuable in helping to identify at least some of what might otherwise be invisible risks (Kendrick, 2003).

In projects, the identified risks can be classified as ”known”, those anticipated during planning, or ”unknown”, further identified during project execution. The purpose of this dataset is to provide a framework to identify risks, in such a way to increase the number of ”known”, and decrease the amount of ”unknown” risks.

Some characteristics of PERIL are:

- the data are not relational, they contain only most significant risks from tens of thousands projects undertaken by the project leaders from whom they were collected;
- they present bias, the information was not collected randomly; they are worldwide, with a majority from the Americas and they do not identify opportunities;
- the relative impact is based on the number of weeks delayed the project schedule;
- typical project had a planned duration between six months and one year and typical staffing was rarely larger than about twenty people.

Risk registers are categorized as scope, schedule and resource. Scope is decomposed in change and defect subcategories. Schedule is decomposed in dependency, estimative and delay. Resources is decomposed in money, outsourcing and people subcategories. One benefit of PERIL is that the author contemplates black swans - risks with large impact, difficult to predict and with rare occurrence (Taleb, 2001).

3.2 Data Preprocessing

First of all, in this analysis, we did not distinguished project location and collected year, those variables were not took into account. Secondly, PERIL contains nominal and numeric values. So, nominal variables were expressed through binary variables. In that point, it is used twelve binaries variables to represent eight nominal variables. Thirdly, impact which represents the real output, are integer numbers. We have noticed that impact probability distribution function fits with log-normal, gamma functions. Therefore, we have done a gamma data normalization (Han et al., 2006) limiting values between the interval I, where $I \in [0.15, 0.85]$. It was suggested by (Valenca, 2005).

Figure 1 and Figure 2 introduce input variables in histograms. All data are binary values represented by bar graphs, that means the number of occurrences for each value interval. Figure 3 presents gamma normalized real outcome from PERIL in a histogram. A shape of the distribution fitting function is also presented in a curve under the histogram.

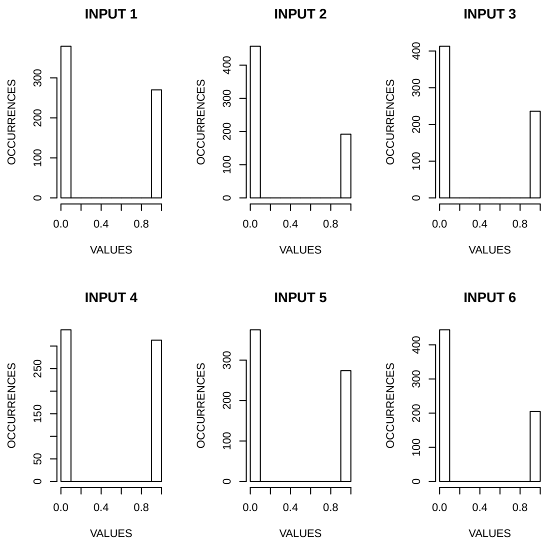

Figure 1: First six input variables

3.3 Tools

MCS was performed in Microsoft Office Excel. Data Analysis complement was utilized to obtain random values from customized sample. R language is a programming language and an environment for statistical computing, data manipulation, calculation and graphical display (Venables et al., 2002). R was utilized to conduct a experiment with Multiple Linear Regression (MLR) and Regression Tree Model (RTM). The MLP model used in this analysis was developed in Java. The source code implements data preprocessing, training, cross-validation, testing and MAE evaluation. It was based on (Valenca, 2005) book. We have utilized WEKA API (Hall et al., 2009) to program SVM. The built-in implementation of SVM is SMOreg. (Smola and Schoelkopf, 1998) proposed

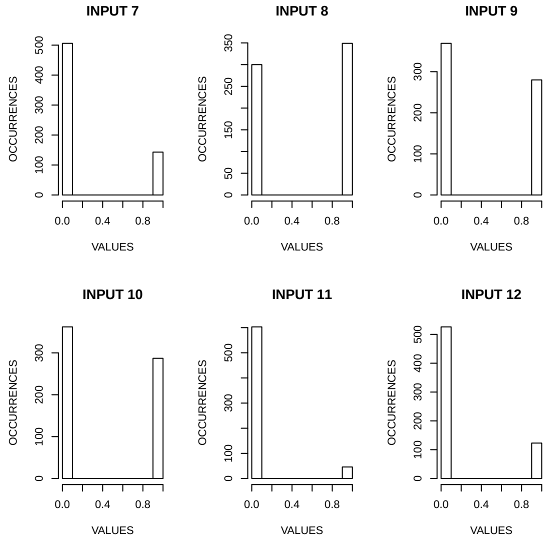

Figure 2: Last six input variables

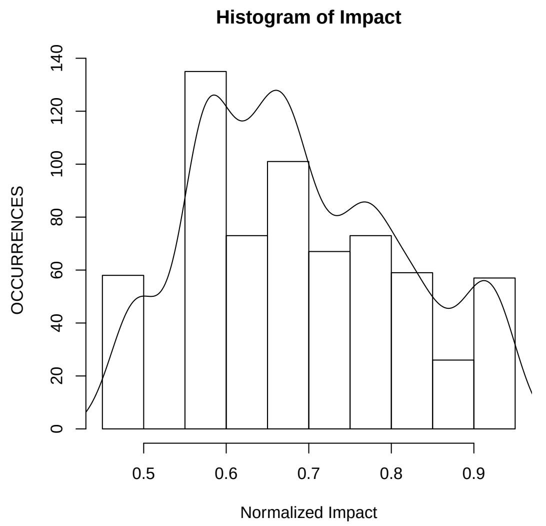

Figure 3: Histogram of impact and shape of the distribution fitting function

an iterative algorithm, called sequential minimal optimization (SMO), for solving the regression problem using SVM. SMOreg is a SMO program. SMOreg come across our needs, because the regression model could be generated after testing and cross-validation as stopping criteria.

3.4 Experiment

MCS technique used the entire dataset. In order to increase the performance prediction, we have filtered only the possible real outcomes to generate the calculated outcome. Towards this decision, we reduce generalization problems and improve its performance.

In this study, the source code of MLR model was adapted from Torgo (Torgo, 2003), in order to perform linear regression model training, cross-validation, outcome prediction and MAE evaluation. MRL model and RTM were analyzed statistically to define the baseline linear regression model for further analysis. The results for this previous analysis are also presented in Section 4.

A three layered back-propagation MLP model was established to model risk impact prediction mechanism. The model consists of one input layer, one hidden layer, and one output layer. The input layer had thirteen neurons, which represent the twelve independent variables plus the bias. The output layer has one neuron, which represents the single impact outcome. The number of neurons in the hidden layer was determined by trial and error to achieve more accurate performance of MLP. The transfer functions of hidden and output layer was sigmoid-logistic. The architecture of the MLP is demonstrated in Figure 4.

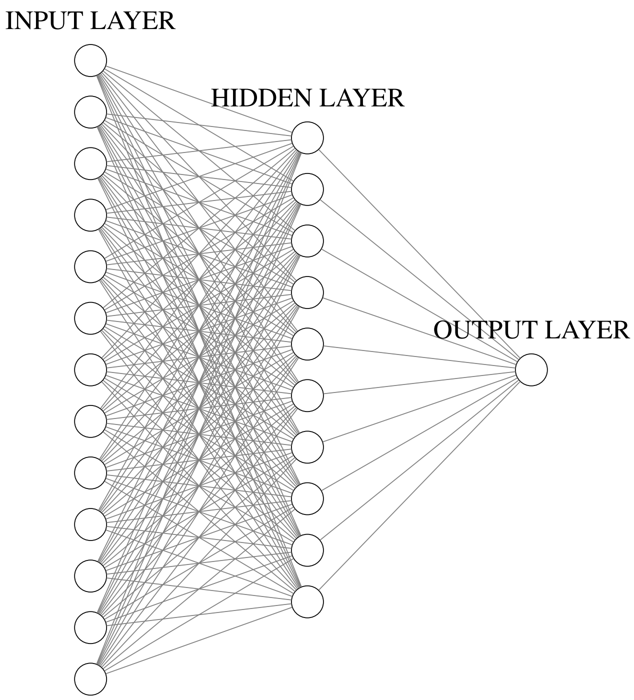

Figure 4: MLP model utilized in the study.

For our purpose, PERIL was split into three disjoint subsets - training, cross-validation and test subset, corresponding to fifty percent, twenty-five percent and twenty five percent of the dataset, respectively. The *early stopping* method and the *split-sample* cross-validation method were combined and used for ANN training (Priddy and Keller, 2005).

A previous experiment must be set to determine better MLP configuration. In this analysis, the maximum training epochs has been set at six hundreds. Starting with one neuron in the hidden layer, the MLP model was trained and tested. At each time, the number of neurons was increased by one, until reach ten, from then the number of neurons was increased by ten, until reach one hundred. Learning rate and momentum were increased by 0.1.

Learning rate, momentum and neurons in hidden layer varied from values presented in Table 1. A better parameters configuration investigated is shown in Table 2. Figure 4 presents the MLP model with better configuration for PERIL. The model contains ten neurons in hidden layer.

Table 1: Parameters intervals to MLP model.

Parameter	Min. Value	Max. Value
Momentum	0.5	0.9
Learning rate	0.1	0.5
Hidden Neurons	1	100

Table 2: A better parameters configuration to MLP model.

Parameter	Value
Momentum	0.5
Learning rate	0.1
Hidden Neurons	10
Maximum Cycles	600

PERIL was also partitioned into three disjoint subsets to be used with SVM, using the same percentage utilized in MLP. In SVM source code, RegSMOImproved object was the optimization algorithm and PolyKernel was the kernel function described in (Shevade et al., 1999).

4 RESULT ANALYSIS

Initially, the previous analysis consists of choosing between MLRM and RTM as baseline for the next analysis. It could be performed after discussing the information provided in Table 3 and in Figure 5.

Table 3 shows descriptive statistics of normalized MAE's to both algorithms. Average, standard deviation, minimum and maximum value are calculated for MLRM (cv.lm.v1), RTM_1 (cv.rpart.v1), RTM_2 (cv.rpart.v2) and RTM_3 (cv.rpart.v3). RTM_1, RTM_2 and RTM_3 are regression tree models instances automatically generated by R in this analysis. The first method has minor values in all statistics.

In Figure 5, normalized MAE's boxplots after predictions for RTM_3, RTM_2, RTM_1 and MLRM are

Table 3: Descriptive statistics for normalized errors of linear regression models.

	MLRM	RTM_1	RTM_2	RTM_3
Average	**0.09912**	0.10238	0.10305	0.10361
Std Dev	**0.00391**	0.00423	0.00441	0.00426
Min.	**0.08956**	0.09214	0.09321	0.09372
Max.	**0.10746**	0.11231	0.11267	0.11359

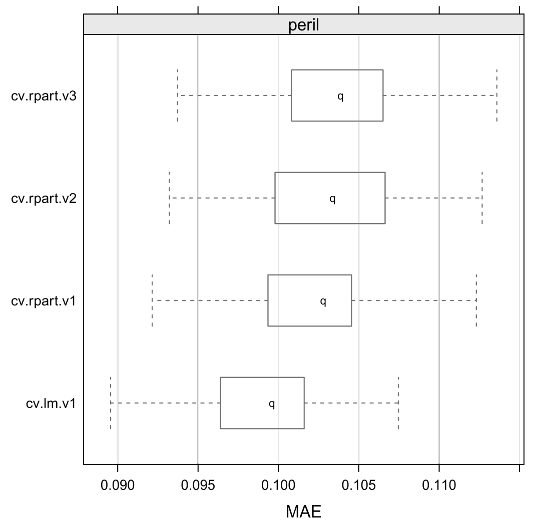

Figure 5: Boxplots for normalized errors of linear regression models.

presented. The boxplot placed more in the left is obtained to MLRM, this regression model also has minor standard deviation. From those information, we can affirm it is a more efficient and precise model and will be introduced in the experiment as baseline method.

Table 4 shows descriptive statistics of normalized MAE's for each approach. It was perceived that SVM has lower values for minimum (Min.), median, mean and maximum (Max.) errors. Nevertheless, MLP has minor standard deviation (Std.) value.

Table 4: Descriptive statistics for SVM, MLP, MLR and MCS.

	SVM	MLP	MLR	MCS
Min.	**0.08347**	0.09736	0.09764	0.10410
Median	**0.09374**	0.10014	0.10798	0.12740
Mean	**0.09430**	0.10005	0.10798	0.12640
Std.	0.00488	**0.00154**	0.00794	0.01250
Max.	**0.10284**	0.10413	0.12927	0.14950

In Figure 6, it is observed that the traditional technique, MCS (MonteCarlo), has a large standard deviation. It is due to randomness in MCS method, one of its limitation. Besides that, MCS has higher results in descriptive statistics. On the other hand, comparing MCS with MLR outcomes, it is noticed that MLR has better statistical values. Thus, we could not identify a reason to justify MCS usage as proposed by (Institute, 2008).

Besides that, MLP seems to be a promising alternative because it is such a optimized MLR. In this analysis, it is a more precise method to risk impact estimation. Since MLP can be interpreted as a nonlinear regression model, it must provide more valuable results than MLR.

Lastly, SVM is a more efficient approach because it explores minor results, but less precise than MLP. It has a good generalization capability such as MLP, since its inter-quartil interval is the second shorter. But above all, SVM could explore MAE minimization. We can realize it looking at Figure 7, where the most of values are near and above median value. Therefore, accordingly with this experiment, SVM seems to be a more accurate method to risk impact estimation using PERIL.

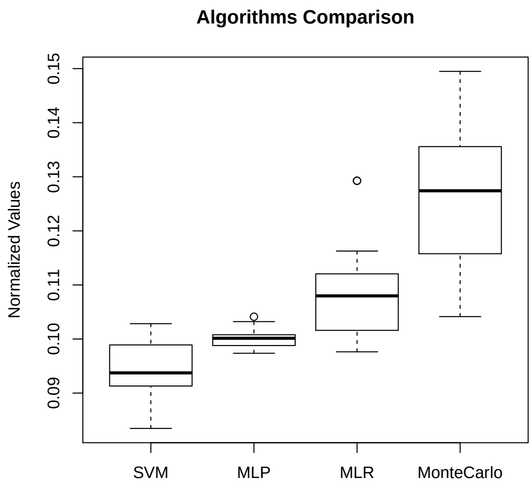

Figure 6: Boxplots of analyzed methods.

5 CONCLUSION

This paper has investigated the use of artificial neural networks algorithms, like SVM and MLP, for estimation of risk impact in software project risk analysis. We have carried out a statistical analysis using PERIL. The results were compared to MLRM and Monte Carlo Simulation, a traditional approach proposed by (Institute, 2008). We have considered improving risk impact estimation accuracy during soft-

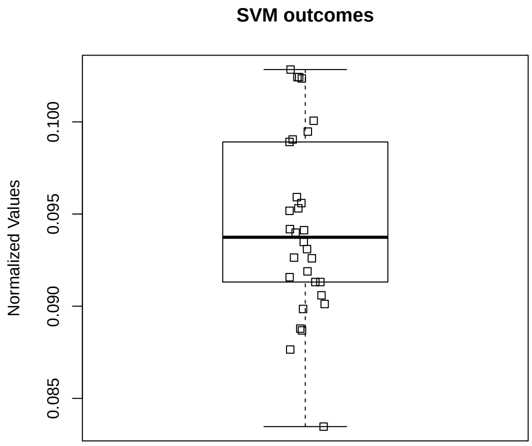

Figure 7: SVM boxplot with individual values.

ware project management, in terms of MAE mean and standard deviation. We have observed that MLP had minor standard deviation estimation error, and showed to be a promissory technique. Moreover, SVM had minor estimation error outcomes using PERIL, which a more accurate method. Therefore, the selected ANN algorithms outperformed both linear regression and MCS. Future works should analyze another ANN models and MLP training methods.

REFERENCES

Amari, S., Murata, N., Müller, K.-R., Finke, M., and Yang, H. (1996a). Statistical theory of overtraining-is cross-validation asymptotically effective? *Advances in neural information processing systems*, pages 176–182.

Amari, S.-i., Cichocki, A., Yang, H. H., et al. (1996b). A new learning algorithm for blind signal separation. *Advances in neural information processing systems*, pages 757–763.

Boehm, B. W. (1991). Software risk management: principle and practices. *IEEE Software*, 8:32–41.

Budzier, A. and Flyvbjerg, B. (2013). Double whammy-how ict projects are fooled by randomness and screwed by political intent. *arXiv preprint arXiv:1304.4590*.

Dzega, D. and Pietruszkiewicz, W. (2010). Classification and metaclassification in large scale data mining application for estimation of software projects. In *Cybernetic Intelligent Systems (CIS), 2010 IEEE 9th International Conference on*, pages 1–6. IEEE.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Han, J., Kamber, M., and Pei, J. (2006). *Data mining: concepts and techniques*. Morgan kaufmann.

Haykin, S. (1994). *Neural networks: a comprehensive foundation*. Prentice Hall PTR.

Higuera, R. P. and Haimes, Y. Y. (1996). Software risk management. Technical report, DTIC Document.

Huang, X., Ho, D., Ren, J., and Capretz, L. F. (2004). A neuro-fuzzy tool for software estimation. In *Software Maintenance, 2004. Proceedings. 20th IEEE International Conference on*, page 520. IEEE.

Institute, P. M. (2008). *A Guide to the Project Management Body of Knowledge*.

Kendrick, T. (2003). *Identifying and managing project risk: essential tools for failure-proofing your project*. Amacom: New York.

Kwak, Y. and Stoddard, J. (2004). Project risk management: lessons learned from software development environment. *Technovation*, 24(11):915–920.

Kwak, Y. H. and Ingall, L. (2007). Exploring monte carlo simulation applications for project management. *Risk Management*, 9(1):44–57.

Lazzerini, B. and Mkrtchyan, L. (2011). Analyzing risk impact factors using extended fuzzy cognitive maps. *Systems Journal, IEEE*, 5(2):288–297.

Lin, C.T., L. C. (1996). Neural fuzzy systems: A neuro-fuzzy synergism to intelligent systems.

Priddy, K. L. and Keller, P. E. (2005). *Artificial Neural Networks: An introduction*, volume 68. SPIE Press.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1985). Learning internal representations by error propagation. Technical report, DTIC Document.

Saxena, U. R. and Singh, S. (2012). Software effort estimation using neuro-fuzzy approach. In *Software Engineering (CONSEG), 2012 CSI Sixth International Conference on*, pages 1–6. IEEE.

Shevade, S., Keerthi, S., Bhattacharyya, C., and Murthy, K. (1999). Improvements to the smo algorithm for svm regression. In *IEEE Transactions on Neural Networks*.

Siegel, S. (1956). Nonparametric statistics for the behavioral sciences.

Smola, A. and Schoelkopf, B. (1998). A tutorial on support vector regression. Technical report. NeuroCOLT2 Technical Report NC2-TR-1998-030.

Support, I. P. *Monte Carlo Simulation*. IbbotsonAssociates, 225 North Michigan Avenue Suite 700 Chicago, IL 60601-7676.

Taleb, N. (2001). *Fooled by randomness*. Nwe York: Random House.

Torgo, L. (2003). Data mining with r. *Learning by case studies. University of Porto, LIACC-FEP. URL: http://www. liacc. up. pt/ltorgo/DataMiningWithR/. Accessed on*, 7(09).

Valenca, M. J. S. (2005). Aplicando redes neurais: um guia completo. *Livro Rpido, Olinda–PE*.

Venables, W. N., Smith, D. M., and Team, R. D. C. (2002). An introduction to r.

Yu, P. (2011). Software project risk assessment model based on fuzzy theory. *Computer Knowledge and Technology*, 16:049.