TECHNICAL UNIVERSITY OF CLUJ-NAPOCA

ACTA TECHNICA NAPOCENSIS

Series: Applied Mathematics, Mechanics, and Engineering Vol. 66, Issue III, August, 2023

ESTIMATION OF THE EFFORT REQUIRED TO DEVELOP A SOFTWARE THROUGH THE K-NEAREST NEIGHBORS METHOD

Anca-Elena IORDAN, Florin COVACIU

Abstract: *The purpose of the study presented in this article is to improve the efficiency of estimating the effort required to develop a software product by means of k-Nearest Neighbours machine learning method (KNN). The data set used for training KNN method is NASA93. The evaluation is related to the parameter tuning concept, KNN method being characterized by 2 parameters: the distance used to determine which are neighbours with common characteristics and the number of used neighbours for determining the prediction. To determine which version of KNN method provides the most accurate values for effort, three metrics were calculated: mean absolute error, mean squared error, and median absolute error. Implementation was done using Python programming language and Scikit-learn tool.*
Key words: *Software Effort Estimation, Machine Learning, KNN, Python, NASA93.*

1. INTRODUCTION

To guarantee precise quality, the effort put into the development and maintenance process of a software product must be approximated by project managers as accurately as possible. To help them, various research had been achieved based on methods from the fields of probability, artificial intelligence [1], and graph theory [2] in order to estimate the effort as precisely as possible.

This paper presents an effort estimation study based on KNN method [3], which is an artificial intelligence method. With this aim, the paper is structured as follows:

- The second section elaborates on KNN intelligent method used in this study.
- The third section includes the description of the data set used for prediction.
- The fourth section presents the design details for the prediction optimization.
- The fifth section describes the steps required to train the intelligent KNN model related to the parameter tuning concept.
- The sixth section evaluates the implemented model from the perspective of 3 used metrics.
- The last section presents the conclusions of this study.

2. KNN

Investigating the suitable literature in the sphere of artificial intelligence [4], two directions of machine learning [5] (ML) are deduced: unsupervised learning and supervised learning. Supervised learning is based on a training model that uses a set of labelled data. The most known supervised learning methods are: KNN [6], decision trees [7], SVM [8], Bayesian networks [9], and artificial neural networks [10].

Because within the analysed problem we want to acquire a numerical value for the effort, and not to classify the result in a category, a supervised learning method [11] based on regression will be used. This type of method approximates the numerical value that an instance of observations can have it. Thus, in the research presented in this article, the KNN method was chosen to predict the effort.

KNN is one of the simplest and fundamental classification and regression methods in machine learning. Distinguished from the other regression techniques, the KNN method is part of lazy learning, this means that there is no evident training phase before regression. Instead, KNN is based on a similarity of the features, which means that the level of similarity

Received: 18.06.23; **Similaritie**s: 04.07.23: **Reviewed**: 20.07./17.08.23: **Accepted**:18.08.23.

of the characteristics of the instances with those of the training set determines how the new value will be obtained.

3. DATASET ANALYSIS

There are a multitude of datasets that can be used by effort prediction models, such as: China, ISBSG, Albrecht, NASA63, NASA93, and Tukutuku. For this study, the NASA93 [12] set was chosen, which includes information from 93 software projects developed between 1971-1987. The information is structured into 24 attributes, and for prediction model presented in this article, 10 attributes were used as input data and one attribute for the output date.

The ten selected attributes are described in table 1. In order to obtain the accurate possible effort, the first 5 attributes: acap, pcap, modp, tool, and lexp should be characterized by large values, and the last attributes: stor, data, time, virt, and cplx should be characterized by small values.

Table 1

List with input chosen attributes.

Used Attributes	Attribute Description
acap	Analysis capability
pcap	Programmer capability
modp	Modern programming practices
tool	Use of software tools
lexp	Language experience
stor	Main memory constraint
data	Data base size
time	Time constraint for cpu
virt	Machine volatility
cplx	Process complexity

The attribute representing the output date is the actual effort (measured in Person-months), the values being obtained following the use of the Intermediate COCOMO model.

4. DESIGN DETAILS

In UML diagram [13] of the use cases (figure 1), the main components of the software implemented for estimating the effort are highlighted. "Data collection" use case represents the process by which the original and unprocessed data is obtained from a csv file. "Data preparation" use case refers to extraction of data from csv file according to the 11 attributes selected for this model, being dependent on the previous use case.

"Data segregation" use case represents the process of dividing the data set into two categories: training data and test data in order to check the performance of the model. From the 93 projects analysed by the NASA93 data set, 79% of the data were used for the training set (meaning 73 projects) and 21% of the data were used for the test set (meaning 20 projects). This case is dependent on "Data preparation" use case as represented in figure 1.

"Model training - KNN" use case refers to the training of the intelligent model based on the KNN method, being dependent on the "Data segregation" use case. This method will be analysed based on 2 parameters: the value of k (neighbours number) and the distance used to calculate the predicted value.

"Model evaluation" use case aims to determine that variant of KNN model that is characterized by the best performances.

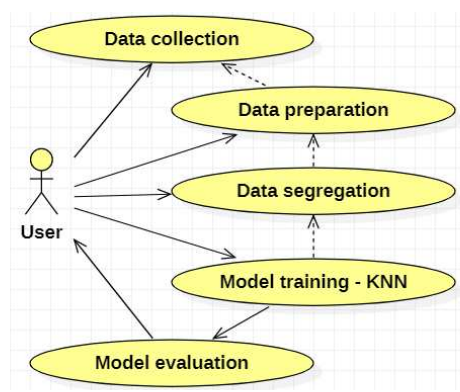

Fig. 1. UML use case diagram

5. MODEL TRANING

For certain methods of machine learning [14], the parameters are variables that they use to be able to learn the characteristics of the data and to be able to adjust the learning according to the data set, in order to obtain the best possible performance. Parameter tuning [15] concept involves finding the optimal parameters for each individual method, so that the results of the classifier are maximum. For KNN method, parameters that have been adjusted to determine optimal performance are as follows:

- kNeighbours - represents the number of neighbours taken into account to determine the prediction for new instances. In this study, the values chosen for this parameter are between 3 and 10.
- pMinkowski - represents the power parameter from Minkowski metric, given by the following formula:

$$d_{Minkowski} = (\sum_{i=1}^{n}(\lfloor x_i - y_i \rfloor)^p)^{\frac{1}{p}}. \quad (1)$$

Minkowski distance [16] is used by the KNN method to determine which are the neighbours that must be considered in order to compare their characteristics with those of a new instance for which a new prediction is desired. Thus, the distance metrics used determine which are the neighbours with the most similar characteristics and select the first kNeighbours from among them to calculate the new prediction.

For the prediction of the effort by the KNN method, 3 values for this second parameter between 1 and 3 were used. In the case of the value 1 for the pMinkowski parameter, the Manhattan metric is obtained (equation 2):

$$d_{Manhattan} = \sum_{i=1}^{n} \lfloor x_i - y_i \rfloor. \quad (2)$$

In case of choosing the value 2 for the pMinkowski parameter, the Minkowski metric is transformed into the Euclidean metric, represented by the following formula:

$$d_{Euclidean} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}. \quad (3)$$

Thus, the activities required for effort prediction by means of the KNN method characterized by the 2 parameters are presented in the UML activity diagram [17] in figure 2.

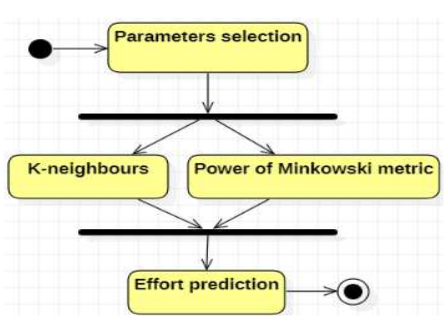

Fig. 2. UML activity diagram

The implementation of the KNN method was achieved using Python programming language [18] and sklearn.neighbors library belonging to the Scikit-learn tool [19].

6. MODEL EVALUATION

KNN method evaluation and establishment of the variant with the best results is achieved on the basis of test data set described in section 4. Thus, for realization of this process, three evaluation metrics [20] used in the case of regression problems were used:

- mean absolute error - signifies the average sum of absolute errors, characterized by the following formula:

$$MAE = \frac{1}{n} \cdot \sum_{i=1}^{n} \lfloor x_i - \hat{x}_i \rfloor. \quad (4)$$

- mean squared error – evaluates the standard deviation of the estimated value, characterized by the following formula:

$$MSE = \frac{1}{n} \cdot \sum_{i=1}^{n} (x_i - \hat{x}_i)^2. \quad (5)$$

- median absolute error – is calculated by taking the median of all absolute differences between the target and the prediction, characterized by the following formula:

$$MdAE = median(\{|x_i - \hat{x}_i|\}_{i=1}^{n}). \quad (6)$$

The calculation of the specific values of the 3 metrics was done through sklearn.metrics library belonging to the Scikit-learn tool [21].

In the case of mean absolute error metric, varying the 2 parameters kNeighbours and pMinkowski, the results shown in table 2 were obtained. A graphic comparison of the values obtained for MAE metric is shown in figure 3.

Analysing the values in table 2, it is found that the minimum value for the MAE metric was obtained in the case of using Manhattan distance and value 5 for the kNeighbours parameter, being equal to 188,812.

In the case of the mean squared error metric, varying the 2 parameters kNeighbours and pMinkowski, the results shown in table 3 were obtained.

Table 2

Mean absolute error values.

k \ d	Manhattan	Euclidean	Minkowski
3	218.280	390.300	339.350
4	208.505	282.732	277.668
5	188.812	333.705	264.915
6	333.392	317.312	241.169
7	359.463	352.978	359.987
8	366.235	415.629	470.837
9	359.453	433.755	494.918
10	337.126	455.511	466.531

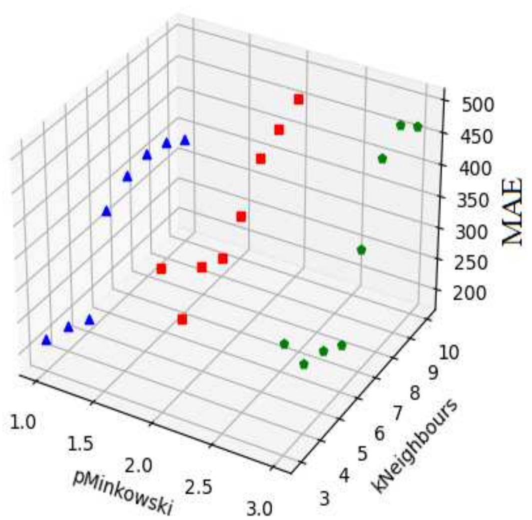

Fig. 3. Graphical representation - MAE

A graphic comparison of the values obtained for the MSE metrics is shown in figure 4.

Analysing the values in table 3, it is found that the minimum value for the MSE metric was obtained in the case of using Manhattan distance and value 5 for the kNeighbours parameter, being equal to 77600.596.

Table 3

Mean squared error values.

k \ d	Manhattan	Euclidean	Minkowski
3	105907.336	718510.204	583997.154
4	122054.139	371385.885	369796.779
5	77600.596	421617.145	272512.748
6	276603.631	274088.560	175444.459
7	313011.964	244778.697	254162.304
8	287861.258	420147.357	503718.432
9	251656.849	459235.509	555439.268
10	246960.323	493201.021	522394.392

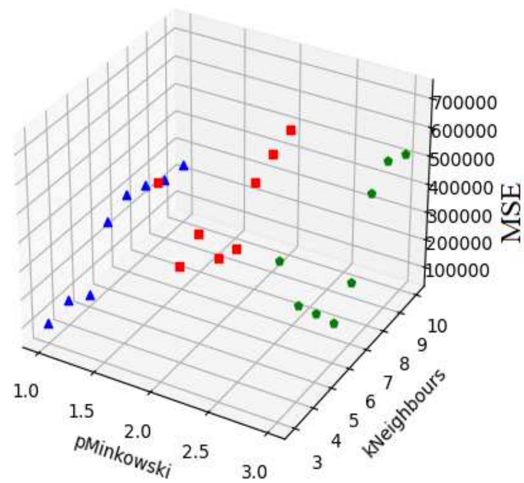

Fig. 4. Graphical representation - MSE

In the case of the median squared error (MdAE) metric, varying the 2 parameters kNeighbours and pMinkowski, the results presented in table 4 were obtained. A graphic comparison of the values obtained for the MdAE metrics is shown in figure 5.

Table 4

Median squared error values.

k \ d	Manhattan	Euclidean	Minkowski
3	94. 667	94.833	94.697
4	89.251	93.625	97.625
5	88.973	92.893	107.376
6	129.437	130.785	138.473
7	147.829	171.417	191.583
8	189.382	201.538	214.643
9	206.573	216.256	235.251
10	214.839	224.375	264.561

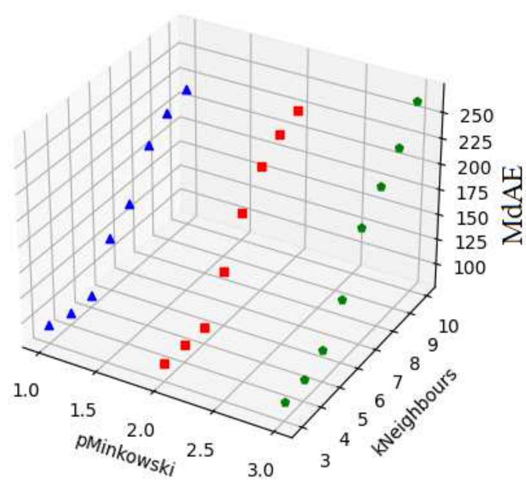

Fig. 5. Graphical representation - MdAE

Analysing the values in table 4, it is found that the minimum value for the MdAE metric was also obtained in the case of using Manhattan distance and value 5 for kNeighbours parameter, being equal to 88,973.

Considering that, according to the 3 metrics used, the best results were obtained in the case of using the Minkowski distance to the power of 1 (case when this becomes Manhattan distance) and the number of neighbours equal to 5, figure 6 shows a comparison between the current effort and the values predicted by the KNN method.

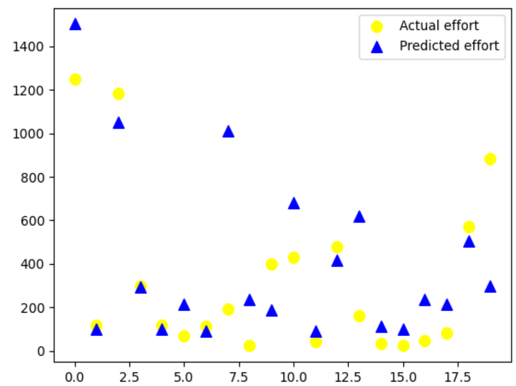

Fig. 6. Testing results

Analysing the previous results, it is found that KNN method characterized by value 5 for the kNeighbours parameter and value 1 for the pMinkowski parameter has the ability to better capture the sequential dependencies in the data set and to predict, as accurately as possible, the future values.

7. CONCLUSIONS

In the presented work, a software based on KNN machine learning method was designed, implemented, and evaluated to estimate the effort required to develop a software project expressed in Person-months.

To train the model, the NASA93 data set characterized by information obtained from 93 projects was used, and for its evaluation 3 metrics were calculated: mean absolute error, mean squared error, and median absolute error. The implementation was made using the Python programming language and the Scikit-learn tool.

In order to obtain results appropriate to the current effort, the two parameters that characterize the KNN method were varied, and predicted results were extracted in the situation where the 3 metrics are characterized by minimum values. Thus, the most accurate estimated values were obtained when the pMinkowski parameter is equal to 1 and the kNeighbours parameter is equal to 5.

This new approach chosen for improving the estimated effort obtained with the help of the KNN method can be improved, in the future, by comparing it with results generated by other intelligent methods.

8. REFERENCES

[1] Panoiu, M., Panoiu, C., Mezinescu, S., Militaru, G., Baciu, I. *Machine Learning Techniques Applied to the Harmonic Analysis of Railway Power Supply*, Mathematics 2023, Vol. 11(6).

[2] Iordan, A.E., *Optimal Solution of the Guarini Puzzle Extension using Tripartite Graphs*, IOP Conference Series: Materials Science and Engineering 2019, Vol. 477(1).

[3] Marapelli, B., *Software Development Effort Duration and Cost Estimation using Linear Regression and K-Nearest Neighbors*, International Journal of Innovative Technology and Exploring Engineering 2019, Vol. 9(2), 1043–1047.

[4] Covaciu, F., Iordan, A.E., *Control of a Drone in Virtual Reality using MEMS Sensor Technology and Machine Learning*, Micromachines 2022, Vol. 13(4), pp. 1-19.

[5] Rob, R., Panoiu, C, Rusu Anghel, S., *Intelligent System for tracking and logging the zigzag pantograph motion*, Innovations in Intelligent Systems and Applications 2018.

[6] Goyal, R., Chandra, P., Singh, Y., *Suitability of KNN Regression in the Development of Interaction Based Software Fault Prediction Models*, IERI Procedia 2014, Vol. 6, 15-21.

[7] Sanchez, E.R., Santacruz, E.FV., Maceda, H.C., *Effort and cost estimation using decision tree techniques and story points in agile software development*, Mathematics 2023, Vol. 11, 1477.

[8] Iordan, A.E., *Supervised learning use to acquire knowledge from 2D analytic geometry problems*, Recent Challenges in Intelligent Information and Database Systems 2022,189-200.

[9] Dragicevic, S., Turic, M., *Bayesian network model for task effort estimation in agile software*

development, Journal of Systems and Software 2017, Vol. 127, 109-119.
[10] Iordan, A.E., *Usage of Stacked Long Short-Term Memory for Recognition of 3D Analytic Geometry Elements*, Proceedings of the 14th International Conference on Agents and Artificial Intelligence 2022, Vol. 3, 745-752.
[11] Panoiu, M., Panoiu, C., Iordan, A., Ghiormez, L., *Artificial neural networks in predicting current in electric arc furnaces*, IOP Conference Series: Materials Science and Engineering 2014, Vol. 57(1), 012011.
[12] Saif, A., *A New Cost-Quality Estimation Model Based on Case-Based Reasoning Technique*, International Journal of Computer Science and Mobile Computing 2021, Vol. 10(3), 46-54.
[13] Iordan, A., Savii, G., Panoiu, M., Panoiu, C., *Visual interactive environment for doing geometrical constructions*, Wseas Transactions on Computers 2009, Vol. 8(2), 258-268.
[14] Iordan, A., *Development of an interactive environment used for simulation of shortest paths*, Annals of the Faculty of Engineering Hunedoara 2012, Vol. 10(3), 97-102.
[15] Mabayoje, A., Balogun, A., Hajarah, H., Atoyebi, J., Mojeed, H., Adeyemo, V., *Parameter tuning in KNN for software defect prediction: an empirical analysis*, Jurnal Teknologi dan Sistem Komputer 2019, Vol. 7(4), 121-126.
[16] Lu, B., Charlton, M., Brunsdon, C., Harris, P., *The Minkowski approach for choosing the distance metric in geographically weighted regression*, International Journal of Geographical Information Science 2015, Vol. 30(2), 1-18.
[17] Iordan, A., Savii, G., Panoiu, M., Panoiu, C., *Development of a dynamical software for teaching plane analytical geometry*, Mathematic and Computers in Science and Engineering 2008, Vol. 5, 55-60.
[18] Awar, N., Zhu, S., Biros, G., Gligoric, M., *A performance portability framework for Python*, Proceedings of the ACM International Conference on Supercom-puting, USA 2021, 467-478.
[19] Amin, M.Z., Ali, A., *An Intuitive Guide of K-Nearest Neighbor with Practical Implementation in Scikit Learn*, International Journal of Engineering Research and Technology 2019.
[20] Handelman, G.S., Kok, H.K., Chandra, R., Razavi, A., Huang, S., Brooks, M., Lee, M., Asadi, H., *Peering into the Black Box of Artificial Intelligence: Evaluation Metrics of Machine Learning Methods*, American Journal of Roentgenology 2019, 212, 38-43.
[21] Iordan, A.E., Covaciu, F., *Improving design of a triangle geometry computer application using a creational pattern*, Acta Technica Napocensis: Applied Mathematics, Mecha-nics and Engineering 2020, Vol. 63(1), 73-78.

Estimarea efortului necesar dezvoltării unui software prin intermediul metodei KNN

Scopul studiului prezentat în acest articol constă în eficientizarea estimării efortului necesar dezvoltării unui produs soft prin intermediul metodei de învățare automată k-Nearest Neighbors (KNN). Setul de date utilizat pentru antrenarea metodei este NASA93. Evaluarea este raportată la conceptul parameter tuning, metoda KNN fiind caracterizată de 2 parametrii: distanța utilizată pentru a determina care sunt vecinii având caracteristici comune și numărul de vecini luați în considerare pentru a determina predicția. Pentru a determina care variantă a metodei KNN furnizează cele mai exacte valori pentru efort, au fost calculate trei metrici: MAE, MSE și MdAE. Implementarea a fost realizată utilizând limbajul de programare Python și instrumentul Scikit-learn.

Anca-Elena IORDAN, PhD, Lecturer, Technical University of Cluj-Napoca, Computer Science Department, anca.iordan@cs.utcluj.ro, Barițiu street no. 28, Cluj-Napoca 400027, ROMANIA.
Florin COVACIU, PhD Eng., Associate professor, Technical University of Cluj-Napoca, Department of Design Engineering and Robotics, florin.covaciu@muri.utcluj.ro, Muncii Blvd. no. 103-105, Cluj-Napoca 400641, ROMANIA.