International Journal of Engineering and Management Research
e-ISSN: 2250-0758 | p-ISSN: 2394-6962
https://ijemr.vandanapublications.com

Peer Reviewed & Refereed Journal
Volume-13, Issue-3 (June 2023)
https://doi.org/10.31033/ijemr.13.3.42

Software Defect Prediction Based on Support Vector Classifier and Rule Mining

Fareha Bashir[1] and Dr. Akbar Shaun[2]
[1]PG Student, Department of Computer Science & Engineering, Integral University, Lucknow, INDIA
[2]Associate Professor, Department of Computer Science & Engineering, Integral University, Lucknow, INDIA

[1]Corresponding Author: farehabashir62@gmail.com

Received: 01-06-2023 Revised: 16-06-2023 Accepted: 30-06-2023

ABSTRACT

Software defect prediction plays a crucial role in ensuring the quality and reliability of software systems. Rule mining-based approaches have gained popularity in this domain as they provide insights into the relationships between software metrics and the occurrence of defects. This abstract presents an overview of software program defect prediction based totally on rule mining.

The process begins with the collection of historical data from previous software projects, encompassing defect records and associated software metrics. Relevant features are extracted from the data, including static code analysis metrics, change metrics, process metrics, and dynamic metrics. The collected data is then prepared by addressing data quality issues, handling missing values, and splitting it into training and testing sets.

Using a rule mining algorithm, such as association rule mining or decision tree induction, patterns and rules are discovered that correlate the software metrics with defect occurrences. The goal is to identify rules with high support and confidence, indicating strong associations between specific metrics and defect-prone areas of the software.

The discovered rules are evaluated using appropriate metrics, such as precision, recall, F1 score, or AUC-ROC, to assess their effectiveness in predicting defects. Once validated, the rules are applied to new software projects, where the software metrics are fed into the rule model to classify components as defect-prone or defect-free.

Continuous validation and improvement of the defect prediction model are necessary to ensure its accuracy and performance. This involves incorporating new data, refining the rules or metrics, and adapting the model to changing software development practices.

Software defect prediction based totally on rule mining offers a valuable approach for identifying potential defects early in the software development lifecycle. By leveraging historical data and discovering meaningful relationships between software metrics and defects, organizations can proactively allocate resources and implement preventive measures to improve software quality and reliability.

Keywords— Software Defect Prediction, Classification Algorithm, Cofusion Matrix

I. INTRODUCTION

Software Defect Predictor

There are various approaches for software defect prediction, each with its own strengths and limitations. Here are some commonly used approaches[1,2,3].

Statistical Models: Statistical models utilize machine learning algorithms to analyze historical data and identify patterns that correlate with software defects. Techniques such as logistic regression, Naive Bayes, random forests, and guide vector machines can be employed to build predictive models based on features extracted from software metrics. These models learn from past data to predict the likelihood of defects in new software [4,5].

Rule Mining: Rule mining techniques, such as association rule mining and decision tree induction, discover rules that capture relationships between software metrics and defect occurrence [10]. These rules can provide actionable insights into which combinations of metrics are indicative of defects. Rule mining approaches are often interpretable and can be used to gain a better understanding of the factors influencing software quality [6,7].

Text Mining and Natural Language Processing (NLP): Text mining and NLP techniques focus on analyzing textual artifacts such as bug reports, code comments, and documentation to predict defects. By extracting keywords, sentiment analysis, and other textual features, these approaches can uncover patterns and identify linguistic markers associated with software defects [8,9].

Text Mining and Natural Language Processing (NLP): Text mining and NLP techniques focus on analyzing textual artifacts such as bug reports, code comments, and documentation to predict defects. By extracting keywords, sentiment analysis, and other textual features, these approaches can uncover patterns and identify linguistic markers associated with software defects [10,11].

Peer Reviewed & Refereed Journal
Volume-13, Issue-3 (June 2023)
https://doi.org/10.31033/ijemr.13.3.42

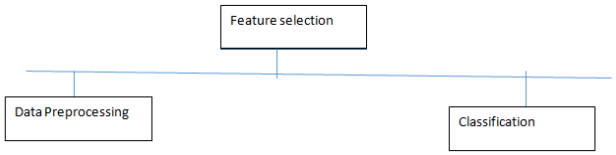
Ensemble Methods: Ensemble methods combine multiple models or predictions to achieve better defect prediction accuracy. This can be done by combining the outputs of different algorithms or training multiple models on different subsets of the data. Ensemble methods help mitigate the weaknesses of individual models and improve overall prediction performance [12,13].

Process-based Models: Process-based models leverage software development process metrics to predict defects. These metrics include code churn, code review activities, development effort, and developer experience. By analyzing process-related data, such as version control logs and issue tracking systems, these models can identify process factors that contribute to defect proneness [14,15].
Hybrid Approaches: Hybrid approaches combine multiple techniques mentioned above to improve defect prediction accuracy. For example, a hybrid model might utilize statistical features derived from software metrics and textual features extracted through NLP techniques to achieve a more comprehensive prediction [16,17].

A defect predictor is a tool or method that guides testing activities and software development lifecycle. According to Brooks, half the cost of software development is in unit and systems testing. Harold and Tahat also conform that testing phase requires approximately 50% or more of the whole project schedule. Therefore, the main challenge is the testing phase and practitioners seek predictors that indicate where the defects might exist before they start testing. This allows them to efficiently allocate their scarce resources. Defect predictors are used to make an ordering of modules to be inspected by verification and validation teams: • In the case where there are insufficient resources to inspect all code (which is a very common situation in industrial developments), defect predictors can be used to increase the chances that the inspected code will have defects. • In the case where all the code is to be inspected, but that inspection process will take weeks to months to complete, defect predictors can be used to increase the chances that defective modules will be inspected earlier. This is useful since it gives the development team earlier notification of what modules require rework, hence giving them more time to complete that rework prior to delivery [18].

II. DEFECT PREDICTION AS A CLASSIFICATION PROBLEM: ALGORITHMS

Binary Classification Algorithms
Bayesian Classification



Data Preprocessing: Data preprocessing is an important step in machine learning which highly affects the accuracy of the model. Two main steps in data preprocessing are: data cleaning and data transformation [19]. In data cleaning, we will look for the missing values and outliers in the data and will try to remove them using different methodologies that includes; mean substitution or the most probable value. Similarly in data transformation, the data can be converted from one format to another. Normalization, Discretization, and Attribute selection are normally used for this purpose[19,20].

In data normalization, we normalized the data which was very high in range as compared to the other samples in the column. To make all the data fall in the specified range we have used min_max normalization.. Min-max normalization is one of the most common transformation strategies, which linearly transforms the input data to a new range y. The normalized data is then concatenated with the data which was already in the specified range. The min-max normalization is done using the equation 1 where y is the normalized value, x is the value to be normalized, max is the maximum value of the attribute, min is the minimum value of the attribute, is the maximum value of the new range in which we are converting the input and is the minimum value of the new range in which we are converting the input

$$y = \frac{(x - \min)}{(\max - \min)}(n_{max} - n_{min}) + new_{min}$$

Feature Selection

The analysis to identify the strongest predictors (feature importance) is best addressed using correlation analysis. This allows it to assess all the features and complete training data to pick the strongest predictors. Other supervised learning approaches that are non-parametric such as K-Nearest Neighbors would not be able to rank the predictors by their importance. We have applied a heat-map analysis using a Spearman correlation matrix of the attributes. It is a visualization method in two-dimension where numerical values are displayed in colors and arranged in rows and columns [The heat-map was coded in Python using the function heat-map from the Seabornlibrary[21,22]

Peer Reviewed & Refereed Journal
Volume-13, Issue-3 (June 2023)
https://doi.org/10.31033/ijemr.13.3.42

Naive Bayes Classifier: Naive Bayes Classifier is one of the clasification that works based on the method of Bayes theory. The eqution of bayes theory is as follow

$$P(c \mid A) = \frac{P(A/C)P(C)}{P(A)}$$

where c is a class variable, A is a class of data not yet known, P(A|c) is Probability based on condition on hypothesis, P(c) is probability of hypothesis (prior probability), while P(A) ignored because of constant value for all classes. A further description of the Bayes formula is made by describing P(c|A1,A2 ..., An) using the rules of multiplication[23,24].

This is where the assumption of independence is very high (naive), that each instruction is independent of each other. With these assumptions, then apply a similarity as follows:

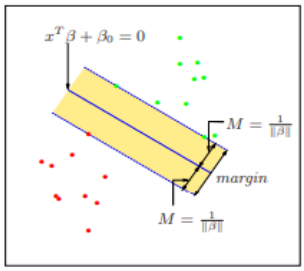
$$P(A/c) = P(a_1, a_2, a_3 \dots a_n) = \prod_{i=1}^{n} P(A_i/c)$$

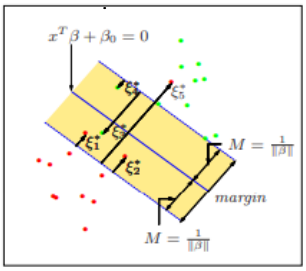
the naive bayes classifier can be calculated as follow

$$a_n(A) = \frac{P(c = +)}{P(a = -)} \prod_{k=1}^{n} \frac{P(ak/c = +)}{P(a_k/c^-)}$$

If there is a label that never appears in the event then the way to handle the probability value 0 (zero) is to add 1 piece of data by using Laplace Correction method or also known as Laplacian Estimator method. In this paper the performance of naive bayes is be tested in 3 ways: supplied testing data set, cross validation, where has been taken as 10, and percentages of 66% as training data, 34% as testing data.

Support Vector Classifier [10]: by the linear classifier the two classes are linearly separable; by the support vector machine the two non-separable classes' or two overlap classes are classified. The support vector machine produced nonlinear boundary by constructing a linear boundary in a large. Transform the version of the feature space. The second approach for generalized no separable region by fisher linear discriminant analysis[10, 25,26].





The left panel show the separable classifier, solid line is the decision boundary, the shaded broken line bound the maximum margin of the width 2M=2/ $\|\beta\|$

The right panel here the non-separable cases that point labelled ξ_j^* are the wrong side of their margin by an amount $\xi^* = M\xi_j$ point on the correct side have $\xi_j^* = 0$

Our training data having N pair of data like(x1,y1), (x2,y2),…..(xn,yn)

With $x_i \in \mathrm{R}^p$ and $y_i \in \{-1,1\}$

Hyper plan can be define by $\{x: f(x) = x^T\beta + \beta_0 = 0\}$

Where β is the unit vector $\|\beta\|$=1 now classification introduce by f(x)

$$G(x) = \text{sign}[x^T\beta + \beta_0]$$

The hyperplan$f(x) = x^T\beta + \beta_0 = 0$ since the classes are separable, now we can find the solution $f(x) = x^T\beta + \beta_0$ with $y_i f(x_i) > 0 \forall i$hence to find the hyperplan that creat the biggest margin between the training point for classes 1 and -1 the optimaization problem[27]

$$\max_{\beta,\beta_0 \|\beta\|=1} M$$

Subject to $y_i = (x_i^T\beta + \beta_0) > Mi = 1,2,\cdots N$ the problem more be more convienientrepharase as

$$\min_{\beta,\beta_0} \|\beta\|$$

Subject to $y_i(x^T\beta + \beta_0) \geqslant 1i = 1,\cdots N$

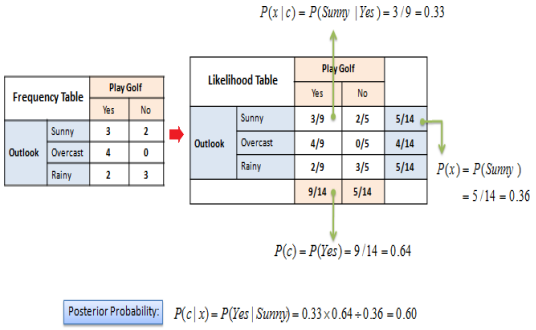
Analysis: An analysis of the results of any performance that performed to predict the weight of a newborn, it has been evaluated by looking at prediction accuracy, Precision and Recall for the accuracy of the resulting model.

Where TP (True Positive) is data that can be properly calcified on a positive label, TN (True Negative) is the amount of data correctly specified in the negati class. While FP is a lot of data that is recognized as a positive label when the actual value in the negative class, FN (Negative Flase) is the amount of data introduced as a negative class, it is classified as a positive class.

III. RULE-BASED CLASSIFICATION

Rules are a good way of representing information or bits of knowledge. A rule-based classifier uses a set of IF-THEN rules for classification. An IF-THEN rule is an expression of the formIF condition THEN conclusion Example:

Peer Reviewed & Refereed Journal
Volume-13, Issue-3 (June 2023)
https://doi.org/10.31033/ijemr.13.3.42



IF age=youth AND student=yes THEN buys computer=yes There are many rule-based classifier algorithms are there. Some of them are: DecisionTable,OneR, PART, JRip, ZeroR.

Data Set.

We used the records(data) taken from the public NASA MDP repository, which was once additionally used by way of MGF and many others, e.g., [10], [11], [12], [13]. Thus, there are 12 facts units in complete from NASA MDP repository. Table 4.1, and 4.2 provides some basic summary information. Each data set is comprised of a number of software modules (cases), each containing the corresponding number of defects and various software static code attributes. After pre-processing, modules that contain one or more defects were labelled as defective. A moredetailed description of code attributes or the origin of the MDP data sets can be obtained from [5].

Data Set	System	Language	Total Loc
CM 1-5	Spacecraft Instrument	C	17K
KC 3-4	Storage management for ground data	JAVA	8K and 25K
KC 1-2	Storage management for ground data	C++	*
M W1	Database	C	8K
PC1, 2,5	Flight Software for Earth orbiting Software	C	26K
PC 3,4	Flight Software for Earth orbiting Software	C	30-36K

Table 3.1: NASA MDP Data Sets

Data Set	System	Language	Total Loc
CM 1-5	Spacecraft Instrument	C	17K
KC 3-4	Storage management for ground data	JAVA	8K and 25K
KC 1-2	Storage management for ground data	C++	*
M W1	Database	C	8K
PC1, 2,5	Flight Software for Earth orbiting Software	C	26K
PC 3,4	Flight Software for Earth orbiting Software	C	30-36K

Table 3.2: Data Sets

Calculations

The Performance measured according to the Confusion matix given in table:3.3, which is used by many researchers e.g [14], [5]. Table 3.3 illustrates a confusion matrix for a two type hassle having superb and poor class.

Predicted Class

		Positive	Negative
Actual class	Positive	Trure Positive	False Negative
	Negative	False Positive	True negative

Table 3.3: Confusion Matrix

Software defect predictor performance of the proposed scheme based on Accuracy, Sensitivity, Specificity, Balance, ROC Area defined as —

IV. RESULT DISCUSSION

This section provides simulation results of some of the Classification algorithm techniques collected by simulation on Software tool named weka (version 3.6.9). In the thesis, however, proposed schemes are more comprehensively compared with competent schemes.

According to best accuracy value we choose classification algorithm among many classification algorithms. All the evaluated values are collected and compare with different performance measurement parameter.

Accuracy

From the accuracy table 4.1 we can see different algorithm giving different accuracy on different data set. But the average performance nearly same.

For Storage management software(KC1-3) LOG, J48G giving better Accuracy value. For database software written in c programming language (MW1) only PART

Peer Reviewed & Refereed Journal
Volume-13, Issue-3 (June 2023)
https://doi.org/10.31033/ijemr.13.3.42

giving better accuracy value.

The performance graph is given in the figure 4.3.

Methods	NB	LOG	DT	JRip	OneR	PART	J48	J48G
CM1	83.94	87.68	89.13	86.23	89.13	73.91	86.23	86.96
JM1	81.28	82.02	81.57	81.42	79.67	81.13	79.8	79.83
KC1	83.05	86.87	84.84	84.84	83.29	83.89	85.56	85.56
KC3	77.5	71.25	75	76.25	71.25	81.25	80	82.5
MC1	94.34	99.27	99.25	99.22	99.3	99.19	99.3	99.3
MC2	66	66.67	56.86	56.86	56.86	70.59	52.94	54.9
MW1	79.25	77.36	85.85	86.79	85.85	88.68	85.85	85.85
PC1	88.82	92.11	92.43	89.14	91.45	89.8	87.83	88.49
PC2	94.29	99.05	99.37	99.21	99.37	99.37	98.9	98.9
PC3	34.38	84.67	80.22	82.89	82.89	82.67	82.22	83.56
PC4	87.14	91.79	90.18	90.36	90.18	88.21	88.21	88.93
PC5	96.56	96.93	97.01	97.28	96.9	96.93	97.13	97.16

Sensitivity

From the accuracy table 4.2 we see that NB algorithm gives better performance in maximum data set. In case of DecisionTable gives the sensitivity zero(sometimes), that means it considering all the class as a true negetive. It can not be cosider for defect prediction. LOG, OneR, PART, J48, J48G algorithms giving average performance.

Methods	NB	LOG	DT	JRip	OneR	PART	J48	J48G
CM1	0.4	0.267	0	0.2	0.133	0.333	0.2	0.2
JM1	0.198	0.102	0.07	0.157	0.109	0.03	0.131	0.123
KC1	0.434	0.238	0.197	0.328	0.254	0.32	0.32	0.32
KC3	0.412	0.412	0.118	0.118	0.176	0.353	0.353	0.353
MC1	0.548	0.161	0.194	0.161	0.161	0.194	0.161	0.161
MC2	0.571	0.545	0	0	0.091	0.5	0.045	0.045
MW1	0.429	0.286	0.429	0.143	.071	0.286	0.214	0.214
PC1	0.28	0.24	0.16	0.16	0.08	0.36	0.24	0.24
PC2	0.333	0	0	0	0	0	0	0
PC3	0.986	0.178	0	0.233	0.014	0.137	0.288	0.288
PC4	0.431	0.538	0.231	0.508	0.323	0.677	0.692	0.677
PC5	0.427	0.308	0.332	0.521	0.303	0.474	0.498	0.479

Balance

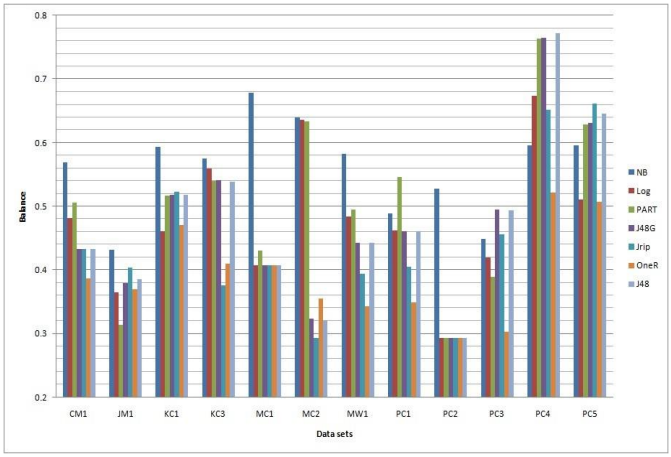
Looking to the Accuracy, Sensitivity and Specificity performance table we consider the NB, LOG, JRip, OneR, PART, J48, J48G, as their performance are average.

From the graph figure 4.1 we see that, in maximum of cases the OneR algorithm giving lowest balance value than others. So, no need to use for defect prediction.

Methods	NB	LOG	DT	JRip	OneR	PART	J48	J48G
CM1	0.569	0.481	0.293	0.433	0.387	0.505	0.433	0.433
JM1	0.432	0.365	0.342	0.403	0.369	0.314	0.385	0.379
KC1	0.593	0.461	0.431	0.523	0.47	0.516	0.518	0.518
KC3	0.575	0.559	0.374	0.375	0.409	0.54	0.539	0.541
MC1	0.678	0.407	0.43	0.407	0.407	0.43	0.407	0.407
MC2	0.639	0.636	0.293	0.293	0.355	0.633	0.321	0.323
MW1	0.582	0.484	0.593	0.394	0.343	0.495	0.443	0.443
PC1	0.489	0.462	0.406	0.405	0.349	0.546	0.461	0.461
PC2	0.527	0.293	0.293	0.293	0.293	0.293	0.293	0.293
PC3	0.448	0.419	0.292	0.456	0.303	0.389	0.494	0.495
PC4	0.595	0.673	0.456	0.651	0.521	0.763	0.772	0.764
PC5	0.595	0.511	0.528	0.661	0.507	0.628	0.645	0.631

Depending on Accuracy, Sensitivity, Specificity, Balance performance we choosen 6 Algoritms from 8 algoritms are–

NaiveBayesSimple
Logistic
JRip
PART
J48 and J48Graft



Peer Reviewed & Refereed Journal
Volume-13, Issue-3 (June 2023)
https://doi.org/10.31033/ijemr.13.3.42

V. CONCLUSION

Summarize the key findings and implications of the study. State whether the combination of Support Vector Classifier and Rule Mining was effective in predicting software defects and how it contributes to the existing body of knowledge in software defect prediction.
Remember that the specific content of the conclusion will depend on the actual results and outcomes of the study. The conclusion should provide a clear and concise summary of the research, highlighting the significance of the chosen approaches and their impact on software defect prediction.

REFERENCES

[1] Chandra, S. D. (2013). Software defect prediction based on classification rule mining. *Department of Computer Science and Engineering National Institute of Technology Rourkela Rourkela {769 008, India*.

[2] Shao, Y., Liu, B., Wang, S. & Li, G. (2020). Software defect prediction based on correlation weighted class association rule mining. *Knowledge-Based Systems*, *196*, 105742.

[3] Shao, Y., Liu, B., Wang, S. & Li, G. (2018). A novel software defect prediction based on atomic class-association rule mining. *Expert Systems with Applications*, *114*, 237-254.

[4] Srivastava, S., Haroon, M. & Bajaj, A. (2013, September). Web document information extraction using class attribute approach. In *2013 4th International Conference on Computer and Communication Technology (ICCCT)*, pp. 17-22.

[5] Khan, W. & Haroon, M. (2022). An efficient framework for anomaly detection in attributed social networks. *International Journal of Information Technology*, *14*(6), 3069-3076.

[6] Khan, W. & Haroon, M. (2022). An unsupervised deep learning ensemble model for anomaly detection in static attributed social networks. *International Journal of Cognitive Computing in Engineering*, *3*, 153-160.

[7] Husain, M. S. & Haroon, D. M. (2020). An enriched information security framework from various attacks in the IoT. *International Journal of Innovative Research in Computer Science & Technology (IJIRCST)*.

[8] Khan, W. (2021). An exhaustive review on state-of-the-art techniques for anomaly detection on attributed networks. *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, *12*(10), 6707-6722.

[9] Husain, M. S. (2020). A review of information security from consumer's perspective especially in online transactions. *International Journal of Engineering and Management Research*, *10*.

[10] Minasny, B. (2009). *The elements of statistical learning, trevorhastie, roberttishirani, jeromefriedman*.

[11] Khan, A. M., Ahmad, S. & Haroon, M. (2015, April). A comparative study of trends in security in cloud computing. In: *Fifth International Conference on Communication Systems and Network Technologies*, pp. 586-590.

[12] Khan, W., Haroon, M., Khan, A. N., Hasan, M. K., Khan, A., Mokhtar, U. A. & Islam, S. (2022). DVAEGMM: Dual variational autoencoder with gaussian mixture model for anomaly detection on attributed networks. *IEEE Access*, *10*, 91160-91176.

[13] Khan, N. & Haroon, M. (2022). Comparative study of various crowd detection and classification methods for safety control system. *Available at: SSRN 4146666*.

[14] Siddiqui, Z. A. & Haroon, M. (2022). Application of artificial intelligence and machine learning in blockchain technology. In: *Artificial Intelligence and Machine Learning for EDGE Computing*, pp. 169-185. Academic Press.

[15] Czibula, G., Marian, Z. & Czibula, I. G. (2014). Software defect prediction using relational association rule mining. *Information Sciences*, *264*, 260-278.

[16] Tripathi, M. M., Haroon, M., Khan, Z. & Husain, M. S. (2022). Security in Digital Healthcare System. *Pervasive Healthcare: A Compendium of Critical Factors for Success*, 217-231.

[17] Haroon, M., Tripathi, M. M. & Ahmad, F. (2020). Application of machine learning in forensic science. In *Critical Concepts, Standards, and Techniques in Cyber Forensics*, pp. 228-239. IGI Global.

[18] Ma, B., Dejaeger, K., Vanthienen, J. & Baesens, B. (2011). Software defect prediction based on association rule classification. *Available at SSRN: 1785381*.

[19] Shakeel, N., Haroon, M. & Ahmad, F. (2021). A study of wsn and analysis of packet drop during transmission. *International Journal of Innovative Research in Computer Science & Technology*.

[20] Haroon, M., Tripathi, M. M., Ahmad, T. & Afsaruddin. (2022). Improving the healthcare and public health critical infrastructure by soft computing: An overview. *Pervasive Healthcare: A Compendium of Critical Factors for Success*, 59-71.

Peer Reviewed & Refereed Journal
Volume-13, Issue-3 (June 2023)
https://doi.org/10.31033/ijemr.13.3.42

[21] Khan, N. & Haroon, M. (2023). A personalized tour recommender in python using decision tree. *International Journal of Engineering and Management Research*, *13*(3), 168-174.

[22] Haroon, M., Misra, D. K., Husain, M., Tripathi, M. M. & Khan, A. (2023). Security Issues in the Internet of Things for the Development of Smart Cities. In: *Advances in Cyberology and the Advent of the Next-Gen Information Revolution*, pp. 123-137. IGI Global.

[23] Scholar, P. G. (2021). *Satiating a user-delineated time constraints while scheduling workflow in cloud environments*.

[24] Siddiqui, Z. A. & Haroon, M. (2022). Application of artificial intelligence and machine learning in blockchain technology. In: *Artificial Intelligence and Machine Learning for EDGE Computing*, pp. 169-185. Academic Press.

[25] Haroon, M., Tripathi, M. M. & Ahmad, F. (2020). Application of machine learning in forensic science. In: *Critical Concepts, Standards, and Techniques in Cyber Forensics*, pp. 228-239. IGI Global.

[26] Hoda, M. (2015). *Computing for sustainable global development*. (INDIACom).

[27] Chen, Y., Shen, X. H., Du, P. & Ge, B. (2010, February). Research on software defect prediction based on data mining. In: *The 2nd International Conference on Computer and Automation Engineering, 1*, pp. 563-567. IEEE.

[28] Khan, W., Haroon, M., Khan, A. N., Hasan, M. K., Khan, A., Mokhtar, U. A. & Islam, S. (2022). DVAEGMM: Dual variational autoencoder with gaussian mixture model for anomaly detection on attributed networks. *IEEE Access, 10*, 91160-91176.

[29] Tripathi, M. M., Haroon, M., Khan, Z. & Husain, M. S. (2022). Security in digital healthcare system. *Pervasive Healthcare: A Compendium of Critical Factors for Success*, 217-231.

[30] Hasanpour, A., Farzi, P., Tehrani, A. & Akbari, R. (2020). Software defect prediction based on deep learning models: Performance study. *arXiv preprint arXiv:2004.02589*.

[31] Khan, N. & Haroon, M. (2023). A personalized tour recommender in python using decision tree. *International Journal of Engineering and Management Research, 13*(3), 168-174.

[32] Khan, W. & Haroon, M. (2022). A pilot study and survey on methods for anomaly detection in online social networks. In: *Human-Centric Smart Computing: Proceedings of ICHCSC 2022*, pp. 119-128). Singapore: Springer Nature Singapore.

[33] Tripathi, M. M., Haroon, M. & Ahmad, F. (2022). A survey on multimedia technology and internet of things. *Multimedia Technologies in the Internet of Things Environment, 2*, 69-87.

[34] Soe, Y. N., Santosa, P. I. & Hartanto, R. (2018, March). Software defect prediction using random forest algorithm. In: *12th South East Asian Technical University Consortium (SEATUC), 1*, pp. 1-5. IEEE.

[35] Kumar, L., Sripada, S. K., Sureka, A. & Rath, S. K. (2018). Effective fault prediction model developed using least square support vector machine (LSSVM). *Journal of Systems and Software, 137*, 686-712.