REVIEW PAPER DOI: 10.21844/16202117206

Vol. XVII, No. 2; September 2024 - February 2025 Print ISSN: 0975-024X; Online ISSN: 2456-1371

Machine Learning Models for Software Defect Detection: A Strategic Management Approach

Sripriya Roy Chowdhuri[1], Manjari Gupta[2]

[1]Assistant Professor, School of Management Sciences, Varanasi, & Research Scholar, DST-Centre for Interdisciplinary Mathematical Sciences, BHU, Varanasi

[2]Departtment of Computer Science, Banaras Hindu University, India & Coordinator, DST-Centre for Interdisciplinary Mathematical Sciences, BHU, Varanasi

Abstract

Businesses are at serious risk from software flaws, which can affect consumer trust, security, and operational effectiveness. Proactive quality assurance is made possible by machine learning's (ML) creative approaches to software defect detection and prediction. From the standpoint of strategic management, companies can manage the allocation of resources efficiently, increase decision-making, and improve overall software quality by incorporating machine learning (ML) models into defect detection procedures. This study examines the business ramifications of AI-driven quality assurance as well as the different machine learning models used for software defect identification and their incorporation into IT management plans. The paper emphasizes ML's significance in long-term strategic planning and operational efficiency alongwith highlighting the difficulties, advantages, and potential future directions of using it for defect management in organizational settings.

Keywords: Machine learning, software defects, defect detection, IT management, strategic management, AI in quality assurance.

Introduction

Defects in software applications produce substantial economic damage and negative reputation effects and security holes which makes defect management essential for effective software engineering and IT governance. The software industry faces intense market competition together with rapid growth and technological advancements according to *Trayambak et al. (2016).* Software reliability stands today as both a technical obligation and strategic business requirement in organizations of the present business world. Today's organizations depend heavily on their software systems for operational efficiency together with enhanced customer trust which leads to market competitiveness. Manual testing together with rule-based methods have demonstrated their ineffectiveness in managing large software projects because they produce excessive costs while requiring significant time and introducing human mistakes. Machine learning (ML) has become the foundational transformative solution that enables automatic defect identification and prediction through data algorithms. Organizations who incorporate ML into defect management systems can gain predictive abilities for software vulnerability identification as well as better decision processes and improved resource optimization. Finding quality people has become essential due to a competitive industry, rising technology levels for conducting business, and increasingly demanding customers both domestically and internationally *(Singh and Sabharwal, 2020).* This will lead to emergence of quality decision making for better software development.

(Corresponding Author: Sripriya Roy Chowdhuri, Assistant Professor, School of Management Sciences, Varanasi, & Research Scholar, DST-Centre for Interdisciplinary Mathematical Sciences, BHU, Varanasi
E-mail: sripriya@smsvaranasi.com
How to Cite the article: Chowdhuri, S.R.; Gupta, M. (2025). Machine Learning Models for Software Defect Detection: A Strategic Management Approach . Purushartha, 17(2), 87-98
Source of Support: Nil
Conflict of interest: None

The world has experienced tremendous industrialization as a result of the astounding

© The Author(s). 2025 Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and non-commercial reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The Creative Commons Public Domain Dedication waiver (http://creativecommons.org/publicdomain/zero/1.0/) applies to the data made available in this article, unless otherwise stated.

scientific and technological advancements over the past few decades *(Ali and Sinha, 2016). Padma and Aravamudhan (2021)* suggest that employees should be made aware that nothing in the company they work for is constant and that constant change is necessary to keep things from becoming outdated. In addition to evaluating the operational environment's scope and regulations, one must also focus on making the best use of the limited resources available *(Kumar and Bhandarkar, 2020).*

From a strategic management perspective, leveraging ML models for defect detection allows businesses to align their software quality assurance processes with long-term corporate goals. AI-driven defect management fosters operational resilience, supports agile development practices, and enhances risk mitigation strategies. While successful firms institutionalize innovation, which is consistently reflected in their products, processes, and business models, many companies exhibit bursts of innovation during brief periods of time or in certain business or product lines *(Botla and Kondur, 2018).* Software defects place businesses under significant danger because they harm consumer trust and operational effectiveness as well as security. Machine learning through its innovative approaches enables the development of proactive quality assurance systems for software defect identification and forecast. Strategic management gains enhanced operational efficiency through resource allocation management and better decision-making and overall software quality outcomes when ML models are integrated into defect detection procedures. The research evaluates the operational effects of AI-controlled quality assurance while exploring multiple machine learning models which detect software defects and their implementation in IT management plans. The study presents details about ML's essential role in strategic planning and operational efficiency yet also explains challenges with future directions and benefits of using it for organizational defect management systems. Global software market demands unprecedented code comprehension because it requires enterprises to combine information systems and share knowledge through cooperative platforms *(Gupta et al., 2018).*

Business operations suffer from major risks because of software defects which cause decreased efficiency, security vulnerabilities and loss of customer confidence. Machine learning through ML delivers modern methods to forecast and spot software problems which leads to active quality control practices. The adoption of ML models for defect detection in organizations enables better strategic resource usage while strengthening managerial choices and better total software quality outcomes. This paper examines AI-driven software defect detection through several ML models together with their IT management integration and their business consequences for quality assurance. An investigation presents key challenges and advantages of using ML-based defect management while identifying future growth directions alongside its importance in enterprise operational and strategic programs. The paper examines multiple ML approaches for detecting software defects while discussing how they fit into IT management practices and business implications for strategic development and sustainable software creation.

The major goals of this study includes the following:

i. The study focuses on how ML models can automate the detection and forecasting of software defects.

ii. The study evaluates how AI and ML systems improve strategic decision making processes in organizations as well as allocate resources effectively.

iii. A review of different ML models for software defect detection in various software environments exists as an evaluation objective.

iv. The research studies business advantages of ML technology for software defect management through an evaluation of cost reduction and competitive advantage assessment.

v. The research addresses different implementation obstacles of ML-based defect detection systems by providing recommendations for their resolution.

Literature Review

Machine learning has undergone extensive assessment for detecting software defects. Different research groups studied multiple models alongside their performance results in different operational contexts.

Traditional Approaches to Software Defect Detection

The existing software defect detection process used manual and rule-based testing methods. The research team performed software tests which included both heuristic-based analysis and peer reviews of source code and static code inspection methods. According to *Basili and Rombach (2002)* the combination of these methods delivered no advantage when applied to complex large-scale software development projects due to growing project complexity and insufficient effectiveness as well as errors from human operators. Different datasets processed by *Lessmann et al. (2008)* through their chosen classification techniques resulted in encouraging prediction accuracy while developing a new defect prediction methodology.

Machine Learning in Software Defect Prediction

Research has proven that using decision trees with support vector machines (SVM) and neural networks produce better results than conventional approaches because these ML models demonstrate exceptional abilities in software defect detection. The analytic tool accesses historical defect databases while finding programming related patterns of code that help it detect forthcoming system vulnerabilities before error formation *(Menzies et al., 2006; Hall et al., 2011)*. Combining hyperparameter optimization strategies with the resolution of data inconsistencies in software defect detection models provides an approach to improve their operational capability. NSGA-II together with SMOTE function as methods to increase both model accuracy and generalization capabilities according to *Elshamy et al. (2023)*. Better predictive results result from the technical approach because it bridges traditional defect prediction features with code semantic understanding *(Wang et al., 2020)*.

Strategic Management Implications of Machine Learning in Defect Detection

Current research puts Artificial Intelligence (AI) implementation in strategic management with defect detection applications at its core *(Kending, 2020)*. Execution of AI within strategic management procedures improves decision quality while optimizing resources and operations which enables sustainable business conduct *(Jankovic and Curovic, 2023)*. Companies gain market advantages by implementing AI technology to sort and manage data efficiently according to industry requirements *(Mikalef et al., 2017)* Yet *Wang et al. (2018) and Nishant et al., (2020)* detected higher potential for exploitation in regular business operations, then *Schneider et al.(2019)* discovered comprehensive business process data yielded substantial value for strategic choices and operational enhancement and customer understanding. Organizations gain maximum benefits when they implement complementary technologies while conducting internal research and development *(Lee et al., 2022)*. The study by Kelly *(Kelly et al., 2023)* assessed more than 60 AI adoption elements found in the research literature.

The Role of Machine Learning in Software Defect Detection

The standard software flaw identification methods consisting of code review workflows and rule-driven inspections require enormous manual effort and show both high expense and frequent human mistakes. These limitations in traditionnal methodologies get resolved through ML-based methods which extract knowledge from existing defect records to establish prediction models that detect faults effectively. The application of ML performs three vital functions in defect management systems for software:

Automated Defect Prediction:

Computers utilize ML models to study past software defect records so they can determine patterns for detecting potential errors in new programming code. The systems utilize previous bug reports for training purposes to identify code quality issues and detect potential flaws before system release. The dual advantage of programmed defect prediction systems includes shorter debugging periods together with early detection of faulty code segments to enhance software reliability.

Anomaly Detection:

Machine learning algorithms conduct anomaly detection using multiple approaches to pinpoint irregular patterns that point to problems when software is running and code sequences execute. Unsupervised learning practices support these detection methods by helping them discover code patterns that break normal programming protocols. The abilities of anomaly detection technology exceed traditional testing methods in vulnerability discovery making it required for software testing processes. Users can detect inspection needs through the combination of deep learning and clustering algorithms which operate within ML models.

Automated Root Cause Analysis:

Designers who implement Machine Learning models can conduct automated Root Cause Analysis by allowing analytical tools to detect system defects automatically in order to accelerate defect resolution processes. The stepwise analysis method breaks down past defect data using ML to establish patterns that anticipate likely defective factors in new system problems. Through this capability teams shorten their debugging time which enables them to work on solution implementation while they no longer need to spend time detecting error sources manually.

Risk-Based Testing:

Designers who implement Machine Learning models can conduct automated Root Cause Analysis by allowing analytical tools to detect system defects automatically in order to accelerate defect resolution processes. The stepwise analysis method breaks down past defect data using ML to establish patterns that anticipate likely defective factors in new system problems. Through this capability teams shorten their debugging time which enables them to work on solution implementation while they no longer need to spend time detecting error sources manually.

Adaptive Learning for Continuous Improvement:

ML uses adaptive learning that improves continuously to drive defect detection systems in a period of consistent progress. Software development creates new versions which enable ML models to obtain updated information to develop better detection of emerging system issues. Through continuous learning the systems retain their ability to find new vulnerabilities and maintain compatibility with updated development approaches.

Integration with DevOps and Agile Workflows:

The implementation of ML for defect detection operates effectively inside DevOps automation systems and Agile work management frameworks. The automation in CI/CD pipelines accepts defect prediction and anomaly detection systems to automatically detect and resolve real-time bugs at runtime. Integration brings together improved software development velocity with no negative impact on superior quality and reliability standards.

Natural Language Processing (NLP) for Defect Classification:

The combination of NLP functionality enables ML models to analyze documentation alongside development dialogue alongside bug reports to track defects in an organized manner. The software provides operational advantages through the automation of failure detection and error identification which aids producers in making more reliable software.

Deep Learning for Code Analysis:

Neural network-based deep learning techniques succeed in finding code-based features automatically to analyze software code defects effectively. Neural networks of both types (CNNs and RNNs) examine software code structures combined with execution traces to find potential defects. The models surpass traditional rule-based approaches because they correctly identify the complex patterns which exist within software codebases.

Machine Learning Models for Software Defect Detection

Linear Regression:

The supervised learning process known as linear regression enables predictions of continuous target variables based on multiple input values. The model assumes a linear relationship between the dependent variable y and independent variables x, expressed as:

$$y=\beta_0+\beta_1x_1+\beta_2x_2+\ldots+\beta_nx_n+\epsilon \qquad (1)$$

The x_1, x_2 through x_n variables function as independent variables or predictors or features while the intercept term is β_0 and β_1 through β_n represent the weights that apply for corresponding input attributes to output values with ϵ as the error component *(Montgomery et al., 2021)*.Training occurs through the utilization of the least squares approach when minimizing the total square error. Linear regression remains widespread across various fields including biology and economics and machine learning because it offers easy implementation together with straightforward interpretation *(James et al., 2013)*. Despite being useful for complex cases the three assumptions of linearity, homoscedasticity and the absence of multicollinearity become constraints for its application.

Decision Trees and Random Forest:

Supervised learning through Decision Trees and Random Forest operates as a decision tree method to solve classification and regression tasks. Data segmentation using feature values produces an orderly tree structure through which the data flows until it reaches predictions located at the leaf nodes while internal nodes provide feature-based decisions and branches present potential outcomes. The division process employs defined metrics and values to reduce data set impurities *(Quinlan, 1986)*. A Random Forest is an ensemble learning method that builds multiple decision trees and combines their outputs for better accuracy and generalization. It reduces overfitting by averaging predictions (regression) or using majority voting (classification). Random forests introduce randomness by bootstrapping (sampling data with replacement) and selecting random subsets of features for each tree *(Breiman, 2001)*. This results in robust models resistant to noise and outliers. Random forests outperform single decision trees in many real-world applications, such as medical diagnosis, fraud detection, and recommendation systems, due to their ability to handle high-dimensional data effectively *(Liu et al., 2012).*

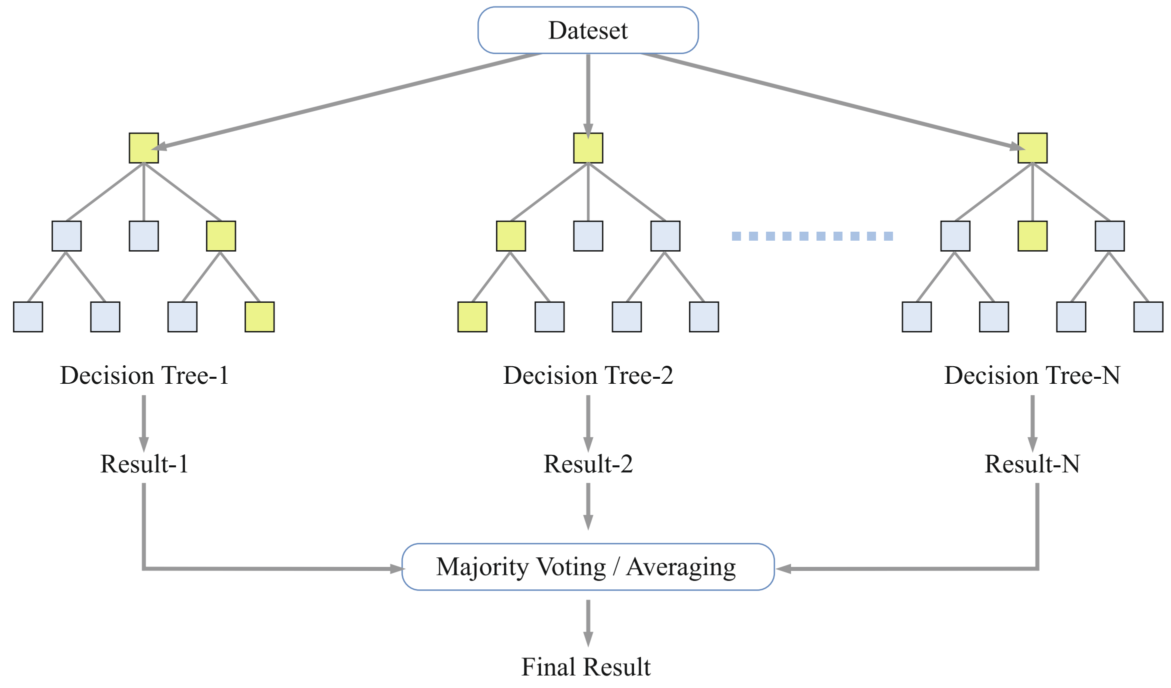

Figure 1: Working of Random Forest by combining results from different Decision Trees
(Source: https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/)

In order to solve issues like distributional shifts and interoperability, recent research has brought novel improvements to decision trees and random forests. *DeLise (2023)* presented "Era Splitting," a method that incorporates era-wise data into decision tree models to improve out-of-distribution generalization, particularly in financial markets. The algorithm assists tree-based models to identify optimal split sites which enhance their performance when processing different data periods thus enhancing their distribution-resistant capabilities. The research of *Ren et al. (2022)* implemented intuitionistic fuzzy decision trees as part of their random forest framework through the development of Intuitionistic Fuzzy Random Forest (IFRF). The ensemble approach uses fuzzy logic strength combined with multiple classifiers to achieve superior classification results than traditional ensemble and fuzzy systems.

Support Vector Machines (SVM):

The Support Vector Machine (SVM) functions as a strong supervised learning method to handle classification together with regression tasks. The algorithm finds the optimum hyperplane in datasets which creates the largest possible distance between different class allocations. The decision-boundary of a classification problem depends heavily on support vectors which represent the points nearest to the hyperplane *(Cortes & Vapnik, 1995)*. The SVM model demonstrates proficiency in two domains: linear classification and non-linear classification along with delivering excellent results in high-dimensional scenarios. The kernel methods of polynomial, radial basis function and sigmoid are used by SVM to transform the initial input dimensions into higher dimensions enabling linear separators to separate data points that cannot be separated by linear boundaries *(Boser et al., 1992).* The high level of resistance exhibited by SVM helps it become the dominant choice for image-recognition tasks together with text-classification and bio-informatics applications *(Hsu et al., 2003).*

The Support Vector Machines (SVMs) have experienced several new applications alongside significant developments during the past years. Support Vector Machines (SVMs) help medical staff utilize MRI for breast cancer detection and PET-CT for lung cancer diagnosis. Bio-informatics makes use of these models to identify genes that cause colon cancer and leukemia. *(Guido et al., 2024)*

Neural Networks:

The deep pattern identification for large-scale defect-prone code-bases is performed by neural networks which derive their design from human brain functions. A neural network contains various synthetic neuron layers that perform data analysis through weighted connection systems. Neural networks rely on activation functions that process input data at their basic components which include an input layer with possible hidden layers and a final output layer *(LeCun et al., 2015).*

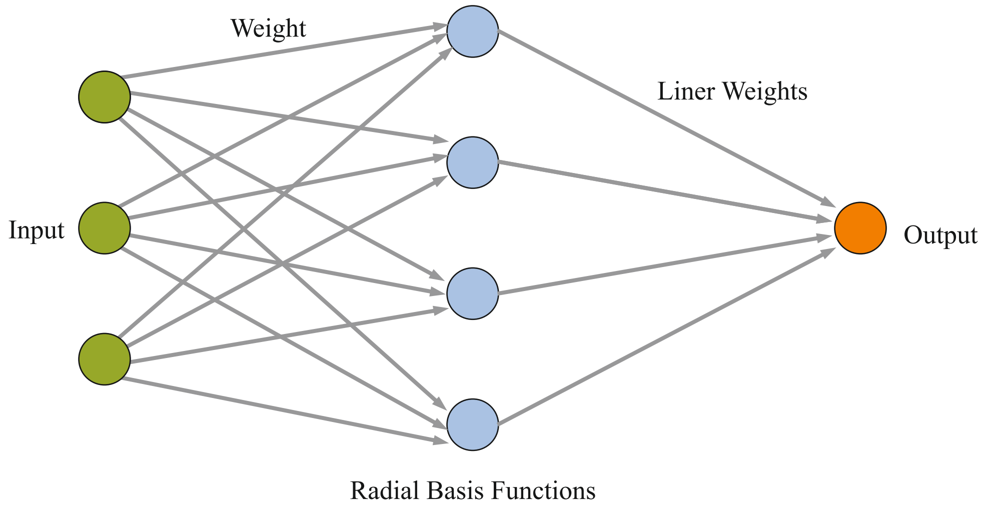

Figure 2: Neural Network (Laxmi and Rohil, 2014).

Deep learning operates as a neural network subset that process numerous hidden layers to extract hierarchical information from raw inputs effectively when optimizing applications such as picture identification and natural language processing and speech recognition *(Krizhevsky et al., 2012).*

Performance Metrics for Evaluating Machine Learning Models in Defect Detection

Strong performance metrics are necessary to evaluate machine learning (ML) defect detection models because they help measure accuracy alongside precision and recall levels and additional vital factors. The metrics enable assessment of model capability to differentiate defective from non-defective items.

Accuracy:

Accuracy measures the proportion of correctly predicted instances out of the total instances:

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+TN+FP+FN)} \qquad (2)$$

- TP (True Positive): Defective items correctly classified as defective.
- TN (True Negative): Non-defective items correctly classified as non-defective.
- FP (False Positive): Non-defective items incorrectly classified as defective (Type I Error).
- FN (False Negative): Defective items incorrectly classified as non-defective (Type II Error).
- The usage of accuracy metric proves beneficial when collections are balanced between defective and non-defective specimens. The solitary use of accuracy proves inadequate when dealing with extremely imbalanced datasets.

Precision:

Precision measures how many predicted defectives are actually defective:

$$\text{Precision} = \frac{TP}{(TP+FP)} \qquad (3)$$

High precision means fewer false positives, reducing unnecessary interventions (e.g., rejecting good products in manufacturing).

Recall:

Recall evaluates the ability to detect all defective items. It is calculated as:

$$\text{Precision} = \frac{TP}{(TP+FN)} \qquad (4)$$

High recall ensures fewer false negatives, preventing defective items from being mistakenly passed as non-defective.

F1-Score:

This metric provides a combined assessment that combines Precision and Recall for a balanced measurement.

$$F1 = 2* \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \qquad (5)$$

The particular metric serves production quality control systems well because it creates minimum errors in the form of false positives and false negatives.

ROC-AUC (Receiver Operating Characteristic – Area Under Curve):

A model can demonstrate its discrimination power using the ROC-AUC metric when operating at different threshold points. Such system strength becomes apparent when its AUC (Area Under Curve) value gets close to 1.0. Use of this method is required when algorithms need to find an optimal point for false negative and false positive rates.

Strategic Management of Machine Learning Based Defect Detection

ML integration into defect detection systems brings transformative value which needs strategic planning along with organizational alignment beyond technological improvements. ML-driven defect detection brings enormous possibilities to enhance quality control methods while cutting operational expenses and maintaining shorter production runs. Business organizations can achieve these advantages by handling the adoption with care while preparing their data adequately and minimizing possible risks and supporting workers through workflow transition.

Adoption Strategies:

The successful implementation of ML integration highly depends on the adoption strategies. AI initiatives need to track their progress with the existing organizational objectives. Organizations need to choose the defect detection applications which matter most followed by success metric definitions that focus on increased defect detection and lowered manual inspection times and product returns. The identification of critical use applications along with setting performance evaluation metrics that include defect detection boost rates and minimum operational downtime represents part of this process. organizations must rely on collaborative teams across different departments while needing executive sponsorship *(Bughin et al., 2017).* The cultural adoption of innovation depends heavily on leadership backing and collaboration between different organizational departments. Organizations run effective solution development through initial testing of small scale projects that help them evolve and expand their work.

Data Management:

Data Management functions as an essential operational foundation that enables the training of dependable ML models. The model requires high-quality datasets for both learning purposes and effective generalization abilities. The model becomes stronger because of diverse datasets that include normal cases and defective ones across different manufacturing conditions.

Risk Mitigation:

Risk Mitigation stands as the key element because AI systems generate errors in their output. False positive errors classify acceptable items as defective thus creating wasteful costs yet false negative errors do not detect genuine product flaws leading to hazardous quality and safety concerns. Threshold tuning enables better precision-recall ratios for companies alongside human-based supervision of uncertain analysis situations. The implementation of bias detection systems during audits promotes model fairness to support various data applications and end user requirements. The combination of model-calibration techniques with bias audits alongside human-controlled verification methods serves as basic approaches to manage potential risks according to *Amershi et al. (2019).* These implemented measures produce trusted automated-decision systems that enable responsible decision making.

Change Management:

AI-system implementation success depends on Change Management to provide training and inspiration about AI system practice for various employees. Most teams experience initial mistrust toward automation when it is introduced for the first time. The adoption of new strategies for progress becomes simpler by giving documentation access and giving training to employees and involving them in project development stages *(Westerman et al., 2014).* The implementation of AI-based defect detection tools faces numerous obstacles when quality-assurance teams needed to perform acceptance during their adoption process. Implementation of training programs paired with documentation methods during interactive workshops will assist people in adapting to such changes. Organizations need an honest dialogue system and employee involvement for creating better trust-building programs between automation and human operators.

Organizations must address critical elements of technology and operational demands and labor requirements when applying strategic approaches to AI-driven defect monitoring systems. Complete achievement of AI potential for defect detection requires organizations to connect their AI initiatives to their business strategy along with strong data systems for implementing change safely throughout the organization. These strategies enable organizations to achieve advanced quality assurance capabilities while establishing a flexible data-oriented method for strategy execution.

Business Implications of Machine Learning Based Software Defect Management

Cost Reduction-

The analytical detection of defects commonly requires workers to spend long durations screening problems manually. The implementation of ML algorithms for test case automation and bug classification and defect prediction allows companies to diminish their requirement for manpower alongside lowering quality assurance duration requirements. Through automation the testing process becomes more efficient which reduces expenses from late-stage modifications while speeding up development phases *(Choudhary et al., 2021).*

Faster Time-to-Market-

The main advantage comes from expedited product delivery. ML models evaluate code patterns and system logs and historical defect records to detect potential bugs before the development stage ends. The warning system activated early in the development process lets developers resolve problems ahead of time which speeds up both testing efforts and release timelines. Ultimately both customer satisfaction growth and faster competition response are achieved through faster product delivery *(Sharma & Sharma, 2022).*

Enhanced Software Quality-

Software Quality receives enhancement through Machine Learning because the technology examines enormous datasets which enable examiners to find elusive defects or repeatedly recurring flaws which testing personnel would have overlooked. Higher accuracy emerges due to this process when detecting and ordering defects. Model learning technology advances software reliability which produces fewer problems for end-users that results in better product reputation and increased brand trust *(Zhou et al., 2019).*

Competitive Advantage-

Takeover opportunities in the market emerge through ML-based defect management systems that allow organizations to outperform competitors successfully. These companies benefit from data insights together with automation technologies which enables them to perform innovation at a quicker pace while releasing superior quality products with maximized resource productivity. The fulfillment of software quality standards for customer satisfaction leads to business advantages in market share expansion and extended growth primarily within finance sectors and healthcare sectors and e-commerce industries.

Conclusion and Future Scope

A support system based on machine learning technologies for defect detection represents an advanced level in IT quality management frameworks and software development systems. By moving away from post-release testing organizations can predict defects early in development thus improving software reliability while saving costs of fixing and supporting the software after release. Information technology department leaders now possess sufficient authority to replace traditional quality assurance procedures by utilizing analytical methods that enhance their defect handling practices. For organizations to achieve full implementation of ML-based defect detection they must invest specifically in multiple vital areas. The achievement of effective defect detection systems by organizations depends on access to high-quality substantial training data. The first step of effective ML model development requires datasets which include precise information alongside diverse content accompanied by appropriate documentation. The development of strong ML algorithms which are both explainable and adaptive leads to better defect detection performance throughout various code bases. To properly handle ethical dangers together with model transparency and performance evaluation organizations must adopt AI governance frameworks.

Several opportunities for future ML application in defect detection extend across various fields. Organizations attain a competitive advantage when they integrate AI-driven defect detection systems into their operational workflows because they achieve better delivery time coupled with superior software quality while establishing improved customer satisfaction. Apart from Machine Learning, organizations need to explore the domain of Deep Learning for software defect detection and also the application of Statistical analysis of the results will support in the adoption of techniques that will greatly enhance the productivity of the organizations in various aspects and also effective resource utilization.

References:

Ali, Z., & Sinha, A. R. (2016). Integrating Ethics in Technical Education for sustainable development. *PURUSHARTHA-A journal of Management*, Ethics and Spirituality, 9(1), 85-97.

Amershi, S., Weld, D., Vorvoreanu, M., Fourney, A., Nushi, B., Collisson, P., ... & Horvitz, E. (2019, May). *Guidelines for human-AI interaction*. In Proceedings of the 2019 chi conference on human factors in computing systems (pp. 1-13).

Basili, V. R., & Rombach, H. D. (2002). The TAME project: Towards improvement-oriented software environments. *IEEE Transactions on software engineering*, 14(6), 758-773.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992, July). A training algorithm for optimal margin classifiers. *In Proceedings of the fifth annual workshop on Computational learning theory* (pp. 144-152).

Botla, L., & Kondur, H. (2018). Socio technical systems of a company: the dimensionality of socio technical systems. PURUSHARTHA-A journal of Management, *Ethics and Spirituality*, 11(1), 24-38.

Bughin, J., Hazan, E., Sree Ramaswamy, P., DC, W., & Chu, M. (2017). *Artificial intelligence the next digital frontier.*

Capretz, L. F., & Lee, P. A. (1992). Reusability and life cycle issues within an object-oriented methodology. *Computing Laboratory Technical Report Series.*

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning,* 20, 273-297.

DeLise, T. (2023). Era Splitting: Invariant Learning for Decision Trees. arXiv preprint arXiv:2309.14496.

El Emam, K., Benlarbi, S., Goel, N., & Rai, S. N. (2001). Comparing case-based reasoning classifiers for predicting high risk software components. *Journal of Systems and Software*, 55(3), 301-320.

Elshamy, N., AbouElenen, A., & Elmougy, S. (2023). Automatic Detection of Software Defects based on Machine Learning. *International Journal of Advanced Computer Science and Applications*. https://doi.org/10.14569/ijacsa.2023.0140340.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1, No. 2). Cambridge: MIT press.

Guido, R., Ferrisi, S., Lofaro, D., & Conforti, D. (2024). An overview on the advancements of support vector machine models in healthcare applications: a review. *Information*, 15(4), 235.

Gupta, R. G., Mazumdar, B. D., & Yadav, K. (2018). Agent Mediated Code Comprehension: A Cognitive Challenge. *PURUSHARTHA-A journal of Management*, Ethics and Spirituality, 11(2), 49-59.

Hall, T., Beecham, S., Bowes, D., Gray, D., & Counsell, S. (2011). A systematic literature review on fault prediction performance in software engineering. *IEEE Transactions on Software Engineering*, 38(6), 1276-1304.

Hsu, C. W., Chang, C. C., & Lin, C. J. (2003). *A practical guide to support vector classification.*

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, No. 1). *New York: springer.*

Jankovic, S., & Curovic, D. (2023). Strategic Integration of Artificial Intelligence for Sustainable Businesses: Implications for Data Management and Human User Engagement in the Digital Era. *Sustainability*. https://doi.org/10.3390/su152115208.

Keding, C. (2020). Understanding the interplay of artificial intelligence and strategic management: four decades of research in review. *Management Review Quarterly*, 71, 91 - 134. https://doi.org/10.1007/s11301-020-00181-x.

Kelly, S., Kaye, S. A., & Oviedo-Trespalacios, O. (2023). What factors contribute to the acceptance of artificial intelligence? A systematic review. *Telematics and Informatics*, 77, 101925.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

Kumar, S., & Bhandarker, A. (2020). Experiential learning and its efficacy in management education. *PURUSHARTHA-A journal of Management, Ethics and Spirituality*, 13(1), 35-55.

Laxmi, V., & Rohil, H. (2014). License plate recognition system using back propagation neural network. *International Journal of Computer Applications*, 99(8), 29-37.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning*. nature, 521(7553), 436-444.

Lee, Y. S., Kim, T., Choi, S., & Kim, W. (2022). When does AI pay off? AI-adoption intensity, complementary investments, and R&D strategy. *Technovation*, 118, 102590.

Lessmann, S., Baesens, B., Mues, C., & Pietsch, S. (2008). Benchmarking classification models for software defect prediction: A proposed framework and novel findings. *IEEE transactions on software engineering*, 34(4), 485-496.

Menzies, T., Greenwald, J., & Frank, A. (2006). Data mining static code attributes to learn defect predictors. *IEEE transactions on software engineering*, 33(1), 2-13.

Mikalef, P., Framnes, V. A., Danielsen, F., Krogstie, J., & Olsen, D. (2017). Big data analytics capability: antecedents and business value.

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2021). Introduction to linear regression analysis. *John Wiley & Sons*.

Nishant, R., Kennedy, M., & Corbett, J. (2020). Artificial intelligence for sustainability: Challenges, opportunities, and a research agenda. *International journal of information management,* 53, 102104.

Padma, R. N., & Aravamudhan, N. R. (2021). The Impact of Learning and Culture on Organizational Identification: An Indian Case Study. *PURUSHARTHA-A journal of Management, Ethics and Spirituality*, 14(2), 44-56.

Ren, Y., Zhu, X., Bai, K., & Zhang, R. (2022). A new random forest ensemble of intuitionistic fuzzy decision trees. *IEEE Transactions on Fuzzy Systems*, 31(5), 1729-1741.

Schneider, S., & Leyer, M. (2019). Me or information technology? Adoption of artificial intelligence in the delegation of personal strategic decisions. *Managerial and Decision Economics*, 40(3), 223-231.

Singh, A. K., & Sabharwal, S. (2020). Talent management: An empirical analysis of its antecedents and consequences applying structural equation modeling. *PURUSHARTHA-A journal of Management, Ethics and Spirituality*, 13(2), 1-16.

Traymbak, S., Kumar, P., & Jha, A. N. (2016). Examining moderating effects of gender between role stress and job satisfaction among software employees. *PURUSHARTHA-A journal of Management, Ethics and Spirituality*, 9(2), 35-45.

Wang, S., Liu, T., Nam, J., & Tan, L. (2020). Deep Semantic Feature Learning for Software Defect Prediction. *IEEE Transactions on Software Engineering*, 46, 1267-1293. https://doi.org/10.1109/TSE.2018.2877612.

Wang, Y., Kung, L., & Byrd, T. A. (2018). Big data analytics: Understanding its capabilities and potential benefits for healthcare organizations. *Technological forecasting and social change*, 126, 3-13.

Westerman, G., Bonnet, D., & McAfee, A. (2014). Leading digital: Turning technology into business transformation. *Harvard Business Press.*