eISSN 2783-6851

NEW TRENDS in COMPUTER SCIENCES

2024
Volume 2
Issue 1
Pages 46–56
https://doi.org/10.3846/ntcs.2024.21574

THE EFFICIENCY OF MACHINE LEARNING ALGORITHMS IN CLASSIFYING NON-FUNCTIONAL REQUIREMENTS

Milda MACIEJAUSKAITĖ, Jolanta MILIAUSKAITĖ

Department of Information Systems, Faculty of Fundamental Sciences, Vilnius Gediminas Technical University, Vilnius, Lithuania

Article History:
- received 21 May 2024
- accepted 30 May 2024

Abstract. Machine learning (ML) algorithms are more and more widely applied in various types of systems, so the research related to them is also increasing. One of the areas of research under consideration is the classification of non-functional requirements (NFRs) using ML algorithms. This area of research is important because the automatic classification of NFRs using high-performance ML algorithms and corresponding features helps requirements engineers classify non-functional requirements more accurately. This paper examines ML algorithms suitable for solving classification problems and their effectiveness in classifying non-functional requirements. Based on the described stages of the research methodology ML algorithms models were compared using the accuracy, precision, recall, and F-score metrics. A majority voting classifier model was created using Support Vector Machine, Naïve Bayes and K Nearest Neighbor Algorithm algorithms. After K-Fold cross validation were obtained these results: accuracy – 0.710 (scale from 0 to 1), precision – 0.845, recall – 0.814 and F-score – 0.815.

Keywords: machine learning, non-functional requirements, classification, support vector machine, ensemble models, K-Fold cross validation.

Corresponding author. E-mail: *milda.maciejauskaite@stud.vilniustech.lt*

1. Introduction

Machine learning (ML) is a branch of artificial intelligence (AI) and computer science that uses large amounts of data to create complex predictions and decision-making systems that would otherwise be difficult to achieve. Decision-making systems based on ML are increasingly being used in decisions about bank loans (Karthiban et al., 2019), employment (Imam & Ananda, 2022), clinical trials (Miller et al., 2023) and in many other areas. The success of ML-enabled systems depends on the properties of the ML solutions (like performance, transparency, maintainability, interoperability, etc.), which are known as non-functional properties in the domain of requirements engineering. In addition, ML systems' non-functional requirements (NFRs) may differ in their definition, measurement, scope and importance in comparative meaning (Habibullah & Horkoff, 2021). Our understanding of these aspects is inadequate compared to our knowledge of NFRs in traditional domains.

In addition, the task of classifying requirements is distinguished in software engineering because the manual process of classifying non-functional requirements is subjective, potentially erroneous, complex, and time-consuming. With ML, this process would reduce these disadvantages. This area of research is relevant because the automatic classification of NFRs using high-performance ML algorithms and corresponding features helps requirements engineers classify non-functional requirements more accurately (Khurshid et al., 2022). Therefore, this study shows a comparison of ML algorithms to the problem of non-functional requirements classification to answer the question: "Which works best for classifying software requirements into non-functional requirements?" and "Which Machine Learning Algorithm provides the best performance for the requirements classification task?".

The purpose of this work is to explore the performance of ML algorithms for classification of non-functional requirements and to propose a solution to improve the accuracy of the classifier. This work result is to suggest a more accurate method for NRF classification based on ML. The novelty of the research is as follows:

- The study highlights the need and problem of classification of non-functional requirements using ML algorithms.
- The study focuses on using machine learning algorithms to classify non-functional requirements, a growing research area.
- The results of the study show a more accurate non-functional requirement classification method based on ML.

The rest of this paper is structured as follows. Section 2 presents the related works on the topic of classification of non-functional requirements by ML algorithms. Section 3 presents preliminaries related to the research topic. Section 4 shows the methodology and use case. Section 5 presents model accuracy improvement. Finally, Section 6 provides the conclusion and the outlines for future works.

2. Related works

This section presents related work on ML algorithms for classifying non-functional requirements. The aim of this section is to determine which ML algorithms researchers consider suitable for the classification of NFRs. The work on the classification of non-functional requirements using AI techniques relevant to this analysis is summarized in Table 1. The content of this table consists of the following columns: 1) reference (Ref.) of the scientific paper; 2) ML algorithm that showed the highest accuracy; 3) data set from which the requirements for the analyses were taken; 4) instruments used in the research; 5) results, i.e. best ML algorithm for NFR classification; 6) accuracy obtained with the best ML algorithm for NFR classification.

Analyzing the data in Table 1 shows that the authors reported these ML methods as the most accurate for the classification of NFRs: Support Vector Machine (SVM), Naïve Bayes (NB), K Nearest Neighbor Algorithm (KNN).

Table 1. Analysis of studies on NFR classification by ML algorithms

Ref. (1)	ML algorithm with highest accuracy (2)	Data set (3)	Instruments (4)	Findings (5)	Accuracy (6)
Kurtanović et al. (2017)	Support Vector Machine (SVM)	625 requirements data set	Natural language (NL) toolkit NLTK 1, Stanford Parser 2 and ML library SciKit 3	A supervised ML approach using metadata, lexical and syntactic features based on the SVM algorithm.	up to ~92%.

End of Table 2

Ref. (1)	ML algorithm with highest accuracy (2)	Data set (3)	Instruments (4)	Findings (5)	Accuracy (6)
Baker et al. (2019)	Convolutional Neural Network (CNN)	914 requirements from Conference, "PROMISE" data set	NL toolkit NLTK, "NumPy" library, "TensorFlow" library, "TensorFlow Adam Optimizer"	Convolutional Neural Network model.	Accuracy 82–94%, recall 76–97%, F-score ranging 82–92%.
Abad et al. (2017)	Naïve Bayes (NB)	"PROMISE" data set	LingPipe NLP Toolkit	The Naïve Bayes classifier is the best in classifying NFRs into subcategories.	up to 90%
Alashqar (2017)	SVM	"PROMISE" data set	Python programming	ML classifier determines the performance using NFR classifications. The SVM outperforms classifiers in results.	With all data set: precision 71.5%, recall 72.2%, F-score ranging 71.3%.
Binkhonain and Zhao (2019)	SVM and Multinomial NB	"Science Direct", "IEEExplore", "Springer Link", "ACM".	Excel	SVM and Multinomial NB – shows the best classification results among supervised ML classifiers.	–
Khurshid et al. (2022)	K Nearest Neighbor Algorithm (KNN)	"PROMISE" data set; set of 104 requirements	Anaconda Tool	The hybrid KNN Algorithm rule-based ML algorithm for classification .	up to 75.9% with authors data set.
Haque et al. (2019)	Stochastic Gradient Descent Support Vector Machines	"PROMISE" data set	"Intel core i7", 32 GB RAM, "Ubuntu 16.06"	The Stochastic Gradient Descent SVMs classifier achieves the best results.	Precision 66%, recall 61%, F-score 61%, accuracy 76%.

3. Preliminaries

3.1. ML algorithms

ML systems aim to learn from data (Mahesh, 2020) to automate the process of creating an analytical model and solving related tasks (Janiesch et al., 2021). However, the researchers emphasize that there is no single ML algorithm that can best solve different problems. The algorithm used depends on the problem to be solved, its type, the number of variables, the most appropriate model, and other factors (Mahesh, 2020).

ML algorithms are divided into two types of learning: supervised and unsupervised learning (Carta, 2022). The fundamental difference between these two learning algorithms is whether the examples given to the learning algorithm are labelled or not (Bao et al., 2022).

According to Sarker et al. (2020), supervised learning occurs when specific goals are to be achieved with a given set of inputs, i.e., a task-driven approach. In this paper, supervised learning is best suited for NRF classification as a labelled dataset is provided for research. Table 2 lists the ML algorithms that are further used to evaluate the performance of the ML algorithms in NFR classification. These algorithms were selected after analyzing the scientific literature related to the subject area. After various research was analyzed, the conclusion was that these algorithms perform well in NFR classification tasks. Also, in the process of choosing algorithms, one of the most important properties was the algorithms that are characterized by accuracy.

Table 2. Supervised ML algorithms for classification.

ML algorithm	Properties	Solved problems
Decision Tree (DT)	A decision tree models decision logic (Uddin et al., 2019)	Regression and classification problems (Rajaguru & Chakravarthy, 2019)
Random Forest (RF)	Considered a reliable algorithm that provides accurate predictions (Mohd et al., 2019)	Classification and regression problems (Sruthi, 2023)
Support Vector Machine (SVM)	It is considered a powerful tool to make accurate predictions (Ho et al., 2021)	Regression and classification problems (Yang & Shami, 2020)
Naïve Bayes (NB)	Its simplicity allows all functions to contribute equally to the final solution (Ibrahim & Abdulazeez, 2021)	Popular statistical method for spam filtering (Wickramasinghe & Kalutarage, 2021)
K Nearest Neighbor Algorithm (KNN)	A simple ML algorithm used to classify data points by calculating distances between different data points (Yang & Shami, 2020)	This algorithm is used to solve classification and regression problems (Ibrahim & Abdulazeez, 2021)
Logistic regression (LR)	Logistic regression is classified as a statistical ML method (Rymarczyk et al., 2019)	Binary classification tasks by predicting the probability (Kanade, 2022)

Thus, each ML algorithm has its own characteristics and is best suited to tackling certain problems listed in Table 2. Next, models are developed with the analyzed ML algorithms to verify the suitability of the algorithms for NFR classification.

3.2. Measurement metrics

Below, you will find the metrics against which the study results are measured.

Accuracy shows the number of correctly classified data units out of the total number of data units (Harikrishnan, 2019). The accuracy calculation formula is given (Silwal, 2022), (1):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{1}$$

where TP – is the number of correctly classified requirements, TN – is the number of true negative results, FP – is the number of falsely recognized as correct requirements, and FN – is the number of incorrectly classified requirements.

Precision is defined as the number of correctly identified true positives divided by the sum of the number of correctly identified true and false positive results (Koehrsen, 2018). The precision calculation formula is presented (Binkhonain & Zhao, 2019), (2):

$$Precision = \frac{TP}{TP + FP}, \tag{2}$$

where *TP* – is the number of correctly classified requirements and *FP* – is the number of falsely recognized correct requirements.

Recall measures the percentage of correctly classified NFRs (Binkhonain & Zhao, 2019) and can be considered the ability of a model to find all data points of interest in that data set (Koehrsen, 2018). The formula for calculating recall is presented (Binkhonain & Zhao, 2019), (3):

$$Recall = \frac{TP}{TP + FN}, \tag{3}$$

where *TP* – is the number of correctly classified requirements, and *FN* – is the number of incorrectly classified requirements.

F-score considers both precision and recall, it is the harmonic mean of precision and recall (Ghoneim, 2019), and it is calculated according to the formula (Shung, 2018), (4):

$$F\text{-}score = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \tag{4}$$

4. Methodology and use case for Classifying Non-Functional Requirements

Based on Haque et al. (2019), Figure 1 shows a principal schema for analyzing ML models. The models are created using ML algorithms for NFR classification.

The steps involved in analyzing and fitting ML models are as follows:

- *Data processing*. As with most ML projects with big data, data preprocessing is a necessary first step (Baker et al., 2019). This is the preparation of the Kaggle platform data set for further classification. Used Kaggle data set consists of 976 requirements, 346 of which are non-functional and divided into 11 categories (Shukla, 2023). This data set was last modified in February 2023. The requirements for the Kaggle data set consist

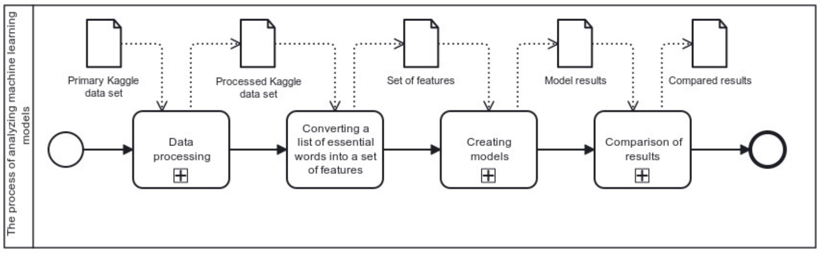

The process of analyzing machine learning models
Primary Kaggle data set
Processed Kaggle data set
Set of features
Model results
Compared results
Data processing
Converting a list of essential words into a set of features
Creating models
Comparison of results

Figure 1. Principal schema of analyzing ML models

of fault tolerance (10 requirements), maintainability (17 requirements), performance (54 requirements), portability (2 requirements), scalability (21 requirements), security (56 requirements), usability (63 requirements), legal and licensing (10 requirements), availability (21 requirements), look and feel (34 requirements), operability (58 requirements) NRF classes. Special characters are removed, as well as uppercase letters are converted to lowercase to classify the data set as simply as possible, words that complicate the algorithm are removed (such as "a", "an", etc.) and tokenization is performed – this divides the text into smaller parts (Haque et al., 2019). Python programming language, a Google Colab software tool, were used for data processing.

- *Converting a list of essential words into a set of features.* This is changing the text so that it can be understood by ML algorithms. The data is divided into groups and used for model training and requirements classification. The data is automatically partitioned as follows: 80% of the data set is for training, and 20% of the data set is for testing. The data divided into training and testing sets are further vectorized using Term Frequency and Inverse document Frequency (TF-IDF). Other vectorization techniques such as Bidirectional Encoder Representations from Transformers (BERT) have been tried, but the best results have been obtained using TF-IDF.
- *Creating models.* By applying DT (Decision Tree), RF (Random Forest), SVM (Support Vector Machine), NB (Naïve Bayes), KNN (K Nearest Neighbor) and LR (Logistic Regression) ML algorithms for data classification, ML models are created, which are trained to classify NFRs, and their results are tested.
- *Comparison of results.* All the results of the obtained models were compared according to accuracy, precision, recall and F-score results. Table 3 compares the achieved models results.

Table 3. Comparisons of the accuracy, precision, recall and F-score results of the models

	Accuracy	Precision	Recall	F-score
DT	0.529	0.539	0.529	0.512
RF	0.7	0.738	0.7	0.696
SVM	0.814	0.77	0.814	0.788
NB	0.786	0.811	0.786	0.785
KNN	0.743	0.769	0.743	0.735
LR	0.757	0.717	0.757	0.727

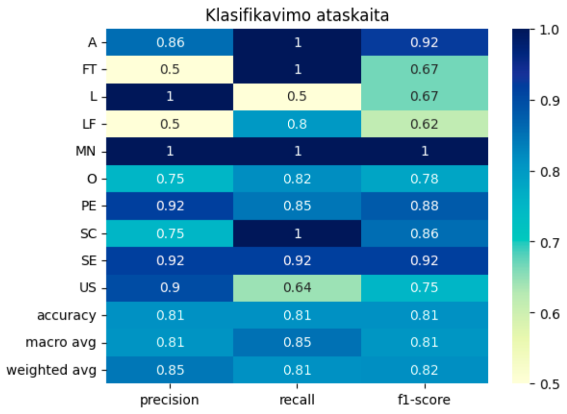
The model of the support vector method provided the best accuracy results, while the accuracy of the decision tree reached only 0.529, which was the lowest among those examined. The best precision results were obtained using the simple Naïve Bayes classifier – 0.811. The rest of the models gave similar rates, except for the decision tree model, which only had a rate of 0.539. Analysing recall results, the model of the support vector method achieved the best results. The best F-score was obtained using the support vector method (0.788). Also, the simple Naïve Bayes classifier gave a similar F-score of 0.785. The decision tree model produced the worst F-score.

Summarizing the results of all models, the model of the support vector method achieved the best indicators, but the results of other models, except for the decision tree model, can be considered good. By industry standards, results are considered good when they are between 70% and 90%. Anything above 70% is acceptable as valuable data output for the model (Hendricks, n.d.). Meanwhile, the indicators of the decision tree model do not fall within the specified range.

5. Improvement of ML model accuracy

To create a more accurate method for NRF classification, it is worth considering combining ML algorithms. Ensemble models can be used for this. Combining different sets of individual ML models can improve the stability of the overall model, resulting in more accurate predictions (Nelson, 2020). One of the ensemble ML techniques is majority voting classifier. To make a final prediction, voting classifier combines the predictions of several individual classifiers (Kumar, 2023). Majority voting classifier is commonly used for classification problems (Singh, 2023). The advantage of majority voting is that it reduces the prediction error rate (Bajaj, 2023).

In order to improve the accuracy of the NFR classification, it was found that the best results were achieved with the SVM, NB and KNN algorithms using majority voting classifier. Figure 2 shows the obtained model classification report.



	precision	recall	f1-score
A	0.86	1	0.92
FT	0.5	1	0.67
L	1	0.5	0.67
LF	0.5	0.8	0.62
MN	1	1	1
O	0.75	0.82	0.78
PE	0.92	0.85	0.88
SC	0.75	1	0.86
SE	0.92	0.92	0.92
US	0.9	0.64	0.75
accuracy	0.81	0.81	0.81
macro avg	0.81	0.85	0.81
weighted avg	0.85	0.81	0.82

Figure 2. The majority voting classifier model classification report

The results of the majority voting classifier provided better indicators and ML algorithms in precision and F-score. However, when analysing precision and recall, the SVM model produced the same results as the majority voting classifier. Overall, the model showed better results in the classification of NFR. In order to determine the stability of the model, K-Fold cross validation of the model was performed. 10 subsets were used for model stability assessment. Table 4 shows the results of each of the subsets and their average.

Results: accuracy between 0.559 and 0.829 (from 0 to 1), precision between 0.617 and 0.861, recall between 0.559 and 0.829 and F-score between 0.525 and 0.817. Figure 3 presents an average of 10 subsets confusion matrix of majority voting classifier model after K-Fold cross validation.

Table 4. Model subsets results

	Accuracy	Precision	Recall	F-score
1 subset	0.771	0.815	0.771	0.751
2 subset	0.686	0.749	0.686	0.644
3 subset	0.800	0.861	0.800	0.806
4 subset	0.743	0.800	0.743	0.739
5 subset	0.657	0.712	0.657	0.654
6 subset	0.829	0.845	0.829	0.817
7 subset	0.559	0.617	0.559	0.525
8 subset	0.647	0.736	0.647	0.636
9 subset	0.676	0.709	0.676	0.645
10 subset	0.735	0.772	0.735	0.722
Average	0.710	0.762	0.710	0.694

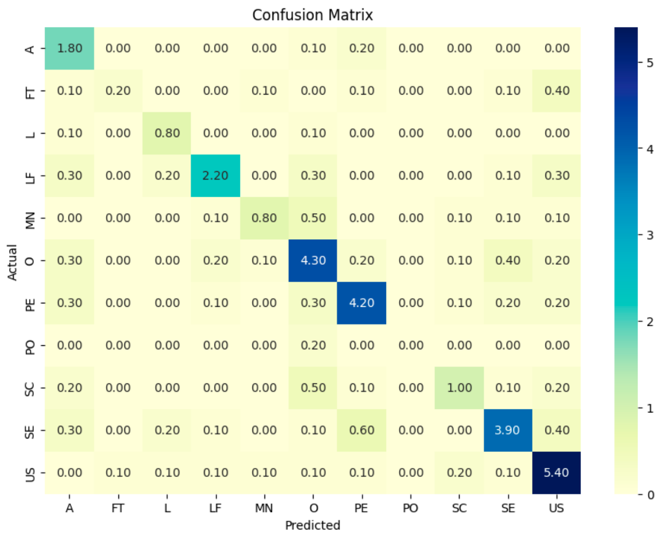


Figure 3. The majority voting classifier model confusion matrix

Confusion matrix presents how many classes were predicted as actual. In order to compare the results of the new majority model with the results of the best performing SVC algorithm model, after cross-validation, Table 5 is presented.

Table 5. Model results after cross-validation.

	Accuracy	Precision	Recall	F-score
Majority voting classifier	0.710	0.762	0.710	0.694
SVC	0.693	0.683	0.693	0.661

Comparing the majority voting classifier model to the best-performing SVC model up to that point, we see a 1.7% increase in accuracy, a 7.9% increase in precision, a 1.7% increase in recall, and a 3.3% increase in F-score.

6. Conclusions

After analyzing the articles on the topic of ML algorithms for classifying non-functional requirements, it was found that one of the research directions is the classification of NFRs using ML algorithms. After deeper analyzing this, it is found that most researchers use SVM as the most accurate algorithm for classifying NFRs, but the opinions of researchers differ.

With the selected Kaggle data set and suitable tools, the development of ML methods was carried out, and performance indicators of models were obtained, based on which the accuracy of ML algorithms in classifying NFRs was compared. Among the ML algorithms used, the best results were provided by SVM, where model accuracy – 0.814 (on a scale from 0 to 1), precision – 0.770, recall – 0.814 and F-score – 0.788 were achieved.

After conducting experiments combining a ML algorithm, a more accurate NFRs classification method based on ML was obtained. The best results were obtained using majority voting classifier with SVM, NB, KNN algorithms, where accuracy – 0.814 (on a scale from 0 to 1), precision – 0.845, recall – 0.814 and F-score – 0.815. K-Fold cross validation for majority voting classifier with SVM, NB, KNN were performed. Model accuracy – 0.710, precision – 0.845, recall – 0.814 and F-score – 0.815.

The future research direction of this paper can be improving the results achieved by the non-functional requirements classification model by using hybrid algorithm or explore solutions contributed from other perspectives and AI techniques, such as Search-based Software Engineering.

References

Abad, Z. S., Karras, O., Ghazi, P., Glinz, M., Ruhe, G., & Schneider, K. (2017). What works better? A study of classifying requirements. In *2017 IEEE 25th International Requirements Engineering Conference* (*RE*), (pp. 496–501). Lisbon. https://doi.org/10.1109/RE.2017.36

Alashqar, A. M. (2022). Studying the commonalities, mappings and relationships between non-functional requirements using machine learning. *Science of Computer Programming*, *218*, Article 102806. https://doi.org/10.1016/j.scico.2022.102806

Bajaj, A. (2023, April 27). *Ensemble models: How to make better predictions by combining multiple models with Python codes* (*explained*). https://aryanbajaj13.medium.com/ensemble-models-how-to-make-better-predictions-by-combining-multiple-models-with-python-codes-6ac54403414e

Baker, C., Deng, L., Chakraborty, S., & Dehlinger, J. (2019). Automatic multi-class non-functional software requirements classification using neural networks. In *2019 IEEE 43rd Annual Computer Software and Applications Conference* (*COMPSAC*), (pp. 610–615). Milwaukee. https://doi.org/10.1109/COMPSAC.2019.10275

Bao, W., Lianju, N., & Yue, K. (2019). Integration of unsupervised and supervised machine learning algorithms for credit risk assessment. *Expert Systems with Applications*, *128*, 301–315. https://doi.org/10.1016/j.eswa.2019.02.033

Binkhonain, M., & Zhao, L. (2019). A review of machine learning algorithms for identification and classification of non-functional requirements. *Expert Systems with Applications*. https://doi.org/10.1016/j.eswa.2019.02.031

Carta, S. (2022). *What is machine learning?* Wiley-Blackwell. https://doi.org/10.1002/9781119815075.ch18

Ghoneim, S. (2019, April 02). *Accuracy, recall, precision, f-score & specificity, which to optimize on?* https://medium.com/towards-data-science/accuracy-recall-precision-f-score-specificity-which-to-optimize-on-867d3f11124

Habibullah, K. M., & Horkoff, J. (2021, September). Non-functional requirements for machine learning: Understanding current use and challenges in industry. In *2021 IEEE 29th International Requirements Engineering Conference* (*RE*) (pp. 13–23). IEEE. https://doi.org/10.1109/RE51729.2021.00009

Haque, M. A., Rahman, M. A., & Siddik, M. S. (2019). Non-functional requirements classification with feature extraction and machine learning: An empirical study. In *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology* (*ICASERT*), (pp. 1–5). Dhaka, Bangladesh. https://doi.org/10.1109/ICASERT.2019.8934499

Harikrishnan, N. B. (2019, December 10). *Confusion matrix, accuracy, precision, recall, F1 score*. https://medium.com/analytics-vidhya/confusion-matrix-accuracy-precision-recall-f1-score-ade299cf63cd

Hendricks, R. (n.d.). *What is a good accuracy score in Machine Learning?* https://deepchecks.com/question/what-is-a-good-accuracy-score-in-machine-learning/

Ho, W. K., Tang, B.-S., & Wong, S. W. (2021). Predicting property prices with machine learning algorithms. *Journal of Property Research*, *38*(1), 48–70. https://doi.org/10.1080/09599916.2020.1832558

Ibrahim, I. M., & Abdulazeez, A. M. (2021). The role of machine learning algorithms for diagnosing diseases. *Journal of Applied Science and Technology Trends* (*JASTT*), *2*(1), 10–19. https://doi.org/10.38094/jastt20179

Imam, T., & Ananda, J. (2022). Machine learning for characterizing growth in tourism employment in developing economies: an assessment of tourism employment in Sri Lanka. *Current Issues in Tourism*, *25*(16), 2695–2716. https://doi.org/10.1080/13683500.2021.1991895

Janiesch, C., Zschech, P., & Heinrich, K. (2021). Machine learning and deep learning. *Electronic Markets*, *31*, 685–695. https://doi.org/10.1007/s12525-021-00475-2

Kanade, V. (2022). *What is logistic regression? Equation, assumptions, types, and best practices*. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/

Karthiban, R., Ambika, M., & Kannammal, K. E. (2019, January). A review on machine learning classification technique for bank loan approval. In *2019 International Conference on Computer Communication and Informatics* (*ICCCI*) (pp. 1–6). IEEE. https://doi.org/10.1109/ICCCI.2019.8822014

Khurshid, I., Imtiaz, S., Boulila, W., Khan, Z., & Abbasi, A. (2022). Classification of non-functional requirements from IoT oriented healthcare requirement document. *Frontiers Public Health*, *10,* Article 860536. https://doi.org/10.3389/fpubh.2022.860536

Koehrsen, W. (2018, March 03). *Beyond accuracy: Precision and recall*. https://medium.com/towards-data-science/beyond-accuracy-precision-and-recall-3da06bea9f6c

Kumar, R. (2023, August 12). *VotingClassifier*. https://medium.com/@ranjankumar_29097/votingclassifier-3f85ba8e4580

Kurtanović, Z., & Maalej, W. (2017). Automatically classifying functional and non-functional requirements using supervised machine learning. In *2017 IEEE 25th International Requirements Engineering Conference* (*RE*), (pp. 490–495). Lisbon. https://doi.org/10.1109/RE.2017.82

Mahesh, B. (2020). Machine learning algorithms – A review. *International Journal of Science and Research* (*IJSR*), *9*(1), 381–386.

Miller, M. I., Shih, L. C., & Kolachalama, V. B. (2023). Machine learning in clinical trials: A primer with applications to neurology. *Neurotherapeutics*, *20*(4), 1066–1080. https://doi.org/10.1007/s13311-023-01384-2

Mohd, T., Masrom, S., & Johari, N. (2019). Machine learning housing price prediction in Petaling Jaya, Selangor, Malaysia. *International Journal of Recent Technology and Engineering*, *8*(2S11), 542–546. https://doi.org/10.35940/ijrte.B1084.0982S1119

Nelson, D. (2020, October 26). *Kas yra ansamblinis mokymasis?* https://www.unite.ai/lt/kas-yra-ansamblinis-mokymasis/

Rajaguru, H., & Chakravarthy, S. (2019). Analysis of decision tree and K-Nearest neighbor algorithm in the classification of breast cancer. *Asian Pacific Journal Cancer Prevention*, *20*(12), 3777–3781. https://doi.org/10.31557/APJCP.2019.20.12.3777

Rymarczyk, T., Kozłowski, E., Kłosowski, G., & Niderla, K. (2019). Logistic regression for machine learning in process tomography. *Sensors*, *19*(15), Article 3400. https://doi.org/10.3390/s19153400

Sarker, I. H., Kayes, A. S., Badsha, S., Alqahtani, H., Watters, P., & Ng, A. (2020). Cybersecurity data science: An overview from machine learning perspective. *Journal of Big Data*, *7*, Article 41. https://doi.org/10.1186/s40537-020-00318-5

Shukla, V. (2023, February). *Software requirements dataset*. https://www.kaggle.com/datasets/iamvai-bhav100/software-requirements-dataset?resource=download

Shung, K. P. (2018, March 15). *Accuracy, precision, recall or F1?* https://medium.com/towards-data-scien-ce/accuracy-precision-recall-or-f1-331fb37c5cb9

Silwal, D. (2022, January 05). *Confusion matrix, accuracy, precision, recall & F1 score: Interpretation of performance measures*. https://www.linkedin.com/pulse/confusion-matrix-accuracy-precision-re-call-f1-score-measures-silwal

Singh, A. (2023, November 22). *A comprehensive guide to ensemble learning (with Python codes)*. https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/

Sruthi, E. R. (2023, April 26). *Understand random forest algorithms with examples (updated 2023)*. https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

Uddin, S., Khan, A., Hossain, M. E., & Moni, M. A. (2019). Comparing different supervised machine lear-ning algorithms for disease prediction. *BMC Medical Informatics and Decision Making*, *19*, Article 281. https://doi.org/10.1186/s12911-019-1004-8

Wickramasinghe, I., & Kalutarage, H. (2021). Naive Bayes: Applications, variations and vulnerabilities: a review of literature with code snippets for implementation. *Soft Computing*, *25*, 2277–2293. https://doi.org/10.1007/s00500-020-05297-6

Yang, L., & Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, *415*, 295–316. https://doi.org/10.1016/j.neucom.2020.07.061