SEMARAK ILMU PUBLISHING
202103268166(003316878-P)

Journal of Advanced Research in Applied Sciences and Engineering Technology

Journal homepage:
https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index
ISSN: 2462-1943

Comparative Performance of Machine Learning Algorithms for Predicting Future Committer in Blockchain Projects

Alawiyah Abd Wahab[1], Huda H. Ibrahim[2], Shehu M. SarkinTudu[2,*], Maslinda Mohd. Nadzir[2], Zhamri Che Ani[2]

[1] School of Computing, Universiti Utara Malaysia. 06010 Sintok, Kedah, Malaysia
[2] Institute for Advanced and Smart Digital Opportunities, School of Computing, Universiti Utara Malaysia. 06010 Sintok, Kedah, Malaysia

ARTICLE INFO

Article history:
Received 29 March 2023
Received in revised form 28 May 2023
Accepted 18 August 2023
Available online 5 December 2023

Keywords:

Developer Assessment; Committer Prediction; System Integration; Project Desertion; Developer Involvement; XGBoost

ABSTRACT

The success of blockchain projects depends, to a large extent, on the developer's involvement. Source code-commit privilege to the project is typically restricted to a few developers known as committers. Project leaders continuously search for new committers to evolve into a high-quality blockchain. However, promoting developers into committers is risky, mainly when the promoted committers exhibit low involvement behaviour. Hence, the committer assessment process is critical to the successful evolution of blockchain projects. The phenomenon of developer induction as code committers has previously been explored in the literature. However, previous studies employed objective measures such as the number of bugs report and code patches to appraise the developer's activities for promotion purposes. Although these approaches are significant, behavioural tendencies influencing developers' activities are ignored. There have, however, been comparatively few investigations on a subjective measure that evaluate developers' perception of their activities for promotion purposes. This study aims to appraise blockchain developers' perceptions for predicting future committers in blockchain projects. In this study, 173 blockchain developers' perceptions were gathered. The study employed hybrid analyses such as Structural Equation Modelling (SEM) to first analyse the survey data and Machine Learning (ML) algorithms to predict future committers. The performance of various ML algorithms was compared based on classification performance indicators such as the F1 score, accuracy, precision, and Area Under the Curve (AUC). In addition, the study also investigates the most important factors that predict future committers. The results indicate that XGBoost algorithm achieves the best performance with an accuracy of 0.94. Moreover, the most important factors in predicting future committers in a blockchain project are project desertion, developer involvement, decision-right delegation and system integration.

1. Introduction

The blockchain software development model maintains a project through collaboration among globally distributed volunteer developers. For a successful blockchain project, it is essential to attract

* *Corresponding author.*
E-mail address: stjabo@gmail.com

https://doi.org/10.37934/araset.34.2.7287

new participants and retain old developers. Blockchain, like any other large-scale OSS project, has core developers (referred to as "committers") who have role permission to commit edited source code directly to their project [1]. Blockchain project communities give such permission to a very limited number of developers who were promoted to committer roles [2]. The small number of committers ensures that blockchain projects can maintain the high quality of the project source code [3]. When researchers investigate the evolution of developer roles in OSS projects, they frequently overlook the assessment and promotion of developers to committers based on integrating of social and technical features. However, as the popularity of blockchain projects grow [3], the number of contributed patches increases at a very fast pace. Such rapid growth of contributed patches makes it difficult for relatively few committers to handle and manage in a timely fashion [4,5]. Blockchain projects need to increase the number of committers to evolve successfully [1]. However, due to the technical attraction of blockchain, the projects have hundreds of thousands of developers across the globe [3]. It is difficult to identify a promising (i.e., a developer with a good chance of staying in the project for a long time) potential committer.

Developers in software projects play a key functional role in coordinating the development tasks and lead the success of projects [6]. They interact and evolve to a new functional role to achieve their desired career and social life [7]. Project leaders must create mechanisms for role evolution best practices to ensure that only competent developers get elevated to committer responsibilities [8]. Therefore, project leaders offer the best opportunity to address committer assessment problems [8–10]. However, the project leaders do not know whether a developer promoted to committer status will continue to contribute to the project for a long time. A newly promoted committer may leave the project soon after being promoted to a committer role, resulting in knowledge and expertise loss that is difficult to recover [11,12]. A solid base study is essential to comprehend this phenomenon and consider ways to minimize problems with developer career evolution by recognizing the factors driving their contribution behaviour [13,14]. In this view, algorithms for data mining and machine learning effectively categorise datasets and extract factors to understand these behavioural patterns. Machine learning (ML) methods have been applied in a wide range of industries, including health [15], education [16] and Fintech [17].

As blockchain project is driven by an online-based developers' community that are autonomous as its structural form, without central authority [12]. The project leaders do not have formal control over the behaviour of the developers, and thus their contribution activities depend on individual-ingenuities. Therefore, in the context of blockchain project development, a developer's perception on the project was expected to be very important. In this investigation, the main objective was to assess developers' perceptions and use it to predict future committers in blockchain projects by evaluating the effectiveness of various ML classification methods. Voluntary developer perceptions of blockchain vary with various characteristic variables [3], and it is clear that this diversity of variables influences the performance, run time, and hyperparameter such as learning rate, base estimator and maximum features tuning of ML algorithms, which was used in this study [18–20]. This study examines how ML classification algorithms perform on datasets of developer perceptions in the context of committer evaluation. From an academic perspective, this research contributes both intellectual and practical value. First, the study aids in comprehending the advantages and shortcomings of ML algorithms, which could provide the groundwork for future research aimed at enhancing the ML classification algorithms for survey datasets. Secondly, this study explains how the ML classification method is employed in applied research to developer survey datasets. Ultimately, this study's objective is to answer the following questions:

i. Which ML algorithms are most accurate at predicting future committers?
ii. Which factors significantly influence the prediction of future committer?

The paper is structured as follows: The study's theoretical foundation is covered in Section 2, along with a comparison of ML classification algorithms with the literature on committer assessment practice. The methodology is covered in Section 3, along with a brief overview of the dataset's properties and the specifics of the analytic procedure. Section 4 presents the analysis and findings from the examination of the ML algorithms, and Section 5 presents the discussion and implications. Finally, the conclusion, limitations, and suggestions for future research are summarised in Section 6.

2. Literature Review

Over the past couple of decades, OSS has profoundly impacted on how software is developed, delivered, tested, shared, and maintained [21-23]. In recent times blockchain projects have been considered a new form of OSS project, that development relies fundamentally on a global community of developers for performing a variety of tasks such as reporting bugs, submitting feature requests, contributing patches and code, providing documentation, and performing beta testing [1,22]. The blockchain project is considered large-scale software development that is a very complex, effort-consuming, and expensive activity [4]. However, there exist two classes of developers in large-scale OSS, including blockchain project; an external developer who voluntarily contribute patch code only but do not have direct commit right to the main control unit of the project, while a committer not only contributes but also have commit right and can directly edit and push code to the project. As different developer plays key functional roles in OSS projects [8,10,24], they must interact and evolve over time in the project [25–27]. These blockchain projects experience role migration phenomena [1,28]. Existing literature on OSS development community turnover, to the best of our knowledge, focuses primarily on developer motivation to join, sustain and leave [3,27,29,30]. Although [8] explored the factors that influence developers’ chance to evolve into committer role, we have not yet seen extensive research on the committer assessment process that lead to the evolution of developers. As many tasks that the developer community can perform on a project, the contribution of code is important [31]. Although code contribution by a wide community is desirable, it must be controlled with a mechanism to ensure the quality of the blockchain project.

2.1 Committer Assessment Practice Overview

A committer assessment practice is a process of promoting a developer to a new committer through the recommendation of current committers or project leaders [10]. In other words, developers that contribute frequently and in a meaningful way to a project may be elevated to the position of "committer" for that project [32,33]. Prior researchers have studied developer induction to committers phenomenon [1,10,32,33]. For example, [9] developed an automated approach based on mining archive records of code repositories and bug-tracking systems in Eclipse OSS projects. Findings show that the trustworthiness of an external developer and his credibility are significant key factors that determine the developer’s promotion to commit role privileges. Moreover, the project community dynamic is another factor although less significant. Ref [34] carried out a quantitative study of the Apache, Postgres, and Python projects. They analysed three determining factors: level of skill, developer reputation on mailing lists, and commitment to continued participation. They objectively measured the time between an individual’s developer first involvement in a project’s mailing list and their first accepted code contribution. Ref [35] explore developers’ interaction in the

Python OSS project as they evolve from being developers to committers. An individual developer can become a committer by the amount of filing bug reports and offering fixes for defects, and this factor was objectively measured by mining emails from the developers' mailing list and code databases. Ref [10] develop a committer-identification model to assist Eclipse and Firefox OSS project leaders in identifying future committers. The study objectively measures the number of activities by developers. The finding shows that some developers are promoted to a committer role very rapidly because they have a large number of contributions within a few months from the start of their contributions while some of the developers take over one year to become a committer due to limited contributions. In the context of blockchain projects, [1] explored how the Ethereum project community searched for and promoted a developer to the new committer. The study uses GitHub to extract qualitative data. The study discovered that Ethereum project leaders utilised GitHub to request that individual developers showcase their skills, consider what they might offer for the project, and use it as a basis for promotion to the committer. Developers may not be considered for promotion to new committer status if they lack a particular level of ability that the community is searching for and instead are seen as a possible threat to the project rather than an asset.

In general, based on the aforementioned perspectives, extant empirical studies have been able to develop several committer promotion models by taking into consideration of objective measures of developers' activities [1,9,10,32]. There are several shortfalls in these studies. First, behavioural tendencies which influence human choice activities have been neglected. Second, most of the study focused on traditional OSS projects such as Eclipse, Apache and Mozilla projects, which are successful OSS projects. The blockchain project phenomenon has received limited attention. Although blockchain is an open-source project, the results of these studies might not apply to blockchain projects in general because blockchain implementation places a greater emphasis on security than most traditional OSS projects do, has higher defect costs. For instance, a blockchain project uses a Decentralised Application (DApp) to manage sensitive data or financial transactions. However, malicious actors may use security flaws in the coding or the blockchain network's design to compromise and manipulate transactions. Furthermore, is a complex and decentralized, lacks specialized tools for development tasks, and is challenging to upgrade after it has been released [3]. Third, from a methodological perspective, most of the studies used objective measures to appraise developer activities using project archive records to determine the frequency of individual developer's number of contributions to the project. Such data tend to be quite narrow, and project leaders may be hesitant to make full archival project records available to the researchers due to privacy and confidentiality concerns. Hence, it is quite difficult to obtain the exact number of developer activities using such objective measures. Another concern with objective measures is that such measures can easily be tampered due to human nature. The transparent nature and social media attached to the project development platforms such as GitHub and SourceForge.NET also led developers to become extremely aware that their contribution actions had an audience and therefore focused on the number of contributions rather than the quality of the contribution. Although these models can help in understanding how a developer is promoted to a committer role, there have been few studies that use survey methods to subjectively measure developer perceptions that influence their contribution behaviour [10,36–39]. This suggests the potential use of surveys to understand individual developer involvement with the blockchain project.

2.2 Factors of Committer Assessment Practice

Based on the prior literature presented in this study, we have considered ten factors that lead to project desertion [2]. As shown in Table 1 we have included intention to learn, financial gain intention, technical contribution norm, system integration, code testing task, contributed code decoupling, developer involvement, decision right delegation, and developer experience as the factors that lead to project desertion behaviour. Generally, these factors could influence project desertion behaviour [40,41].

2.3 Comparative Analysis

Information systems (IS) and related software engineering (SE) developer turnover literature mainly concentrated on creating behavioural theories and backing their evidence with empirical investigation [11,24,26–28]. Moreover, an overwhelming majority of studies have concentrated on enhancing the efficiency of ML algorithms via archive record datasets and a particular ML algorithm [18,42,43]. This has stopped the findings from being generalised. Only a few studies have examined different machine learning algorithms, and even those that have succeeded in their endeavours have used several survey datasets [44]. Recent research has shown that using multiple datasets to evaluate the performance of algorithms for a similar problem does not always produce accurate and reliable results because the datasets may contain different factors [45]. A detailed analysis is required to compare the effectiveness of several ML algorithms with a single dataset and translate the dataset's characteristics to the best ML algorithm. By taking into account individual survey datasets gathered from developers' perspectives in a Blockchain project, we seek to further this research. To further understand ML algorithms and their relationship to survey data, a set of performance measures will be used with existing literature, and their efficacy will be examined.

3. Methodology

Figure 1 depicts the four phases of this study.

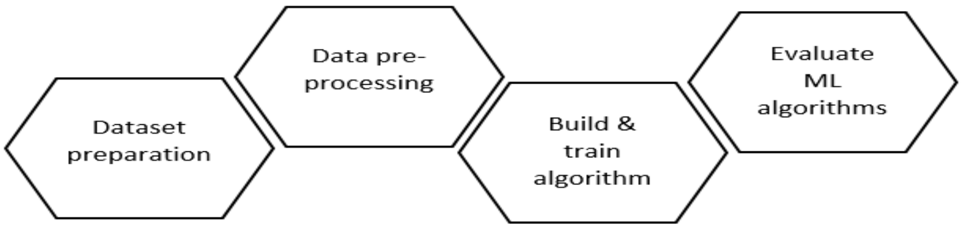

Fig. 1. ML Algorithms building, training and evaluation

3.1 Dataset Preparation

The primary objective of this study is to explore how well ML algorithms predict future committer in blockchain projects using subjective data collected through a survey. In relation to this goal, we sent an online survey to thousands of active blockchain developers on GitHub and the bitcoin project mailing list. The survey received 173 valid responses for analysis. We chose the GitHub development platform and bitcoin mailing list to distribute the questionnaires because they are popular platforms that provide socio-technical interactions among blockchain developers and capture the developers' expertise diversity [40]. To ensure valid results, we adhered to the recommendations of [3] and surveyed contributors with enough experience as developers. We discovered blockchain projects based on the following four standards:

i. Blockchain, which is made up of the cryptocurrencies Bitcoin, Ethereum, and Dash
ii. Has at least ten developers who have featured it
iii. Have at least five unique developers
iv. The repository's status as a mailing list for a blockchain project was confirmed by a manual check of the repository.

The datasets come from projects that host and manage survey data on GitHub. Because of this, the datasets can be used with different ML methods. Table 1 shows the survey factors that were considered. Factors are assessed on the same scale across all datasets. A 7-point scale is used to evaluate each factor. Even though each factor has a different number of items, the 7-point scale is frequently used. A widely used statistical measure of Cronbach's alpha was employed to assess the reliability of the measurement items. Each factor's elements are internally reliable because all factors possess Cronbach's alpha values higher than 0.6 [46]. The factors were coded, and the mean of each 173 respondents was computed against each measurement item using PLS-SEM [2,47]. PLS-SEM is a statistical modelling technique that combines elements of both structural equation modelling and partial least squares regression. An additional pre-processing is performed on the datasets by addressing and excluding the omitted values during this step.

Table 1
Factors obtained from blockchain developer survey

Factors	Code	Cronbach's Alpha	No. of Items
Intention to learn	I TL	0.780	3
Financial gain intention	FGI	0.699	4
Technical contribution norms	TCN	0.706	6
System integration	SI	0.827	4
Code testing task	CTT	0.868	5
Contributed code decoupling	CCD	0.767	5
Developer involvement	DI	0.799	5
Decision right delegation	DRD	0.778	5
Project desertion	PD	0.832	5
Developer experience	DE	0.762	3

3.2 Data Pre-Processing

Here are the steps taken to pre-process the data before training a machine learning model:

i. Load the data into a (pandas data frame) suitable data structure.
ii. Check for missing values and handle them appropriately. This could involve removing rows or columns with missing values,
iii. Checking for outliers and directing them appropriately.
iv. Standardizing and normalising the data
v. Save the pre-processed data to feed it to the machine learning algorithm.

3.3 Build and Train Algorithm

To build and train the algorithm, the following steps were followed:

i. Importing required python packages (dependencies), importing the dataset and checking if there are any NULL values,

ii. Break the dataset into training and test sets.
iii. Select an appropriate algorithm for the problem and instantiate it.
iv. Train the algorithm on the training dataset by calling the appropriate methods or functions.

We can feed a learning algorithm one row of data at a time to allow the algorithm to predict the target label giving it feedback as to whether it predicted the right answer or not (see Table 2). Over time, the algorithm will learn to approximate the exact nature of the relationship between labels. When fully trained, the supervised learning algorithm will be able to observe a new, never-before-seen and predict a good label for it. The performance indicators on the dataset are used to assess each ML algorithm. The five metrics used to evaluate performance were accuracy, precision, recall, F1 score, and area under the curve (AUC) [48]. The dataset includes several factors that forecast project abandonment.

Table 2
Reading dataset

	ITL	FGI	TCN	SI	CTT	CCD	DRD	DI	PD	DE
0	6.000	5.000	4.670	5.500	4.800	3.800	4.200	4.000	7.000	1
1	5.670	4.750	4.500	4.750	4.600	4.000	3.600	6.200	2.400	5
2	6.670	6.500	4.830	6.000	6.200	5.400	3.600	6.800	4.000	1
3	5.670	5.750	5.330	6.500	3.800	1.800	2.200	7.000	1.400	1
4	5.000	5.500	4.670	5.000	4.200	3.200	4.200	4.400	3.800	2

3.4 Evaluate ML Algorithms

It is vital to compare ML algorithms to find the most suited algorithm to solve a particular problem. In this study, Scikit-learn, a well-known Python toolkit was used for the task. Although all of the ML algorithms used in this study are useful in many situations, the best estimation is dependent on data type. That's why selecting a particular ML algorithm is essential to come up with a good estimation [44,49]. There are several parameters that need to be compared to judge the best model. After that, the best-found model needs to be tested on an independent dataset for its performance. Visualization of the performance is also a good way to compare the models quickly. Therefore, it is crucial to evaluate the differences between several ML algorithms by comparing them using the various fit statistics offered by Scikit-Learn and plotting the results. The following steps are used to compare ML algorithms [18,44].

i. Choose a group of suitable algorithms for the study's current problem.
ii. Assess how well they did on a delayed test set.
iii. Comparing the effectiveness of the algorithms requires using metrics like Area Under Curve, accuracy, precision, recall, and F1-score.
iv. Choose the algorithm that performs the best using cross-validation methods.
v. Assess the completed model for a fair assessment of its performance.
vi. Creating Receiver Operating Characteristic (ROC) curve for all the algorithms applied
vii. Repeat the procedure using different algorithms to verify the findings.

To teach a machine to learn and make predictions, detect patterns, or classify data, a data must be presented to it [50]. The three categories of ML techniques include supervised, unsupervised, and reinforcement learning [51]. This study utilised supervised learning techniques for predicting future committer problems. This is because in supervised learning, a set of input data is collected and corresponding outputs are also obtained and assembled, which can be either from subjective or objective measurements [43]. In this study, the data was obtained from human experts (blockchain developers survey). Python Jupyter Notebook was used to build and train an algorithm with ten-fold cross-validation. The accuracy and the area under the curve for logistic regression, decision trees, random forest, AdaBoost, Xtreme Gradient Boosting (XGBoost) and naive bays algorithms are calculated in order to evaluate and make comparisons. Supervised learning is the most suitable ML algorithm for this study since the dataset has a label. The ML algorithms classifier is designed to predict committer promotion based on the dataset. Each row of the standardized data represents the sample values for each factor (variable).

4. Analysis and Results

4.1 Evaluation Metrics for ML Algorithms

Multiple ML algorithms were examined using single developer survey dataset and other ML performance measures to assess each algorithm. The ML algorithms are Logistic Regression, Decision Trees, Random Forest, AdaBoost, XGBoost, and Naive Bays algorithm. We looked at metrics like accuracy, precision, recall, F1 score, and area under curve (AUC) to make the analysis easier. The percentage of correctly categorised positive observations among all positive assertions is known as a recall. The harmonic average of recall and precision, or the F1 score, is the fraction of correctly recognised positive examples classified as positive. The area under the curve, or AUC, measures how effectively a classifier predicts the future. The fraction of accurate predictions is called accuracy [19,44]. In most datasets, we find that the most popular ML performance parameter is accuracy [19]. The description of the measures used to assess the effectiveness of the ML algorithm is shown in Table 3.

Table 3
ML algorithms metrics

Ml Algorithms Metrics	**Definition**	**Reference**
Area Under Curve (AUC)	Referring to the area under the curve to shows that classifier is effective in making a prediction.	
Accuracy	The ratio of the number of correct predictions.	[19,44]
Precision	The proportion of positively predicted values that were accurately made predictions to all positively predicted values.	
Recall	The proportion of positive values that were accurately predicted to all positive values.	
FI score	The harmonic average of memory and accuracy	

4.2 Descriptive Statistics

The averages and standard deviations for each factor in the dataset are shown in Table 4. All the factor means are higher than the scale's middle value of 4. The findings demonstrate that every t-value at the 1% level, it is statistically significant. Accordingly, we draw the conclusion that the sample data is valued and believed to be reliable by most of the study population.

Table 4
Descriptive statistics

Factors	N	Minimum	Maximum	Mean	Std. Deviation
ITL	173	1.00	7.00	5.5607	1.21296
FGI	173	1.00	7.00	5.4263	1.29483
TCN	173	1.00	5.33	4.6339	0.77419
SI	173	1.00	7.00	5.6272	1.20790
CTT	173	1.00	6.20	4.7064	1.09535
CCD	173	1.00	6.00	4.7066	0.99292
DI	173	1.00	5.40	4.8276	0.91989
DRD	173	1.00	7.00	5.1399	1.17477
PD	173	1.00	7.00	4.1595	1.49743
DE	173	1.00	7.00	4.8276	1.39075

4.3 Classification Performance of ML Algorithms

We considered specific elements from the datasets to analyse and assess how well the various ML algorithms performed using the developer survey datasets. Although researchers utilize a variety of factors to determine the effectiveness of ML algorithms, it is equally beneficial to look at the significance and selection of the elements. We began our analysis by comparing four sets from six of algorithms such as Logistic regression versus Decision trees, Random forest versus Decision trees, AdaBoost vs Random forest and XGBoost vs AdaBoost ML algorithms using the Python Jupyter Notebook data science tool to generate a true positive rate to understand whether an outcome from the algorithm correctly predicts the positive class. Similarly, a true negative is whether an outcome correctly predicts the negative class. The results for the comparison based on the Area Under the Curve (AUC) is the measure of the ability of a given algorithm to distinguish between classes [43]. The higher the AUC, the better the performance of the algorithm in distinguishing between the positive and negative classes [44]. The results are shown in Figures 2(a) and 2(b).

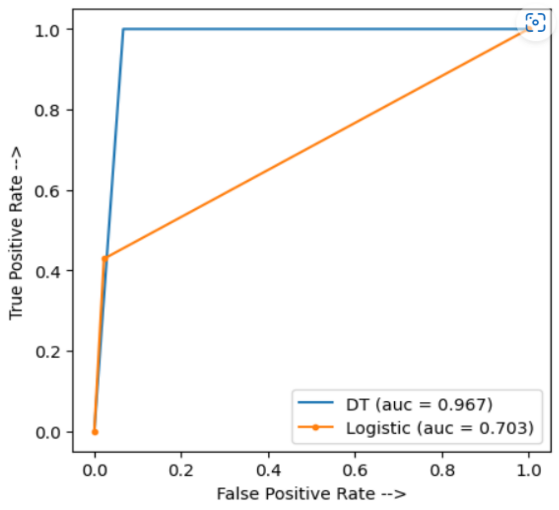

Fig. 2(a). Decision trees vs Logistic regression

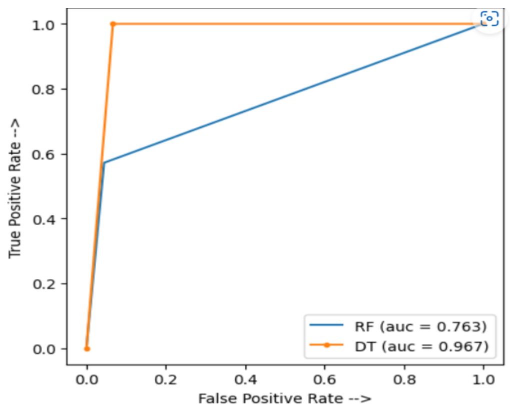

Fig. 2(b). Random forest vs Decision trees

Similarly, the true positive rate and false positive rate to compare AdaBoost (AB) and random forest (RF) and between XGBoost and AdaBoost (AB) are compared and shown in Figure 2(c) and Figure 2(d) respectively. In sum, decision tree (DT) and AdaBoost (AB) has AUC value between 0.9-1 which were considered excellent results, while random forest and XGBoost has between 0.7-0.8, which is considered fairly good AUC values.

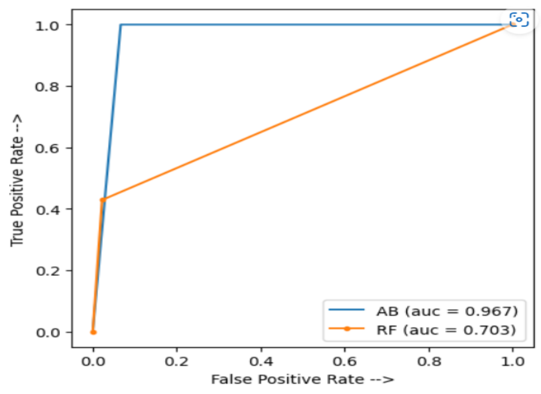

Fig. 2(c). AdaBoost vs Random forest

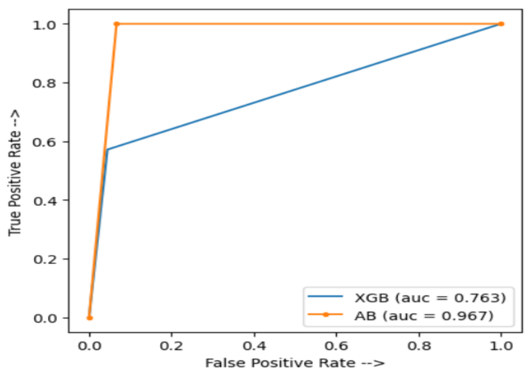

Fig. 2(d). XGBoost vs AdaBoost

However, the result indicates that the algorithm classifier is effective in making predictions. Accuracy, precision, recall, and F1 score were the other four ML metrics examined to assess the classification performance of each algorithm. Based on the accuracy metric, we conclude that XGBoost is the best performing method for the dataset, with a value of 0.94. Random Forest, Decision tree, and Naïve Bays all received 0.92, while Logistic regression received the lowest score of 0.85. Table 5 shows that XGBoost outperform the other algorithms in terms of precision, recall, and F1 score. Therefore, XGBoost is the best performing algorithm.

Table 5
ML algorithm performance matrix

	ML Metrics				
ML Algorithms	**AUC**	**Accuracy**	**Precision**	**Recall**	**F1 score**
Logistic regression	0.703	0.85	0.73	0.70	0.75
Decision trees	**0.967**	0.92	0.75	0.77	0.79
Random forest	0.703	0.92	0.80	0.76	0.78
AdaBoost	**0.967**	0.90	0.80	0.76	0.78
XGBoost	0.763	**0.94**	**0.83**	**0.83**	**0.83**
Naive Bays	0.702	0.92	0.72	0.74	0.73

We considered accuracy in this study because it is a very commonly used metric, easily understandable and in-built even for a non-technical person compared to other metrics. Furthermore, accuracy is another factor that is considered an accurate indicator of classification success. The study's findings show that XGBoost consistently outperforms other algorithms in terms of accuracy, precision, recall, and F1 score. Because accuracy is a better indicator of classification performance than other metrics, this study employed XGBoost to analyse the importance of the factors in the dataset while taking the Pareto efficiency theory into account [52].

4.4 Most Importance Factors of the Dataset

We look at the key factors that significantly impact the outcome variable to further this study. Importance factors are relevant for accurately forecasting the result variable to improve algorithm performance [53]. The relevance of elements for improved algorithm performance is sometimes referred to as the critical component of interpretable machine learning (IML), where crucial factors, come from the best algorithm fit Permutation factor importance (PFI) and conditional factor importance are the two forms of importance factor approaches (CFI) [19]. PFI was exclusively used in this study to examine factor importance because permutation importance can be computed either on the training or on the validation set. Again, in permutation, it is possible to highlight which factor contributes the most to the generalization power of the inspected algorithm. Using Python Jupyter Notebook, the data were analysed for factor importance.

In this study, we integrated and analysed the datasets using XGBoost and the scikit-learn package to determine the factor importance. Two techniques were used to determine the factor importance: the permutation method in the XGBoost regressor and the built-in factor importance in the XGBoost algorithm. Since XGBoost uses the scikit-learn interface API, we utilised scikit-learn to execute the regression. The technique computes the XGBoost algorithm's performance change by randomly reorganising each factor. The factors influencing the algorithm are the most critical factors that impact the performance. Figure 3 displays factor importance through the method applied to the XGBoost regressor. The topmost four factors include project desertion (Proj_Desertion), developer involvement (Dev_Inv), decision right delegation (Dec_Rigth_Del) and system integration (Sys_Int).

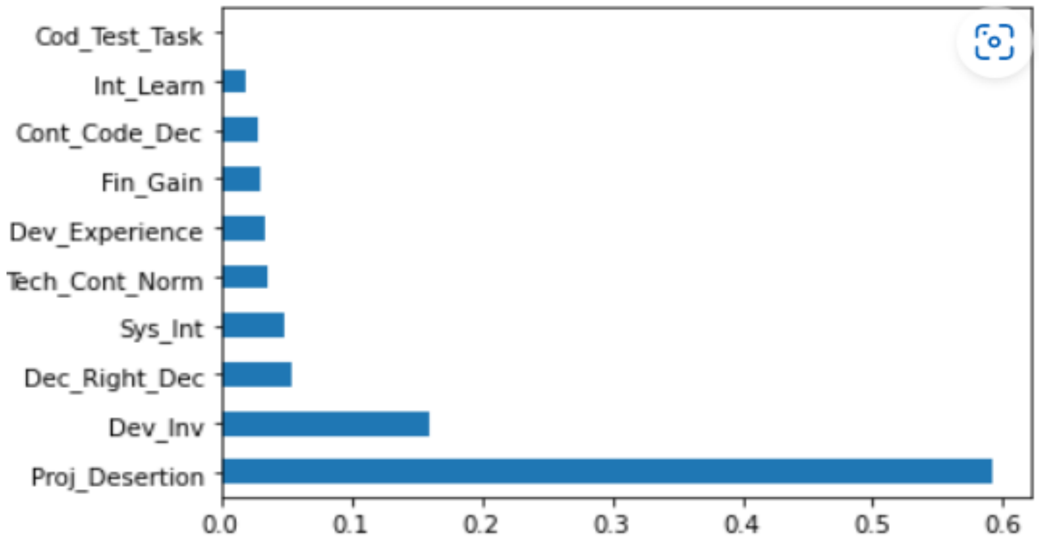

Fig. 3. Factor Importance in-built in XGBoost Regressor

This investigation demonstrates that when different ML classification methods are used together with the built-in factor significance technique, four factors continuously rank at the top.

We conclude that these four factors; project desertion (Proj_Desertion), developer involvement (Dev_Inv), decision right delegation (Dec_Rigth_Del) and system integration (Sys_Int) are significantly responsible for promoting a developer to a committer role among Blockchain developers taking into account the method of factor importance in XGBoost algorithm.

5. Discussion and Implication

This study's initial objective was to determine the optimal machine learning algorithm for classifying blockchain developer data. The second goal was to assess the relative strength of each factor in the dataset. Results indicate that the best classification algorithm for a developers' dataset is XGBoost. The XGBoost should be used while analysing the result variable for predicting future committers. In earlier research, classifying data in education using XGBoost was quite successful [44]. The investigation of the XGBoost algorithm's use in this study provides evidence for the assertions made in earlier research in other knowledge-sharing and collaborative datasets. It also considers how various factors' relative importance affects how well ML algorithms respond. After studying the dataset with the integrated method and the XGBoost regressor, we conclude that four factors;

project desertion (Proj_Desertion), developer involvement (Dev_Inv), decision right delegation (Dec_Rigth_Del) and system integration (Sys_Int) are significantly responsible for influencing software developer's promotion to a committer role, further resulting in helping blockchain project leaders in the decision-making process related to the evolution of developer community.

5.1 Practical Implications

The findings of this study are valuable for machine learning (ML) and artificial intelligence (AI) application developers from a practical standpoint. We used accuracy as a machine learning parameter to assess and contrast the classification algorithm performance. The most crucial metric for measuring algorithm performance is accuracy [54], particularly when the algorithm is meant to examine a balanced dataset. When predicting the target class (target variable) as represented by the same unlimber of input samples, we discovered that XGBoost performed better than other algorithms. For a software application that requires analysing employee or volunteer turnover data, the accuracy of the prediction while keeping a respectable classification performance would be the major consideration when creating the application. XGBoost would be a superior algorithm in this scenario to other algorithms.

5.2 Theoretical Consequences

From a theoretical standpoint, this paper adds two concepts that support prior ML algorithm literature. The application of the XGBoost algorithm to a developer turnover dataset, primarily composed of survey-based data, is presented first. Second, it explores the relevant factor using the built-in techniques or functions of the XGBoost regressor. According to this study's analysis of additional ML measures beyond accuracy, the best classification-performing method is still XGBoost. Accuracy, Precision, Recall, and F1 score are all metrics on which all ML systems did well. AUC was used as an additional metric to assess an algorithm's classifier's capacity to distinguish between classes. The higher the AUC, the better the model's performance distinguishing between the positive and negative classes [55]. The most effective algorithm was determined to be XGBoost. Even though these findings were predicted based on how ML algorithms performed on available metrics in prior literature, this work theoretically added by using five measures to the comparative analysis study on OSS developer turnover. Previous studies largely focus on archive record datasets derived from open data e.g. GitHub and GHTorren among others to objectively measure various developer participation [1,10]. To the best of our knowledge, no study has been found to have used a subjective survey dataset in studying OSS project turnover and has studied the comparison of ML algorithm using the survey dataset derived from blockchain software developer communities. We have found the ML algorithm that achieves an accuracy of 0.94 and the precision, recall and F1 score of 0.83 [44]. This study's investigation of a factor essential for the survey dataset of software developers was novel. It was done to comprehend which factors significantly affect the outcome variable predicting future committer. The build-in method in XGBoost was used because it is the best-performing algorithm with regard to the dataset to investigate factor importance while taking validity into account. By examining four key elements that affect committer promotion decision-making, we theoretically contribute to the science of software developer role evolution in online community environment issues. This is considered unique if compared to the traditional way of committer assessment using archive records to promote a developer to a committer.

6. Conclusion, Limitations and Future Work

In this study, we analysed developer perceptions of their activities to predict future committers among the hundreds of developers in blockchain projects. Furthermore, we used hybrid analyses such as Structural Equation Modelling (SEM) to analyse the survey data and finally used Machine Learning (ML) algorithms to predict future committers. The study found that project desertion, developer involvement, decision right delegation and system integration factors are significantly responsible for predicting future committers**.** The study makes a methodological contribution to the field of Information Systems (IS) as one of the few to apply a complementary multi-analytical technique to predict future committers in OSS projects such as blockchain. This study has flaws that other studies might overcome. So, first of all, our sample size is small—it includes Bitcoin developers—and might not be enough to draw a broad conclusion. However, given the widely-held belief that it is challenging to get developers to reply to an online community survey and the fact that blockchain developers are volunteers. The most challenging aspect of survey-based data collection is efficient and thorough sampling. Nevertheless, we can make a strong case on this dataset that XGBoost outperforms other classification techniques. However, to generalize their findings, researchers need to examine a broader dataset with various characteristics. Second, we have chosen the top-performing algorithm based on accuracy. Researchers should examine diverse datasets of other blockchains and related OSS projects to determine whether hybrid metrics can affect any outcome. They must provide theoretical and empirical backing for their conclusions.

Acknowledgement
This research was supported by the Ministry of Higher Education (MoHE) of Malaysia through Fundamental Research Grant Scheme (FRGS/1/2020/ICT03/UUM/02/4).

References

[1] Nilsson, Oscar, and Dorna Garagol. "Public blockchain communities A study on how governance mechanisms are expressed within blockchain communities." Master's thesis, 2018.

[2] Sarkintudu, Shehu M., Alawiyah Abd Wahab, and Huda H. Ibrahim. "PREDICTING KEY PREDICTORS OF PROJECT DESERTION IN BLOCKCHAIN: EXPERTS'VERIFICATION USING ONE-SAMPLE T-TEST." *Interdisciplinary Journal of Information, Knowledge & Management* 17 (2022). https://doi.org/10.28945/5022

[3] Bosu, Amiangshu, Anindya Iqbal, Rifat Shahriyar, and Partha Chakraborty. "Understanding the motivations, challenges and needs of blockchain software developers: A survey." *Empirical Software Engineering* 24, no. 4 (2019): 2636-2673. https://doi.org/10.1007/s10664-019-09708-7

[4] Chakraborty, Partha, Rifat Shahriyar, Anindya Iqbal, and Amiangshu Bosu. "Understanding the software development practices of blockchain projects: a survey." In *Proceedings of the 12th ACM/IEEE international symposium on empirical software engineering and measurement*, pp. 1-10. 2018. https://doi.org/10.1145/3239235.3240298

[5] De Filippi, Primavera, and Benjamin Loveluck. "The invisible politics of bitcoin: governance crisis of a decentralized infrastructure." *Internet policy review* 5, no. 4 (2016). https://doi.org/10.14763/2016.3.427

[6] Bühlmann, Noah, and Mohammad Ghafari. "How do developers deal with security issue reports on github?." In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, pp. 1580-1589. 2022. https://doi.org/10.1145/3477314.3507123

[7] Joblin, Mitchell, Sven Apel, Claus Hunsen, and Wolfgang Mauerer. "Classifying developers into core and peripheral: An empirical study on count and network metrics." In *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, pp. 164-174. IEEE, 2017. https://doi.org/10.1109/ICSE.2017.23

[8] Cheng, Can, Bing Li, Zeng-Yang Li, Yu-Qi Zhao, and Feng-Ling Liao. "Developer role evolution in open source software ecosystem: An explanatory study on GNOME." *Journal of computer science and technology* 32 (2017): 396-414. https://doi.org/10.1007/s11390-017-1728-9

[9] Sinha, Vibha Singhal, Senthil Mani, and Saurabh Sinha. "Entering the circle of trust: developer initiation as committers in open-source projects." In *Proceedings of the 8th Working Conference on Mining Software Repositories*, pp. 133-142. 2011. https://doi.org/10.1145/1985441.1985462

[10] Ihara, Akinori, Yasutaka Kamei, Masao Ohira, Ahmed E. Hassan, Naoyasu Ubayashi, and Ken-ichi Matsumoto. "Early identification of future committers in open source software projects." In *2014 14th International Conference on Quality Software*, pp. 47-56. IEEE, 2014. https://doi.org/10.1109/QSIC.2014.30

[11] Rashid, Mehvish, Paul M. Clarke, and Rory V. O'Connor. "Exploring knowledge loss in open source software (OSS) projects." In *Software Process Improvement and Capability Determination: 17th International Conference, SPICE 2017, Palma de Mallorca, Spain, October 4–5, 2017, Proceedings*, pp. 481-495. Springer International Publishing, 2017. https://doi.org/10.1007/978-3-319-67383-7_35

[12] DiRose, Stephen, and Mo Mansouri. "Comparison and analysis of governance mechanisms employed by blockchain-based distributed autonomous organizations." In *2018 13th annual conference on system of systems engineering (SoSE)*, pp. 195-202. IEEE, 2018. https://doi.org/10.1109/SYSOSE.2018.8428782

[13] Raza, Mishal, and Sadia Nadeem. "Drivers of Employee Engagement and their Impact on Job Satisfaction and Turnover Intentions." *Journal of Managerial Sciences* 12, no. 2 (2018).

[14] Arnold, Maxwell. "Google's involvement is a step forward for blockchain." *LSE Business Review* (2018).

[15] Alquran, Hiam, Mohammad Alsleti, Roaa Alsharif, Isam Abu Qasmieh, Ali Mohammad Alqudah, and Nor Hazlyna Binti Harun. "Employing texture features of chest x-ray images and machine learning in covid-19 detection and classification." In *Mendel*, vol. 27, no. 1, pp. 9-17. 2021. https://doi.org/10.13164/mendel.2021.1.009

[16] Yusof, Yuhanis, and Fathima Fajila. *Artificial Intelligence and Machine Learning in Education*. World Scientific Publishing Co. Pte. Ltd., 2022. https://doi.org/10.1142/9789811254154_0007

[17] Stojanović, Branka, Josip Božić, Katharina Hofer-Schmitz, Kai Nahrgang, Andreas Weber, Atta Badii, Maheshkumar Sundaram, Elliot Jordan, and Joel Runevic. "Follow the trail: Machine learning for fraud detection in Fintech applications." *Sensors* 21, no. 5 (2021): 1594. https://doi.org/10.3390/s21051594

[18] Lee, Hee-Sun, Gey-Hong Gweon, Trudi Lord, Noah Paessel, Amy Pallant, and Sarah Pryputniewicz. "Machine learning-enabled automated feedback: supporting students' revision of scientific arguments based on data drawn from simulation." *Journal of Science Education and Technology* 30 (2021): 168-192. https://doi.org/10.1007/s10956-020-09889-7

[19] Cheema, Muhammad Asaad, Nouman Ashraf, Asad Aftab, Hassaan Khaliq Qureshi, Muhammad Kazim, and Ahmad Taher Azar. "Machine Learning with Blockchain for Secure E-voting System." In *2020 First International Conference of Smart Systems and Emerging Technologies (SMARTTECH)*, pp. 177-182. IEEE, 2020. https://doi.org/10.1109/SMART-TECH49988.2020.00050

[20] Jiang, Jing, Fuli Feng, Xiaoli Lian, and Li Zhang. "Long-Term Active Integrator Prediction in the Evaluation of Code Contributions." In *SEKE*, pp. 177-182. 2016. https://doi.org/10.18293/SEKE2016-030

[21] Huang, Yuekai, Ye Yang, Junjie Wang, Wei Zheng, and Qing Wang. "Identifying Emergent Leadership in OSS Projects Based on Communication Styles." *arXiv preprint arXiv:2201.11897* (2022). https://doi.org/10.1109/SANER56733.2023.00017

[22] Lindman, Juho. "What Open Source Software Research Can Teach Us About Public Blockchain (s)?—Lessons for Practitioners and Future Research." *Frontiers in Human Dynamics* (2021): 13. https://doi.org/10.3389/fhumd.2021.642556

[23] Wang, Dan, Terh Jing Khoo, and Zhangfei Kan. "Exploring the Application of Digital Data Management Approach for Facility Management in Shanghai's High-rise Buildings." *Progress in Energy and Environment* 13 (2020): 1-15.

[24] Khan, Beenish, Muhammad Rafiq Mufti, Asad Habib, Humaira Afzal, Mohammad Abdul Moiz Zia, Afshan Almas, Shahid Hussain, and Bashir Ahmad. "Evolution of Influential Developer's Communities in OSS and its Impact on Quality." *Intelligent Automation & Soft Computing* 28, no. 2 (2021). https://doi.org/10.32604/iasc.2021.015034

[25] Ferreira, Fabio, Luciana Lourdes Silva, and Marco Tulio Valente. "Turnover in Open-Source Projects: The Case of Core Developers." In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering*, pp. 447-456. 2020. https://doi.org/10.1145/3422392.3422433

[26] Iaffaldano, Giuseppe, Igor Steinmacher, Fabio Calefato, Marco Gerosa, and Filippo Lanubile. "Why do developers take breaks from contributing to OSS projects? A preliminary analysis." *arXiv preprint arXiv:1903.09528* (2019).

[27] Calefato, Fabio, Marco Aurelio Gerosa, Giuseppe Iaffaldano, Filippo Lanubile, and Igor Steinmacher. "Will you come back to contribute? Investigating the inactivity of OSS core developers in GitHub." *Empirical Software Engineering* 27, no. 3 (2022): 76. https://doi.org/10.1007/s10664-021-10012-6

[28] Kaur, Rajdeep, and Kuljit Kaur Chahal. "Exploring factors affecting developer abandonment of open source software projects." *Journal of Software: Evolution and Process* 34, no. 9 (2022): e2484. https://doi.org/10.1002/smr.2484

[29] Daniel, Sherae, Shadi Janansefat, E. Ilana Diamant, and Yuqing Ren. "Single-and double-loop learning: linking free/libre open source software (FLOSS) developer motivation, contribution, and turnover intentions." *ACM SIGMIS*

Database: the DATABASE for Advances in Information Systems 51, no. 4 (2020): 68-92. https://doi.org/10.1145/3433148.3433153

[30] Song, Jaeyoon, and Changhee Kim. "What Is Needed for the Sustainable Success of OSS Projects: Efficiency Analysis of Commit Production Process via Git." *Sustainability* 10, no. 9 (2018): 3001. https://doi.org/10.3390/su10093001

[31] Hausberg, J. Piet, and Sebastian Spaeth. "Why makers make what they make: motivations to contribute to open source hardware development." *R&D Management* 50, no. 1 (2020): 75-95. https://doi.org/10.1111/radm.12348

[32] Canfora, Gerardo, Massimiliano Di Penta, Rocco Oliveto, and Sebastiano Panichella. "Who is going to mentor newcomers in open source projects?." In *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering*, pp. 1-11. 2012. https://doi.org/10.1145/2393596.2393647

[33] Long, Ju. "Understanding the Role of Core Developers in Open Source Software Development." *Journal of Information, Information Technology & Organizations* 1 (2006). https://doi.org/10.28945/148

[34] Bird, Christian, Alex Gourley, Prem Devanbu, Anand Swaminathan, and Greta Hsu. "Open borders? immigration in open source projects." In *Fourth International Workshop on Mining Software Repositories (MSR'07: ICSE Workshops 2007)*, pp. 6-6. IEEE, 2007. https://doi.org/10.1109/MSR.2007.23

[35] Ducheneaut, Nicolas. "Socialization in an open source software community: A socio-technical analysis." *Computer Supported Cooperative Work (CSCW)* 14 (2005): 323-368. https://doi.org/10.1007/s10606-005-9000-1

[36] Jongyindee, Anakorn, Masao Ohira, Akinori Ihara, and Ken-ichi Matsumoto. "A Case Study of Committers' Activities on the Bug Fixing Process in the Eclipse Project." In *International Workshop on Empirical Software Engineering in Practice 2011 (IWESEP 2011)*, p. 9. 2011. https://doi.org/10.1109/IWSM-MENSURA.2011.24

[37] Gamalielsson, Jonas, and Björn Lundell. "Long-term sustainability of open source software communities beyond a fork: A case study of libreoffice." In *Open Source Systems: Long-Term Sustainability: 8th IFIP WG 2.13 International Conference, OSS 2012, Hammamet, Tunisia, September 10-13, 2012. Proceedings 8*, pp. 29-47. Springer Berlin Heidelberg, 2012. https://doi.org/10.1007/978-3-642-33442-9_3

[38] Andrychowicz, Marcin, Stefan Dziembowski, Daniel Malinowski, and Łukasz Mazurek. "Secure multiparty computations on bitcoin." *Communications of the ACM* 59, no. 4 (2016): 76-84. https://doi.org/10.1145/2896386

[39] Jongyindee, Anakorn, Masao Ohira, Akinori Ihara, and Ken-ichi Matsumoto. "Good or Bad Committers?—A Case Study of Committer's Activities on the Eclipse's Bug Fixing Process." *IEICE TRANSACTIONS on Information and Systems* 95, no. 9 (2012): 2202-2210. https://doi.org/10.1587/transinf.E95.D.2202

[40] Tsay, Jason, Laura Dabbish, and James Herbsleb. "Influence of social and technical factors for evaluating contribution in GitHub." In *Proceedings of the 36th international conference on Software engineering*, pp. 356-366. 2014. https://doi.org/10.1145/2568225.2568315

[41] Joblin, Mitchell, and Sven Apel. "How do successful and failed projects differ? a socio-technical analysis." *ACM Transactions on Software Engineering and Methodology (TOSEM)* 31, no. 4 (2022): 1-24. https://doi.org/10.1145/3504003

[42] Almarzouqi, Amina, Ahmad Aburayya, and Said A. Salloum. "Prediction of user's intention to use metaverse system in medical education: A hybrid SEM-ML learning approach." *IEEE access* 10 (2022): 43421-43434. https://doi.org/10.1109/ACCESS.2022.3169285

[43] Shah, Dhruvil, Devarsh Patel, Jainish Adesara, Pruthvi Hingu, and Manan Shah. "Integrating machine learning and blockchain to develop a system to veto the forgeries and provide efficient results in education sector." *Visual Computing for Industry, Biomedicine, and Art* 4, no. 1 (2021): 1-13. https://doi.org/10.1186/s42492-021-00084-y

[44] Muzumdar, Prathamesh, Ganga Prasad Basyal, and Piyush Vyas. "An empirical comparison of machine learning models for student's mental health illness assessment." *arXiv preprint arXiv:2202.13495* (2022). https://doi.org/10.24203/ajcis.v10i1.6882

[45] Georgoula, Ifigeneia, Demitrios Pournarakis, Christos Bilanakos, Dionisios Sotiropoulos, and George M. Giaglis. "Using time-series and sentiment analysis to detect the determinants of bitcoin prices." *Available at SSRN 2607167* (2015). https://doi.org/10.2139/ssrn.2607167

[46] Peterson, Robert A., and Yeolib Kim. "On the relationship between coefficient alpha and composite reliability." *Journal of applied psychology* 98, no. 1 (2013): 194. https://doi.org/10.1037/a0030767

[47] Götz, Oliver, Kerstin Liehr-Gobbers, and Manfred Krafft. "Evaluation of structural equation models using the partial least squares (PLS) approach." In *Handbook of partial least squares: Concepts, methods and applications*, pp. 691-711. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. https://doi.org/10.1007/978-3-540-32827-8_30

[48] Hansun, Seng, Arya Wicaksana, and Abdul QM Khaliq. "Multivariate cryptocurrency prediction: comparative analysis of three recurrent neural networks approaches." *Journal of Big Data* 9, no. 1 (2022): 1-15. https://doi.org/10.1186/s40537-022-00601-7

[49] Roslan, Nur Widad, Normaliza Abd Rahim, Nur Maisarah Roslan, and Siti Nur Aliaa Roslan. "Students' presupposition towards incooperating AI (Artifical Intelligence) technology in virtual and face-to-face

classes." *International Journal of Advanced Research in Future Ready Learning and Education* 27, no. 1 (2022): 16-19.

[50] Albreem, Mahmoud A., Abdul Manan Sheikh, Mohammed H. Alsharif, Muzammil Jusoh, and Mohd Najib Mohd Yasin. "Green Internet of Things (GIoT): applications, practices, awareness, and challenges." *IEEE Access* 9 (2021): 38833-38858. https://doi.org/10.1109/ACCESS.2021.3061697

[51] Su, Leona Yi-Fan, Michael A. Cacciatore, Xuan Liang, Dominique Brossard, Dietram A. Scheufele, and Michael A. Xenos. "Analyzing public sentiments online: Combining human-and computer-based content analysis." *Information, Communication & Society* 20, no. 3 (2017): 406-427. https://doi.org/10.1080/1369118X.2016.1182197

[52] Chen, Jason Chou-Hong, P. Pete Chong, and Ye-Sho Chen. "Decision criteria consolidation: A theoretical foundation of Pareto Principle to Porter's Competitive Forces." *Journal of Organizational Computing and Electronic Commerce* 11, no. 1 (2001): 1-14. https://doi.org/10.1207/S15327744JOCE1101_01

[53] Abbasi, Ghazanfar Ali, Lee Yin Tiew, Jinquan Tang, Yen-Nee Goh, and Ramayah Thurasamy. "The adoption of cryptocurrency as a disruptive force: Deep learning-based dual stage structural equation modelling and artificial neural network analysis." *Plos one* 16, no. 3 (2021): e0247582. https://doi.org/10.1371/journal.pone.0247582

[54] Entezari, Zahra, and Masoud Mahootchi. "Developing a mathematical model for staff routing and scheduling in home health care industries: Genetic algorithm-based solution scheme." *Scientia Iranica* 28, no. 6 (2021): 3692-3718.

[55] Bao, Lingfeng, Xin Xia, David Lo, and Gail C. Murphy. "A large scale study of long-time contributor prediction for github projects." *IEEE Transactions on Software Engineering* 47, no. 6 (2019): 1277-1298. https://doi.org/10.1109/TSE.2019.2918536