Task Allocation in Distributed Agile Software Development Environment Using Unsupervised Learning

DOI : 10.36909/jer.ICMET.17167

Madan Singh, Naresh Chauhan, Rashmi Popli,

Department of Computer Engineering, J.C. Bose University of Science & Technology, YMCA, Faridabad, Haryana (India)

Corresponding Author: madansingh.research@gmail.com

ABSTRACT

In the paper, a novel approach for task allocation in DASD environment has been proposed. In the approach, new tasks (in the form of user – stories), are allocated to an employee, who is found to be 'best', on the basis of classification and rank ordering. For applying classification and rank ordering on data set of employees, Meta - Classifier Based Prediction Model (MCBPM) has been used that applied unsupervised learning. Results show that MCBPM – based task allocations provides accurate suggestions for the activity.

Keywords: - Agile software development (ASD), Distributed Agile Software Development (DASD), Task Allocation (TA).

INTRODUCTION

Software development process is organized for the delivery of cost-effective software products quicker with improved quality. Agile software development (ASD) (Nundlall C., Nagowah S.D. 2021) is a development method which adapts lightweight project development strategy and promises to deliver cost- effective quality products. It is an iterative development method. Its basic concept is people-centered and it admires frequent changes in the requirements. This becomes possible by making the customer an inherent part of the team meetings and all discussion forums throughout the life cycle of the product. ASD follows "Agile Manifesto" (M. Simão Filho, P.R. Pinheiro, P. A. Barbosa 2020), which emphasizes more on following outcome based practices. ASD relies on smaller team with self-motivated people. Figure 1 gives an idea about six different phases of ASD.

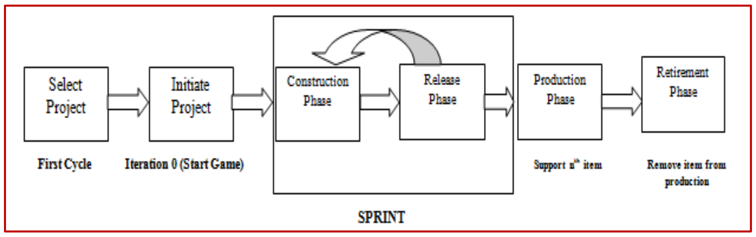


Figure 1-Agile Software Development (ASD) Life Cycle

In the early phase of ASD life cycle, the goal of the project and its market aspects are defined. It explores about the new functionality improvements and its impacts on organization's presence in the market. It identifies the potential stakeholders and their goals. ASD requires focused development team with good understanding about sprint cycles and periodic tasks, elimination of communication gaps among team members as well as with the customer. Agile teams perform well with fixed locations in order to reduce delays and communication gaps. The hunt for bigger markets and need of round the clock software development for large projects is the basis of distributed development. Distributed environment also supports availability of larger talented and motivated people with handful of technical expertise and capabilities with cost optimization. In distributed development, team members working to achieve same project objectives may belong to different locations across the country or across the globe. In another development approach, separate parts of same product may be developed at different sites globally. Exercising Agile manifesto in distributed environment, leads to Distributed Agile Software Development (DASD) (M. Perkusich, L. Chaves e Silva, A. Costa et. al. 2020). DASD approach offers excellent paybacks like-cheaper manual labour, proximity to business markets and engagement of experienced workers with proven capabilities to different projects globally. DASD contains geographically dispersed two or more teams which work on same projects. The dispersed teams use a common framework to perform different activities of work distribution, task allocation and intra-team coordination. In this way, these distributed and dispersed teams provide work scalability with lower cost products, extended talent, and faster turnaround times (Ijaz F, Aslam W., 2019). DASD also gives rise to challenges for "Agility and Distributivity" (Zhe Wang., 2019). The challenges contain those challenges which arise due to multiplicity of teams. The challenges include mistrust among team members, poor association among different development sites and poor communicational and coordination skills (Zhe Wang., 2019). In this built up, task allocation in distributed environment becomes a difficult job. It must be performed properly for the success of the project because it controls software cost as well as time concerns.

In Agile context, task set comprise of set of user stories. Task allocation is based on identification of task set or user stories for which human resources are to be allocated. Project managers allocate task to team members based upon their willingness, goodwill, and experience to handle similar projects (M. Simão Filho, P.R. Pinheiro, A.B. Albuquerque, 2018). Task allocation in DASD environment evolves non-trivial risks. These risks may further lead to improper or sub – optimal task allocations leading to project failures (M. Simão Filho, P.R. Pinheiro, A.B. Albuquerque, 2018). Task allocation plan needs to address these problems and it should consider characteristics and relationships among distributed teams (A. Aslam, N. Ahmad, T. Saba, A. S. Almazyad, A. Rehman, A. Anjum, and A. Khan, 2017). Besides this, there exist many factors that control the decision making for task allocation. These factors include factors based on people understanding and expertise and knowledge based factors. The organization of the paper is as follows: Next section presents literature survey on issues related to task allocation mechanisms followed by an algorithm for the proposed MCBPM model and result analysis.

RELATED WORK

In Agile based development, tasks are available in the form of user stories. The user stories deliver a specific functionality. Many researchers have applied different approaches for task allocation over the years. In the section, a summarized overview of related work has been presented. The literature survey has been done keeping in view two research queries:-

RQ1 - In there any prediction – based system, that is applied by researchers for task allocation?

RQ2 - What is the type of Decision support system used, (i.e. Manual, Semi – Automated, or Automated)?

(Bokhari, S. H. *et al.,* 1981, 1987) identified task assignment as a challenging task. The authors used min cut-max flow approach for task assignment. The approach considered task assignment as a network flow problem and tried to resolve the issue with its help. (Shen M. *et al.*, 2003) proposed qualitative measures based mechanism for task allocation. The evaluation procedure take three criteria based on skill set of workers and required skills for task accomplishment. Suitability scores are calculated and on the basis of suitability score of team members candidates are ranked. The rankings are applied for task allocations. (Duggan J. *et al.*, 2004) proposed an optimized method for task allocation using genetic algorithms which includes fitness assignment, crossover and mutation phases. The authors used multi objective evolutionary algorithm and genetic algorithms for task allocation. The approach was effective for scheduling and cost estimation. (Mak D. K. M. *et al.*, 2006) proposed an analytic hierarchy process methodology to do task allocations and coordination. The authors used quantitative (i.e.: time, resources) and qualitative (i. e.: team member skill set, experience) factors for rank ordering. (Lamersdorf A. *et al*., 2010) proposed a decision support system for task allocation that considers multiple criteria for the decision making in distributed development environment. Three categories of factors had been identified for calculation of transmission cost namely, "*Site Factors having process maturity, Dependency between sites Factors having language differences, cultural differences, common experiences, infrastructure link, time shift and dependency between tasks Factors having task coupling, round-the-clock possibility*". The authors used Bayesian networks in order to suggest a prioritized list of assignments. (Lin J. *et al.*, 2014) proposed a task allocation decision support system for distributed system. The authors grouped task allocation activity in to three categories - namely "*type 1 or equality based group, type 2 or mixed strategy group and type 3 or competence based group*". They formalized the problem as distributed constraint optimization problem with the help of software named HASE. (Marques A. B. *et al.*, 2013) proposed a domain ontology to represent concepts related to task allocation in distributed teams. The ontology was defined based on a literature systematic mapping and on the opinion of experts. It was used to bring awareness to managers regarding the factors related to task allocation planning. (Masood Z. *et al.*, 2017) proposed four dissimilar approaches of workflow across all teams namely – "*team independent, team dependent, skill set / module dependent, hybrid workflow*" and five dissimilar types of task allocation approaches were identified based on increasing level of team and individual autonomy namely - "*individual-driven, manager-driven, team-driven, manager-assisted, team- assisted*". (Filho M.S. *et al.*, 2018, 2020) proposed a decision support system for distributed system and identified task allocation to remote team members as a challenge. The authors worked on eleven influential factors for task allocation in Distributed Agile Software Development. These factors are -"technical *expertise, expertise in business, project manager maturity, and proximity to client, low turnover rate, availability, site maturity, personal trust, time zone, cultural similarities and* *willingness at site*". Following these factors and three criteria, they identified most influential factors and ordered them using a tool named as ZAPROS-III-i. The authors used three criteria namely-"*facilities to carry out work remotely, time for the project and cost for the project*". ARANAÚ tool was used for decision making and for alternative definition and result generation. (Aslam W. *et al.*, 2018) identified task allocation as one of the major challenges in project scheduling and management in distributed Agile software development. The authors suggested a task allocation framework to identify factors and dependencies which influence task allocation decision making. (Zhe Wang *et al.*, 2018, 2019) used Multi – Agent based mechanism for task allocation. The authors further evaluated relation between agent and task using preference-value calculation method for task allocation. The authors suggested a formula based method to calculate time required to complete a given task that can be based on many factors. (Z. Masood *et al.*, 2020) proposed grounded theory based approach for self- assignment work. The study explores how task assignment works for Agile projects. (Nundlall C. et al., 2021) performed a systematic review and categorized the work in field of task allocation into five categories, namely- "Task allocation using a quantitative method, Task allocation using a situational approach, Multi-criteria task allocation, UML-based meta-models in task-assignment and Team coordination approaches in large projects and distributed teams". The literature survey work draws certain findings. The important findings are in response to RQ1, RQ2. The conclusive results of literature survey have been shown through tabular representation. It is quite evident that none of the researcher had applied predictive means for finding best match for task allocation and decision making process used was manual in nature. It becomes more crucial in Distributed Agile based development due to language and cultural barriers among the team members (W. Aslam, F. Ijaz, M. I. Lali, and W. Mehmood,2017). Based on the literature outcome, a Meta-Classifier based prediction model has been proposed. The model has used automated prediction and decision making with the help of unsupervised learning techniques.

MCBPM – META CLASSIFIER BASED PREDICTION MODEL

Machine learning (M. Perkusich, L. Chaves e Silva, A. Costa et al, 2020) has proven to be an efficient tool for prediction and classification. It has classifications as unsupervised, supervised and reinforcement machine learning. Unsupervised learning [9], is machine learning mechanism where models are trained using un-labelled and uncategorized data. After getting input, model interprets these inputs for clustering or association for which, specific Algorithm is applied. The model provides output based on certain processing work. Figure-1 represents unsupervised learning process diagrammatically.

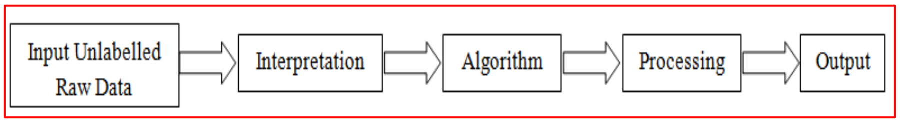


Figure 1– Unsupervised Learning Process

In the paper, task allocation with unsupervised learning has been taken as one of the subtask of Agile based project management activity Different classifiers (M. Perkusich, L. Chaves e Silva, A. Costa et al, 2020), have been trained with the help of unlabelled dataset and meta – classifier have been applied to get the best match among the employees to whom task is allocated. Figure – 2 gives a glimpse to task allocation activity as a subset of overall framework for project management in Agile environment. The framework makes use of MCBPM based task allocation module for finding best suited employee for any specified task. Backlogs produced after each sprint have been processed with the help of MCBRS module (M. Perkusich, L. Chaves e Silva, A. Costa et al, 2020), which prioritize the available set of backlogs applying MoSCoW based prioritization.

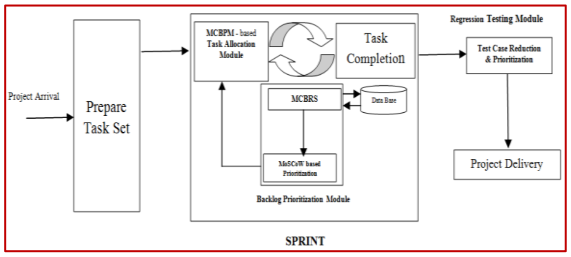

Project Arrival
Prepare Task Set
MCBPM - based Task Allocation Module
Task Completion
MCBRS
Data Base
MoSCoW based Prioritization
Backlog Prioritization Module
Regression Testing Module
Test Case Reduction & Prioritization
Project Delivery
SPRINT

Figure 2– Agile based project management framework having task allocation as a subtask

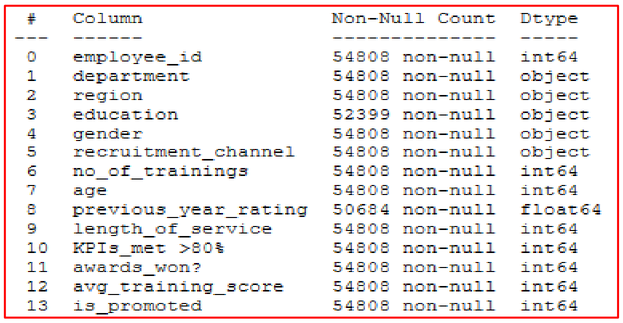
Algorithm -1 gives the bit by bit description of the steps followed by MCBPM model. It works on unlabelled data set. In Meta-Classifier based prediction model, a meta classifier has been used which is initially trained after clustering and LDA based extractions. Trained and tested model is used for the prediction of best suited employee who is best suited for task allocation with the help of the data set passed. The data set provides the required information regarding the employees working in the organization among which a task is required to be allocated in scrum based environment.

```
Inputs Training data D
Output An ensemble classifier H
Step 1:- Training of data.
Step 2:- Do K-MEANS clustering on the Input data.
Step 3:- Use LDA for Extract top topics from clusters.
Step 4:- Train first level classifiers.
        For t ←1 to T do Teach a base classifier h based on D End for
Step 5:- Construct new data set D' from D.
        For i ←1 to M do Construct a new data set D' End for
Step 6:- Learn a second level classifier.
        Learn a new meta classifier h' based on newly constructed data set D'.
Step 7:- Training & testing of Meta - Classifier to classify the data.
Step 8:- Apply Naïve-Bayes algorithm on data set D'.
Step 9:  Apply Support Vector Machine for minimization of margin of hyper – plane.
Step 10: Apply Logistic regression probabilistic part, to minimize the distance of every point from the hyper -
plane, and try for best hyper - plane which gives best results.
Return
```

Algorithm1 - Algorithm for MCBPM Model

RESULTS & DISCUSSION

Prediction based machine learning methods to make task allocation easy and effective remains to be need of the hour. In this section, result analysis for unsupervised learning based task allocation has been shown. Data set named Employee_dataset has been used for implementation. The dataset contains around 24000 employee data set collected from May 2020 - July 2021. Further, Unsupervised learning on the data set has been applied. Data set has been made ready by removing all redundancies available and removal of null values. Further, the data set containing key features related to the employee has been applied to MCBPM Model. Figure -3 gives the snapshot of the columns used from the data set. Training and testing data sets were prepared on the basis of 80 - 20 rule, where 80% of the data set is used for training and remaining 20% data set is used for testing.



```
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   employee_id            54808 non-null  int64
 1   department             54808 non-null  object
 2   region                 54808 non-null  object
 3   education              52399 non-null  object
 4   gender                 54808 non-null  object
 5   recruitment_channel    54808 non-null  object
 6   no_of_trainings        54808 non-null  int64
 7   age                    54808 non-null  int64
 8   previous_year_rating   50684 non-null  float64
 9   length_of_service      54808 non-null  int64
 10  KPIs_met >80%          54808 non-null  int64
 11  awards_won?            54808 non-null  int64
 12  avg_training_score     54808 non-null  int64
 13  is_promoted            54808 non-null  int64
```

Figure 3 – Snapshot of columns used from the data set

Figure -4 shows skill – based classification of the data set. From Highly-skilled employees, employees with higher training score have been extracted. An employee having high training score is considered to be best for selection and any new task, if it is there, it should be allocated to employees having high ratings for their training. Figure-5 represents distribution of training score among the employees. Task In figure-6, pie chart represents gap in employees in terms of required skills needed for project for which we need to assign a team member to whom tasks need to be allocated. This is one of the important factors. By using this we separate skilled employee and under skilled employees.

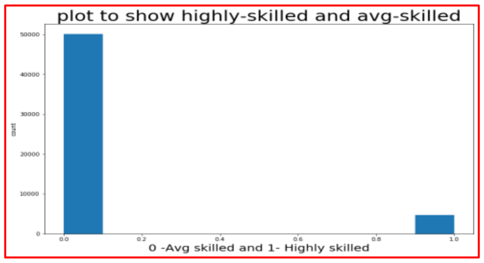


Figure 4– Graph representing highly - skilled and average – skilled Work Force

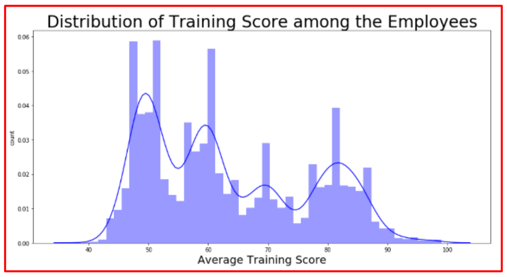
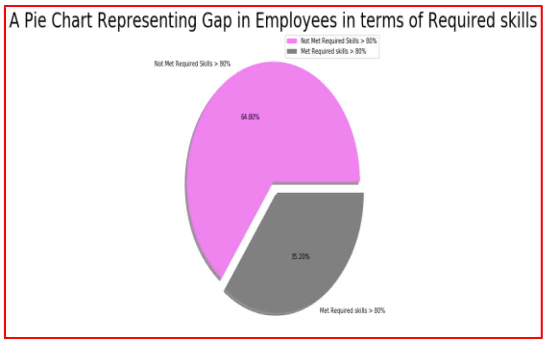


Figure 5– Graph representing Distribution of training score among the employees

Distribution of length of service among the employees is another factor that is important from the project success point of view. An employee with longer time duration in the same firm is supposed to be reliable and a reliable team member is also found to be responsible. We need reliable and responsible members for task allocation. Figure -7, gives distribution of length of service among the employees.



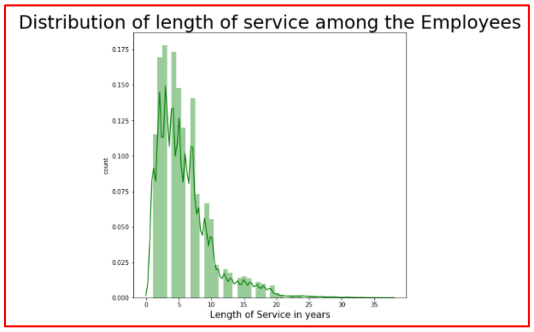
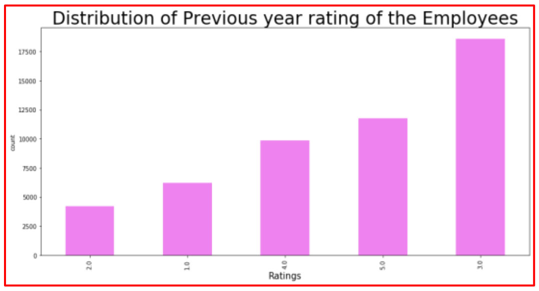


Figure 6 – Pie chart representing gap in employees in terms of required skill set.

Figure 7– Distribution of length of service among the employees

The graph in figure-8, represents distribution of previous year ratings of employees in our data set. This distribution is important from prediction point of view.



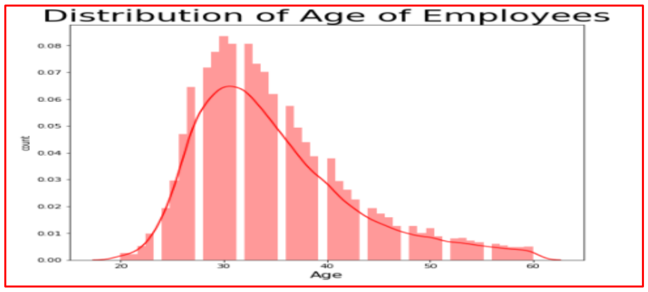


Figure 8– Representation of distribution of previous year rating of the employees

Figure 9– Representation of distribution of age of employees

Age of an employee is another important aspect. A person with age group below 40 is found to be more energetic and their more eager to handle challenges. As acceptance is related to willingness, employees work more happily in distributed Agile environment. Figure 9, represents distribution of age of employees. A pre- trained employee will take lesser time to adapt to the platform and due to this development work is finished as per the target decided. Figure-10, shows the frequency of trainings done by the employees.

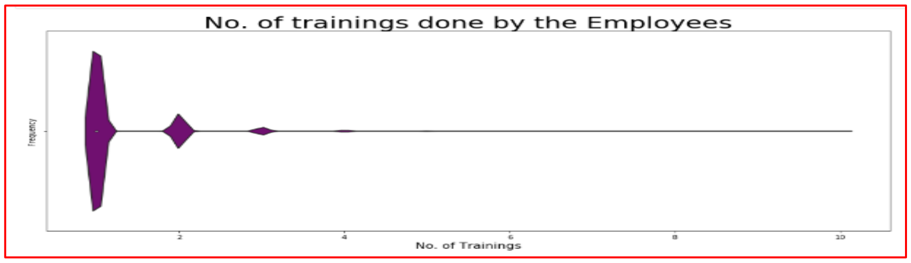


Figure 10– Frequency representation of number of trainings done by the employees

As per the proposal, the model is required to work for distributed Agile software development, distribution of different regions in which the organization exist is also a parameter to be taken care of properly. Prediction model should take into account these regions also before task is allocated.

If a suited match is found from the same region where the vacancy exists, then they are given priority over others.

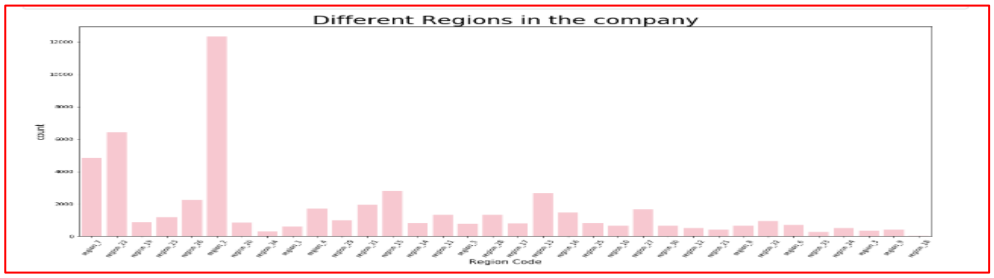


Figure 11 – Different regions in the company

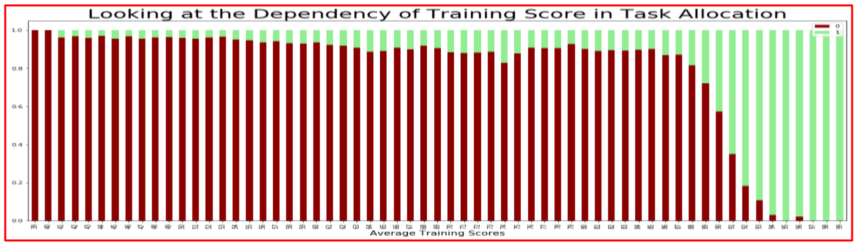


Figure 12– Representation of dependency of training score in Task Allocation

In figure-12, graph represents dependency of training score in task allocation. Based on the training scores best match is selected and allocated with the task. The hypothesis follows the confusion matrix. Accuracy and precision are important parameters to check prediction. F1 is a function of precision and recall. F1 score is needed to seek a balance between precision and recall. The formula to calculate F1 can be given by:

$$F1 = 2* \{(Precision – Recall) / (Precision + Recall)\}$$

Recall and precision are another important aspects, which is given by the following formulas:-

Recall = {True Positives / (True Positive + False Negative)}

Whereas; **Precision = {True Positive / (True Positive + False Positive)}**

The results represent that if we predict the task allocation based on the available data set, and then chances of better task allocation and project completion are increased. Table -1 below represents the accuracy of different models and comparison with MCBPM model.

Table 1 - Comparison of MCBPM model with different models based on certain parameters

Model_Name	**Accuracy**	**Precision**	**Recall**	**Roc_Score**	**F1_Score**
Logistic Regression	0.920993	0.631579	0.073270	0.554961	0.534739
Random Forest	0.935866	0.742285	0.32632	0.719540	0.658135
Decision Tree Classifier	0.903854	0.420801	0.477612	0.697378	0.709642
Naive Bayes	0.928899	0.783133	0.176391	0.625255	0.586028
SVM	0.936363	0.928382	0.23744	0.672320	0.617912
MCBPM Model	**0.987545**	0.766675	0.124323	**0.813412**	**0.712943**

CONCLUSION

For large software firms, working with distributed and dispersed teams has been a substitute increasingly available in modern era of project development. However, task allocation among remote teams is very crucial since there are many factors that project managers should take into consideration before assigning a task to any specific team member. MCBPM model supports decision-making process through machine learning based multi-criteria qualitative analysis. The model has been compared with the support vector machine, random forest, decision tree classifier, naïve bayes and logistic regression models. The results show that MCBPM model shows promising result with parameters like accuracy, Roc_Score, F1_Score etc. Comparative results represent that the usage of supervised learning-based prediction provides timely allotment of best match of human resources to user stories which results in timely project completion, higher customer satisfaction and higher product acceptance by customers in distributed Agile environment.

REFERENCES

[1] Nundlall C., Nagowah S.D. (2021). Task allocation and coordination in distributed agile software development: a systematic review. International Journal of Information Technology, Vol.-13, pp. 321-330.

[2] M. Simão Filho, P.R. Pinheiro, P. A. Barbosa (2020). Dealing Hybrid Model Appling in

the Selection and Prioritization of Software Requirements Using Verbal Decision Analysis, Journal of Applied Sc., pp. 1-20.

[3] M. Perkusich, L. Chaves e Silva, A. Costa et al (2020). Intelligent software engineering in the context of agile software development: a systematic literature review. Information and Software Technology, vol. 119, pp. 106241-106247.

[4] Ijaz F, Aslam W.(2019). Identification of dependencies in task allocation during distributed agile software development, Sindh Univ Res J SURJ (Sci Ser), vol. - 51, no. - 01, pp.31-36.

[5] Zhe Wang (2019). Estimating Productivity in a Scrum team: A Multi Agent Simulation, In Proceedings of the 11th International Conference on Computer Modeling and Simulation (ICCMS 2019), ACM, New York, NY, USA, pp.239-245.

[6] M. Simão Filho, P.R. Pinheiro, A.B. Albuquerque (2018). Verbal Decision Analysis Applied to the Prioritization of Influencing Factors in Distributed Software Development, Springer International Publishing, pp.49-66.

[7] Zhe Wang (2018). The Impact of Expertise on Pair Programming Productivity in a Scrum Team: A Multi-Agent Simulation, 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), Beijing, China, pp. 399-402,

[8] Aslam W, Ijaz F (2018). A quantitative framework for task allocation in distributed agile software development. IEEE Access, vol-6, pp.-15380-15390.

[9] M. Simão Filho, P.R. Pinheiro, A.B. Albuquerque, (2017). Task assignment to distributed teams aided by a hybrid methodology of verbal decision analysis Software IET, Wiley online library, vol. 11, no. 5, pp.- 245-255.

[10] A. Aslam, N. Ahmad, T. Saba, A. S. Almazyad, A. Rehman, A.Anjum, and A. Khan (2017). Decision Support System for Risk Assessment and Management Strategies in Distributed Software Development. IEEE Access, vol. 5, pp. 20349- 20373.

[11] S. Amjad, N. Ahmad, T. Saba, A. Anjum, U. Manzoor, M. A. Balubaid, and S. U. R. Malik (2017). Calculating Completeness of Agile Scope in Scaled Agile Development. IEEE Access, vol. 99, pp. 1-12.

[12] Masood Z., Hoda R., Blincoe K. (2017). Exploring workflow mechanisms and task allocation strategies in Agile software teams, International Conference on Agile Software Development, XP 2017: Agile Processes in Software Engineering and Extreme Programming, pp 267-273.

[13] W. Aslam, F. Ijaz, M. I. Lali, and W. Mehmood (2017). Risk Aware and Quality Enriched Effort Estimation for Mobile Applications in Distributed Agile Software Development. Journal of Information Science and Engineering, vol. 33, no. 6, pp. 1481-1500.

[14] M. Simao Filho, M, Pinheiro, P.R., Albuquerque, A.B. (2016). Task Allocation in Distributed Software Development aided by Verbal Decision Analysis. 5th Computer Science On-line Conference 2016 (CSOC2016), Proceedings of the 5th Computer Science On-line Conference, pp.127-137.

[15] J. Lin, H. Yu, Z. Shen (2014). An Empirical Analysis of Task Allocation in Scrum-based Agile Programming. International school of computer science Vol.-1, pp. 111- 121.

[16] A. B. Marques, J. R. Carvalho, R. Rodrigues, T. Conte, R. Prikladnicki, and S. Marczak (2013). An ontology for task allocation to teams in distributed software development. IEEE 8th International Conference. Of Global Software Engineering, Aug. 2013, pp. 21-30.

[17] Lamersdorf A., Münch J. (2010). A multi-criteria distribution model for global software development projects. The Brazilian Computer Society, pp. 321- 329.