A Review on Random Forest Measurements To Assess And Predict Student Learning of Software Engineering Teamwork

[1]Karthik Suresh, [2]Jibin Biju, [3]Karthika Santhosh

[1]MCA student, [2]MCA student, [3]MCA student
[1]Department of Computer Science,
[1]Santhigiri College of Computer Sciences, Vazhithala, Thodupuzha, India

Abstract: The overall thing of our Software Engineering Teamwork Assessment and Prediction (SETAP) design is to develop effective machine- literacy- grounded styles for assessment and early prediction of pupil learning effectiveness in software engineering teamwork. Through this paper we briefly present a review on SETAP using Random Forest algorithm done at San Francisco State University (SFSU), Fulda University, Florida Atlantic University (FAU). These data are grouped into 11 time intervals, each measuring important phase of design development during the class. Results show that they're suitable to descry pupil teams who are bound to fail or need attention in early class time with good delicacy also the variable significance analysis shows that the features with high prophetic power. These measures can be used to guide preceptors and software engineering directors to insure early intervention for teams bound to fail.

Index Terms: software engineering teamwork, prediction, machine learning, education .

I. INTRODUCTION

There's now a agreement across industry and academia that to be successful in moment's plant, computer wisdom scholars and software masterminds must learn and exercise effective software engineering teamwork skills. This need is substantiated by the unacceptably high prevalence of failures of software systems in industry about 9 are abandoned, about one third fail, and over half experience cost and schedule overruns. These design failures supposedly stem from failures in communication, association and cooperation aspects of software engineering. The emergence of global software development systems exercising geographically distributed teams adds significant difficulty to prostrating these failure points. For the education community, though it is clear where the problem lies, little is known about the factors that influence factual pupil literacy of software engineering teamwork skills or about how to objectively and quantitatively assess, cover and predict pupil progress in the accession of these skills. This knowledge, especially the knowledge of the factors that most influence or stylish prognosticate learning effectiveness, will enable preceptors to more and more efficiently assess and ameliorate software engineering education and classroom practice and apply early classroom intervention when necessary. For industry, this knowledge will profit design directors to improve software engineering design operation. The Software Engineering Teamwork Assessment and prediction (SETAP) design, led by San Francisco State University (SFSU) with collaborators at Fulda University, Germany (Fulda) and Florida Atlantic University (FAU), addressing this need by using the Random Forest (RF) machine literacy (ML) bracket system for assessment, prediction, and most importantly discovery of factors determining the prediction of learning effectiveness of software engineering teamwork in the educational setting. In this exploration the effectiveness of literacy software engineering teamwork is defined as an capability of a pupil team (i) to learn and effectively apply software engineering processes in a teamwork setting, and process element (ii) to work well in developing satisfactory software product (product element). ML has been used in education for other analogous operations similar as predicting pupil powerhouse rate, tutoring effectiveness, grades etc. Machine literacy can be used in education to discover models that can help in understanding or predicting some aspects of educational situations, to give some characterization of the tutoring or literacy process, or to help in generating tools for education. ML ways are applied on private (e.g. checks) or objective e.g. pupil age) data uprooted from an educational terrain. For illustration, a class in a semester may yield data of the demographics of scholars, check responses of the scholars, enrolment and academic data, pupil exertion and grades. Paired with singly attained outgrowth assessments, these data constitute so- called "training databases". ML systems are also trained on those training databases, and tested in terms of their capability to rightly predict or mimic singly attained issues on variables under disquisition similar as grading, powerhouse rate, learning achievement etc..

Though ML styles (frequently RF) have been applied to software engineering, we are not apprehensive of any major work using ML to predict teamwork literacy issues in software engineering. In this paper we give first RF teamwork prediction and factor analysis results on their full data set which covers over 4 times of their common software engineering classes, conducted from Fall 2012 through Fall 2015. These classes constitute 74 pupil teams of over 350 scholars. They attained over 30000 separate data particulars used to produce their ML training database. In this review we compactly unfold on their styles of data collection and description of the data, and also show prediction delicacy results of RF analysis applied to this full data set together with ranking of team activity measures (TAMs) offering the most prophetic power

II. LITERATURE SURVEY

The application of knowledge mining widely spread in education system. This area in education domain there are many the researchers and authors are explored and discussed various applications of knowledge mining in education. The authors had skilled

the survey of the literature to know the importance of knowledge mining application in education, the utilization of knowledge mining to research scientific questions within educational research for the standard improvements during this area.

Effective Tutoring of teamwork skills in original and encyclopedically distributed Software Engineering (SE) brigades is honored as an important part of the education of current and unborn software masterminds. Effective styles for assessment and early prediction of learning effectiveness in SE cooperation aren't only a critical part of tutoring but also of value in artificial training and design operation. This paper presents a new logical approach to the assessment and most importantly, the prediction of learning issues in SE cooperation grounded on data from our joint software engineering class coincidently tutored at San Francisco State University (SFSU), Florida Atlantic University (FAU) and Fulda University, Germany (Fulda). The approach focuses on assessment and prediction of SE team work in terms of capability of pupil teams to apply stylish SE processes and develop SE products. It differs from being work in the following aspects.

a) it develops and uses only objective and quantitative measures of team activity from multiple sources, similar as statistics of pupil time use, software engineering tool use, and educator compliances

b) it leverages important machine literacy (ML) ways applied to team activity measures to identify quantitative and objective factors which can assess and predict literacy of software engineering teamwork skills at the team position.

In this paper, give the following benefactions a) Present in detail for the first time the full team activity dimension data set we developed, conforming of over 40 ideal and quantitative measures uprooted from pupil teams working on class systems. b) Present a ML frame which applies the Random Forest (RF) algorithm to the team activity measures and team issues, fastening on predicting teams that are likely to fail. c) Describe in detail our now completely enforced and functional data processing channel, conforming of data collection styles from multiple sources, ML training database creation, and ML analysis subsystems. d) Present veritably primary results of ML analysis results grounded on the data from our joint software engineering classes in Fall 2012, and Spring 2013, with the data from 17 pupil brigades. While ML training database is presently small, it continuously grows. The primary results, vindicated with two independent delicacy measures, show that RF is suitable to predict SE Process and SE Product platoon performance in intimately resolvable [1].

Next Research Full Paper builds on former exploration in Software Engineering (SE) Teamwork Assessment and Prediction design (SETAP) where we used Random Forest (RF) classifier to predict with over 70 delicacy the pupil learning effectiveness in software engineering teamwork grounded on 115 ideal and quantitative Team Activity Measures (TAM). These TAM measures have been attained from monitoring and measuring conditioning of 74 pupil brigades during the creation of their final class design in a joint software engineering classes which ran coincidently at three universities (San Francisco State University, Fulda University and Florida Atlantic University) over the period of four times and, together with team issues, have been collected in intimately available SETAP database. This paper, give important deeper analysis of how and why RF made its opinions. Also give in- depth analysis of differences in grading of original and global pupil teams (composed of scholars from multiple seminaries), also use these perceptivity to give concrete and practical guidance to preceptors tutoring SE team work in original and global classroom setting [2].

Software engineering is a competitive field in education and practice. Software systems are crucial rudiments of software engineering courses. Software systems feature a emulsion of process and product. The process reflects the methodology of performing the overall software engineering practice. The software product is the final product produced by applying the process. Like any other academic sphere, an early evaluation of the software product being developed is vital to identify the threat brigades for sustainable education in software engineering. Guidance and educator attention can help overcome the confusion and difficulties of low performing teams. The coming study proposed a mongrel approach of information gain point selection with a J48 decision tree to predict the foremost possible phase for final performance prediction. The proposed fashion was compared with the state-of- the- art machine literacy (ML) classifiers, naïve Bayes (NB), artificial neural network (ANN), logistic retrogression (LR), simple logistic retrogression (SLR), repeated incremental pruning to produce error reduction (RIPPER), and successional minimum optimization (SMO). The thing of this process is to predict the teams anticipated to gain a below-average grade in software product development. The proposed fashion outperforms others in the prediction of low performing brigades at an early assessment stage. The proposed J48- grounded fashion outperforms others by making 89 correct predictions [3].

Teamwork plays an essential part in determining the outgrowth of software engineering systems, especially when software is being developed by large teams in geographically distributed surroundings. To understand the successful development of these types of systems, it's important to assess the needed teamwork skills that would help in resolving possible problems and avoiding failure, still it's still not clear how to assess teamwork skills. The coming paper propose an logical frame grounded on a machine learning algorithm to study teamwork skills and factors that impact the success/ failure of software engineering systems. For this purpose, conduct the study on the Software Engineering Teamwork Assessment and Prediction (SETAP) dataset using a machine learning algorithm to prize the applicable features. The dataset provides quantitative data of team activity measures related to the software engineering process and the product at the different software development lifecycle phases. The results show that each of the software lifecycle phases requires different teamwork skills. The results demonstrate the effectiveness of the approach, that has predicted team issues by delicacy score lesser than 90 for process and product data [4].

III. SETAP DATA COLLECTION AND CREATE DATABASE

SETAP data are attained from a joint software engineering class tutored coincidently at SFSU, Fulda and FAU, where pupil teams at all three schools are “embedded and observed” in as realistic design and teamwork development terrain as possible, therefore furnishing a rich literacy terrain for scholars and further realistic data for experimenters. The class now involves about 140 scholars each time, working in 25- 30 teams of 5- 6 scholars each. Original pupil teams are composed of scholars from the same university, and global pupil teams are composed of levy scholars from multiple — generally two — universities. Each pupil team develops the same software operation. The semester is divided into five formally managed milestones, M1 through M5, which are accompanied across all three schools (Table I) All pupil teams use the same software development and

communication tools (source law operation, development and deployment software and waiters), which are hosted on an Amazon Web Service cloud instance, and managed by the SFSU platoon. Details about association and data collection in their common SE class have been reported before. Data collection and analysis is done at SFSU.

TABLE I. STUDENT PROJECT MILESTONE DESCRIPTIONS

Milestone	Description
M1	high level requirements and specs
M2	detailed requirements and specs
M3	prototype development
M4	beta launch
M5	final delivery and demo

The SETAP ML training database used to train and develop the RF prophetic model is a critical element of the design. The most time consuming tasks in this design were creating, curating and maintaining this database. This forced them to pay the utmost attention and significant resources to ensuring data delicacy and validity. The final outgrowth of this work is a dependable training database.

The data are organized by pupil teams and milestones and comprise TAM data for each pupil platoon paired with separate evaluations of software engineering teamwork literacy issues, one for software engineering process and one for software engineering product. These issues, for the purposes of this exploration, are distributed in two ML classes "A", represents pupil work at or above prospects, and "F", represents pupil work below anticipation or demanding attention. The grades A and F are thus considered ML class markers whose prediction they're aiming for. Note that these grades markers are a different from pupil class grades and are deduced by applying cut offs to pupil teams chance grades for process and product. To cover pupil privacy, the ML training database contains no collectively identifiable pupil information.

To concentrate their analysis only on factors impacting teams success displayed during the class and minimize the influence of an individual pupil's experience and skills developed previous to the class, pupil teams were formed with roughly the same overall combination of skills and experience. They form pupil teams by using a team placement check with about 20 questions about pupil experience, and a simple programming test, which they also dissect. The analysis provides the skill criteria that are also used to produce teams similar that the skill profile in each team is roughly equal. Individual pupil skills and gests aren't included in TAMs.

Team leads are chosen from a levy pool of scholars in the class. Implicit team leads are compactly canvassed and chosen by the educator. Teams must authorize of the preceptor's choice of team lead before final appointment. There are several issues of possible bias that they had to address. In order to reduce essential bias, where preceptors grade and at the same time try to use ML to predict grades, two ways were used. For software engineering product grading they involve reviewers external to the class, generally two. Also, grading rubrics have been cooked that in general have more and different particulars from what they measure in TAMs.

TAM data consists of added up individual student activity measures (SAM) from each team. SAM and TAM data are collected use several styles similar as

a) Weekly Time Card Surveys (WTS) these obligatory checks collect information from each pupil about the time spent during the week on rendering, meetings, teamwork etc.

b) Tool Logs (TL) comprise the collected statistics of individual pupil operation of software engineering communication and development tools similar as law depository.

c) Instructor Observation (IO) logs of team exertion similar as team member participation, the number of issues taking educator intervention, number and percent of issues closed late etc..

TABLE II. TIME INTERVAL TO MILESTONE CORRESPONDENCE

Time Interval	Corresponding Milestone
T1	M1
T2	M2
T3	M3
T4	M4
T5	M5
T6	M1 – M2
T7	M1 – M3
T8	M1 – M4
T9	M1 – M5
T10	M3 – M4
T11	M3 – M5

The ML analysis is performed on different time intervals numbered T1-T11 (Table II), which correspond to the five predefined milestones M1- M5 times and groupings of them. Grouped milestones are intended to find different trends and dynamics during the lifecycle of the pupil systems. For illustration, T6 corresponds to M1 and M2 – covering early high level conditions through detailed specs, or T11 which covers M3 – M5 covering performance, testing and delivery. ML analysis is applied singly to each

time interval. Special focus was placed on interpretation of early time intervals (T1, T2, T3, T6) due to their thing of early prediction.

Fig 1.Data flow of the ML portion of SETAP project

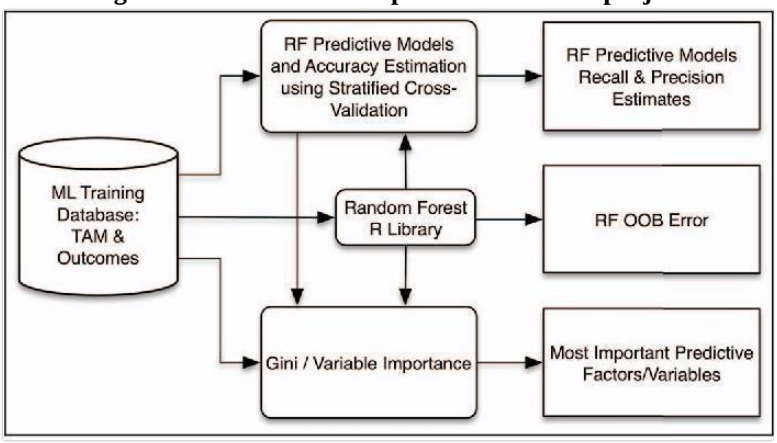
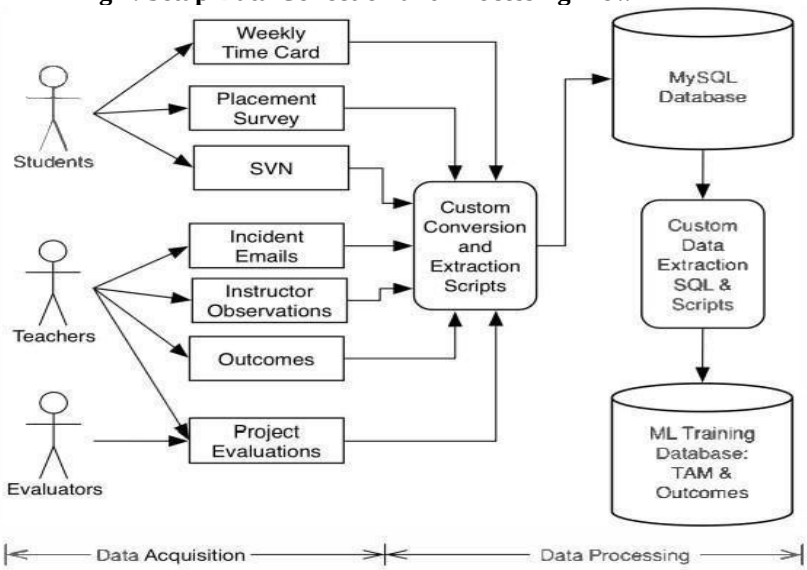


Fig 2. Setap Data Collection and Processing Flow



They added over 10 new measures include SETAP data collection and processing is described in Fig 1. For each team 115 TAMS were calculated from SAMs for every time interval. Many of these TAMs are pars and standard diversions derived from core values, analogous as hours spent in person in meetings, number of commits etc.. over intervals of weeks, pupil, or time interval. They list also only the core values.

General TAMs:

Year, semester, timeInterval, teamNumber, semesterId, teamMemberCount, femaleTeamMembersPercent, teamLeadGender, teamDistribution Weekly Time Cards (WTS) TAMs:

teamMemberResponseCount, meetingHours, inPersonMeetingHours. nonCodingDeliverablesHours, codingDeliverablesHours, helpHours, globalLeadAdminHours, LeadAdminHoursResponseCount, GlobalLeadAdminHoursResponseCount

Tool Logs (TL) TAMs: commitCount, uniqueCommitMessageCount, uniqueCommitMessagePercent, CommitMessageLength

Instructors' Observations (IO) TAMs:

issueCount, onTimeIssueCount, lateIssueCount

For each of the TAMs "core" variables e.g. CommitCount, where applicable they cipher several TAMs. This is done, for illustration, by calculating average for the week independently by the team and also by the pupil, also by computing of standard divagation over daily and pupil pars in each team (the ultimate serving to show dynamics of intergroup participation). The core variable name also gets labels pre-and-post-pended constantly, by use of formal picking alphabet, to reflect specific aggregation system and statistical measures. For example variable CommitCount is added up by week for each pupil to come CommitCountByWeek also its average and standard divagation are reckoned to yield final TAM variables standard divagation CommitCount By Week. These names are made to be fluently read by humans and are constantly used in all data fields. validations and in final training data lines.

To complete the ML training DB, TAMs for each team are paired with two ML class labels, one for software engineering Process (A or F) and one for software engineering Product (A or F). Their thing is to try to predict circumstances of F for software engineering process and software engineering Product especially in early stages of the class (time ages T1, T2, T3, T6). To apply creation of ML training database they developed a complex data collection structure. At its heart a master SETAP MySQL relational DB that first collects all raw (SAM) data from colorful sources (WTS, TL, IO). Devoted scripts extracts data from daily on- line

time card checks (WTS) and tool logs (TL) databases. Some values similar as preceptors compliances (IO) are entered manually from paper forms used in the class. To aggregate SAM into TAMs for each team and in each asked time interval, they created devoted SQL law using its statistical functions. uprooted TAMs are paired with class labels A and F and stored back in the master SETAP DB as a final training database table. A custom Python script also exports training database data for chosen time interval into CSV lines ready to be used by ML analysis software. Each of these CSV lines has expansive mortal- readable title information automatically generated for data provenance and operation. Eventually, ML analysis uses the Random Forest machine literacy package for the R statistical mathematics program to perform RF analysis.

Guided by their experience and the complexity of data collection, as well as criticality of good ML training DB, they decided to pay utmost attention to data delicacy and validity. This was achieved by several software engineering "stylish practices" including a) testing of all data gathering, aggregation and birth software with real and synthetic data. b) homemade spot checking of data by two independent experimenters. c) dealing with NULL or missing data in applicable ways (some records are dropped, some are handled by applicable statistics and some are imputed grounded on specific ways variables were uprooted). In addition, given that they also intend to circulate their training database for others to use, they designed expansive mortal- readable attestation integrated with the lines themselves as train heads for attestation, ease of operation and data provenance.

The current training data is collected from 74 pupil teams from Fall 2012 through Fall 2015 from their common software engineering classes. This data involves 383 scholars and 18 class sections. For each team 115 TAM measures have been added up from affiliated SAM measures. Total number of grades for software engineering Process were 49 As and 25 Fs, and for software engineering Product 42 As and 32 Fs. For each team they collected about 400 data particulars (pupil team selection check, time cards, deliverable shadowing, grading of issues etc.), hence their training DB involves about 30000 data points. In two semesters, for T1 and T4 intervals they had to drop some teams do to missing time card checks.

IV. RF METHODS

They use RF as their ML technology, which they've also tested successfully on other operations and they designed trials to be harmonious with their other gests in using ML for bioinformatics. RF is an ensemble classifier, conforming of a set of CART (decision tree) classifiers, each of which is generated by the Bagging algorithm. To train a RF, two parameters, the number of CARTs (ntree) and the number of aimlessly named features used to estimate at each CART node (mtry), must be supplied, as well as a training database with ground- verity class markers. RF also allows adaptation of the voting threshold or cutoff(bit of trees demanded to bounce for a given class), which they've exploited in this study.

One of the RF algorithm's strengths, and reasons they chose it, is its capability to calculate the variable importance (VI) measure, videlicet Mean decrease Gini (MDG), to determine rank of each RF input variable (in their case TAM) grounded on its donation to the RF prediction. MDG represents variable-wise information gain equaled over all decision trees included in a RF classifier. During the RF's training, each CART is erected by iteratively expanding a tree node by opting the stylish single variable threshold function to split training data to gain utmost information. This information gain is quantified by drop of Gini contaminations between ahead and after the data split. This Gini drop is also associated with the variable chosen for that knot. When a tree structure is completed, The Gini decreases from all tree nodes are added up for each variable. These variable-wise added up Gini diminishments are also equaled over the trees, yielding the MDG measures. Eventually variables are ranked according to MDG values indicating their significance to RF prediction/ classification. The delicacy estimate erected into the RF algorithm and all its software executions, and recommended by formulators of RF is called Out of Bag Error (OOB), which measures the average misclassification rate. We compound our report by also calculating recall and perfection for our target class F, delicacy (1 – OOB), and confusion matrices. In order to apply RF for their study, after evaluation, they chose statistical package R namely its Random Forest package.

V. RESULT AND DISCUSSION

Their trials and specific questions they seek to answer are twofold a) Determine Prediction Accuracy How accurate are we in predicating software engineering process and software engineering product class labels, specifically in the target class F? In which time intervals is the stylish delicacy achieved? This vaticinator delicacy (OOB, delicacy, recall and perfection for F) are estimated by performing RF training and delicacy estimation using R package by varying the RF parameters as follows ntree = 1000; mtry = {} and by varying advancing arrestment threshold as{ 10, 20, 25, 30, 35, 40, } to adjust optimal RF perceptivity for their requirements e.g. favoring F class discovery. This is repeated for software engineering Product and for each time interval T1, T2, T3, T6, T9 and T11. They also repeated each trial several times with different arbitrary seed, observing only veritably minimum changes. Results are shown in Tables III .

b) Discover factors that contribute to prediction: For the above optimal RF prophetic models (e.g. operating points with minimal delicacy) they cipher stylish ranked TAM variables using Gini measures as handed by R package, and probe if they've an intuitive explanation grounded on preceptors ' or any other experience. These factors (e.g. top ranked TAMs) can serve as a guidance to interpreters.

TABLE III DELICACY RESULTS FOR SOFTWARE ENGINEERING PROCESS AND PRODUCT.

Teamwork Component	Time Interval with best prediction	OOB (ntree, mtry, cutoff)	Overall Accuracy	Recall for F	ecision for F

SE process for T2	Predicted A	Predicted F
True A	33	16
True F	6	19

SE Process	T2	0.30 (1000,20, 35%)	0.7	0.76	0.54
SE Product	T3	0.29 (1000, 30, 40%)	0.71	0.81	0.61

TABLE IV. CONFUSION MATRIX FOR SE PROCESS T2.

TABLE V. CONFUSION MATRIX FOR SE PRODUCT T3.

SE product for T3	Predicted A	Predicted F
True A	26	16
True F	6	26

They've contributed toward demonstrating the success of ML (videlicet RF) to predicate the teams that are likely to fail in software engineering educational environment. They also show that this can be done grounded on easy to measure objective and quantitative variables, and can be done beforehand in the class or design which offers great advantages. also, they show that factors contributing to these predictions are intuitive and offer practical guidance to preceptors and directors in software engineering. Further work remains in deeper understanding of why RF workshop. This can be formulated in a number of questions a practitioner (non ML expert) might ask What would be hit on delicacy if we use much lower TAMs so I can reduce the cost of applying this approach? How exactly top ranked variables contribute e.g. does more helping time or lower helping time indicate good team? What variables most frequently mutually interact to produce correct opinions? They are laboriously engaged in this work. They also cannot overemphasize the significance of proper data collection, data operation and testing, which took a lot of their trouble and needed utmost focus.

VI. CONCLUSION

In this paper we presented, the review details of the SETAP Project's full data description, data collection and first primary ML analysis styles and results. The whole system (styles, algorithms, software) for data collection, creation of ML training database and ML analysis is completely functional and stationed for nonstop data collection and analysis from the concurrent SE classes with over 140 scholars in 25 to 30 teams each time. Out primary results show thickness when tested with two independent delicacy measures and prediction intervals indicated by their ML approach offer intuitive explanation. Now, we can conclude that Random Forest is one of the stylish ways with high performance which can be used to assess and prognosticate pupil literacy of software engineering teamwork for its effectiveness. It can handle double, nonstop, and categorical data. Overall, arbitrary RF is a simple, flexible, and robust.

VII. FUTURE SCOPE

Random Forest is one of the stylish ways with high performance which can be used to assess and prognosticate pupil literacy of software engineering cooperation for its effectiveness. It can handle double, nonstop, and categorical data . Overall, RF is a , simple, flexible and robust. In future accuracy can be calculated on actual data for different organizations by modifying or changing anyone can apply different methods to know the best method suited for education domain.

VIII. REFERENCES

1. Dragutin Petkovic, Marc Sosnick-Perez, Shihong Huang, Rainer Todtenhoefer, Kazunori okada, Swathy Arora, Ramasubramanian sreenivasen, Lorenzo Flores, Sonai Dubey "setap: Software engineering teamwork assessment and prediction using machine learning" 2014.

2. Draguntin Petkovic, Sabiha H Barlaskar, Jizhou Yang, Rainer Todtenhoefer " From explaining how random forest classifier predicts learning of software engineering teamwork to guidance for educators" 2018
3. Mehwish Nazeer, Wu Zhang, Wenhao Zhu "Early prediction of a team performance in the initial assessment phases of a software project for sustainable software engineering education" 2020
4. Mohammed Amine Beghoura "Software enginnering Teamwork Data Understanding using an Embedded Feature Selection" International journey of Performability Engineering 17 (5) 2021