Software Defect Prediction Using Supervised Learning Algorithm and Unsupervised Learning Algorithm

Anuradha Chug[1], Shafali Dhall[2]

[1,2]USICT, GGSIPU, New Delhi
[1]a_chug@yahoo.co.in, [2]shafalidhall@gmail.com

ABSTRACT
Software defect prediction has recently attracted attention of many software quality researchers. One of the major areas in current project management software is to effectively utilize resources to make meaningful impact on time and cost. A pragmatic assessment of metrics is essential in order to comprehend the quality of software and to ensure corrective measures. Software defect prediction methods are majorly used to study the impact areas in software using different techniques which comprises of neural network (NN) techniques, clustering techniques, statistical method and machine learning methods. These techniques of Data mining are applied in building software defect prediction models which improve the software quality. The aim of this paper is to propose various classification and clustering methods with an objective to predict software defect. To predict software defect we analyzed classification and clustering techniques. The performance of three data mining classifier algorithms named J48, Random Forest, and Naive Bayesian Classifier (NBC) are evaluated based on various criteria like ROC, Precision, MAE, RAE etc. Clustering technique is then applied on the data set using k-means, Hierarchical Clustering and Make Density Based Clustering algorithm. Evaluation of results for clustering is based on criteria like Time Taken, Cluster Instance, Number of Iterations, Incorrectly Clustered Instance and Log Likelihood etc. A thorough exploration of ten real time defect datasets of NASA[1] software project, followed by various applications on them finally results in defect prediction.

KEYWORDS
Decision Trees (J48), Random Forest (RF), Naive Bayesian (NB), K-means (KM), Hierarchical Clusterer (HC), Make Density Based Clusterer (MDBC).

1. INTRODUCTION

It is said by Lord Kelvin [4] “When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in number, your knowledge is meager and unsatisfactory kind; it may be the beginning of knowledge, but you have scarcity in your thought, advanced to the stage of science”. Or in short, “If you can’t measure it, you cannot manage it”. Managing data is very essential. When data is not well

Managed, it may create defects. Maximum number of defects arrives from only 20 percent of the module. More than half of the module is non-defective. Sometimes even 90 percent of the downtime arrives from at most 10 percent of defects [3].To check the defects in the modules we can use prediction models. A prediction model is effective only if it is able to classify defective and non defective modules accurately. Data which is managed through data mining proves to be of much use. Data mining is extracting/clipping data from large databases. Learning can be of two types: Supervised learning and Unsupervised learning. In Supervised learning data is extracted using the target class, where as in unsupervised learning, there is no previous information and everything is done dynamically. Clustering comes under unsupervised learning and Classification comes under supervised learning. In this paper, both the methods of learning were studied on the basis of NASA Metric Data Program[1] data set. These data sets are used majorly in software defect prediction. The data used in this study was taken from the NASA Metrics Data Program (MDP) repository[1], which contains 13 data sets used for software metrics research. Each of these data sets contains Defect Module which is a great help in prediction. Out of these 13 we used only 10 datasets which includes CM, KC1, KC3, MC1, MC2, PC1, PC2, PC3, PC4 and PC5. These 10 datasets were processed using both supervised learning and unsupervised learning. In case of supervised learning various classification algorithms, decision trees based on J48, Random Forest and Naive Bayes algorithms were explored for software defect diagnosis. Then the other unsupervised learning algorithms were explored .As an example of unsupervised learning, clustering with the help of k-means, Hierarchical Clustering and Make Density Based Clustering algorithm was performed to find out the defect prediction in the modules. We use 10 fold cross validation test for performance evaluation on NASA[1] dataset using Weka tool[6]. This tool is a collection of different machine learning algorithms and it is also used for performance evaluation of the algorithms above. When classification algorithms are applied on the datasets, results are evaluated using certain performance evaluators like Recall, Precision, F-measure, ROC, Mean Absolute Error, Root Mean Square Error, Relative Absolute Error and Accuracy. These evaluators help in predicting results on the basis of confusion matrix. Similarly clustering algorithms when applied on datasets, results evaluators are Time Taken, Cluster Instance, Number of Iterations, Within Cluster Sum of Squared Error, Incorrectly Clustered Instance and Log Likelihood. Much of the work is done on classification using different machine learning algorithms but very less work is done on different types of clustering on similar data sets. Different clustering algorithm predict different variety of results on similar dataset, so it is not very easy task to predict the defects in modules but still we have managed to predict the results based on different result evaluators. For any experiment to be implemented, we need certain things like dataset for processing, a processing method and a conclusion report. In this paper dataset used is from NASA MDP[1], Weka tool[6] is used for processing and conclusion is based on certain result evaluators.

This paper is organized as follows: Next Section reviews the key points of related work in this domain. Then there is a section that explains the dataset used in our study available from NASA MDP repositories[1] which are available publicly. The description of the dataset is also provided in table1. After the data set used, Research methodology section discusses supervised learning and unsupervised learning in the subsections. It also explains the various methods used for classification and clustering. Finally in experimental setup defect prediction results were evaluated using different evaluators in table2 and table3 and the work is concluded in the last section.

2. RELATED WORK

Fault-Proneness and Estimation Models were compared in paper titled "Comparing Fault-Proneness Estimation Models" .This paper concluded that in the past few years, the most essential requirements in software development is software quality and fault proneness estimation could also play a vital role in controlling quality of software products [7]. The methods which they used in their study were logistic regression and discriminate analysis. In the paper "Fault Prediction Using Early Lifecycle Data" Jiang presented the application of machine learning algorithms in software quality estimation using metrics available in the initial development lifecycle [8]. Their study was carried out using software measurement dataset of three NASA projects which are JM1, CM1 and PC1. In that paper, they compared requirement based models and code-based models and then the models that merge requirement metric and code metrics. Such comparison was based on their performance and the prediction models were then used for prediction of defect-prone modules in a software project. Their paper also indicates that initial lifecycle metrics can play vital role in managing project by two methods. First is by indicating the need for the increase in quality monitoring during the development. Second is by using the models to assign validation and verification activities. Finally the conclusion was that significant increase in the effectiveness of defect prediction models is due to combing metrics that describe contrasting yet similar software artifacts. Another comparison which Jiang et al. did was to compare the predictor performances that are studied from static code features, design metrics and both on 13 NASA datasets, concluding that these metrics when used in combination predict more accurately than their individual use [11]. Ammar, Menzies, Stefano and Nikora [15] compared different classifiers like Naïve Bayes, Decision Trees, and 1-rule classifier on the defect dataset of NASA. One particular pattern was not observed but distinct predictors scored better on distinct data sets. They also proposed ROCKY classifier which outstand all the predictor models. Rattikorn Hewett also compared the performance of defect-fixing effort prediction. In this comparison was done on several machine learning algorithms like SVM, Naïve Bayes, 3-NN, 5-NN, Neural Net (NN), Decision Table, and J48 based on the defects dataset of an extensive medical software system [16].T. Menzies, J. Greenwald, and A. Frank [10] showed that there should be more focus on how the attributes are used for predictors construction than which particular attributes are used. They also showed that these defect predictors are useful and the data produce predictors with 71 percent of mean probability detection and 25 percent of mean false alarms rates. For prioritizing a resource-bound investigation of code such predictors would be valuable. Bagging was used by Seliya et al. to enhance the performance of classifiers like Naïve Bayes and C4.5 decision trees on 15 NASA defect data sets [17]. Their results show that the performance of classifiers of imbalanced classes was improved due to Bagging. Shi Zhong, Naeem Seliya, and Taghi M. Khoshgoftaar [19] showed the clustering-quality results on two datasets. They performed clustering through k-means and Neural-Gas algorithm.The Neural-Gas algorithm performs significantly better in terms of mean squared error and comparably when it comes to average purity. However, the k-means algorithm runs much faster. N. Sharma, A.Bajpai , R. Litoriya[21] in their paper compared various algorithms for clustering using weka. They compared different algorithms by using two datasets ISBSG and PROMISE.

3. DATA SET USED

The datasets used for this experiment is publicly available by the NASA IV&V MDP Metric Data Repository[1]. This repository provides the metrics that define the software artifacts from 13 NASA projects. Software artifacts including the requirements, design documentation and code of MDP projects, are generally not available to the public. This data is derived from McCabe and Halstead attributes extractors of source code [12][13]. The NASA MDP data has been widely used by the software engineering research community. In this paper ten datasets CM, KC1, KC3, MC1, MC2, PC1, PC2, PC3, PC4 and PC5 are used. Some of the metrics that are included in these datasets are:-

3.1 McCabe Metrics:

Cyclomatic Complexity Metric, Cyclomatic Density Metric, Normalized Cyclomatic Complexity Metric etc.

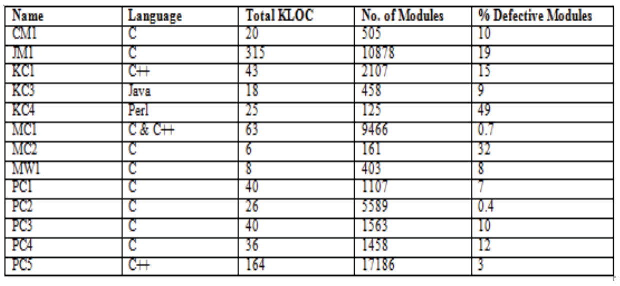
3.2 Halstead Metrics:

Halstead Length, Halstead Difficulty, Halstead Effort, Halstead Level, Halstead Error Estimate, Halstead Programming Time and Halstead Volume etc.

3.3 LOC Metrics:

Lines of Code, Number of Unique Operators, Number of Operands, LOC Blank, LOC Comment, Branch Count etc.

Table 1. NASA MDP data sets.



Name	Language	Total KLOC	No. of Modules	% Defective Modules
CM1	C	20	505	10
JM1	C	315	10878	19
KC1	C++	43	2107	15
KC3	Java	18	458	9
KC4	Perl	25	125	49
MC1	C & C++	63	9466	0.7
MC2	C	6	161	32
MW1	C	8	403	8
PC1	C	40	1107	7
PC2	C	26	5589	0.4
PC3	C	40	1563	10
PC4	C	36	1458	12
PC5	C++	164	17186	3

In this study, out of all 13 of the original NASA data sets[1] only 10 were used. These datasets are real time datasets specially used for defect prediction. The detail of data is given in Table 1

4. RESEARCH METHODOLOGY

In this paper research is carried out using two essential approaches for data mining. One is Supervised learning which is the most popular approach and the other is Unsupervised learning on which a little less work is done. Both the approaches are explained in the subsections below:

5. SUPERVISED LEARNING (CLASSIFICATION ALGORITHM)

Classification is a supervised learning method which is based on the training samples set. This popular machine learning technique is used for software defect prediction. Predictive models are build using machine learners from Weka package [6]. We used three algorithms with their default parameters:

5.1 Random Forest (RF)

Random Forest is a classifier based on decision trees which exhibits great performance in computer engineering studies by Guo et al [9]. As the name indicates, a "forest" of decision trees is build. The trees are build using the following strategy [11]:

- Each tree's root node has a sample bootstrap data which is equal to the actual data. There is a different bootstrap sample for each tree.
- Using best split method subset of variables is randomly selected from input variables.
- Each tree is then grown to the maximum extent possible without pruning.
- When all trees are built in the forest, new instances are attached to all the trees then voting process takes place to select the classification with maximum votes as the new instance(s) prediction.

5.2 Naive Bayes (NB)

Bayesian classifiers are statistical classifiers. Membership probabilities can be redirected by these classifiers and process is based on Bayes theorem. Survey on classification algorithms [14] found that simple Bayesian known as Naïve Bayesian classifier can be compared in performance with decision trees and other classifiers and exhibits high accuracy. Naive Bayes classifiers have widely been used in defect prediction, for example in [10].

5.3 Decision Trees (J48)

Decision tree is a hierarchical structure comprised of decision knots (for determining to which branch data is going to be directed) and leaves (including labels for revealing the class of data at the end of those branches) [18]. This rule based learning is a classifier in the form of a tree structure where each node is either a leaf node or a decision node. Decision trees are said to be classification trees and these trees are used to predict memberships of cases or objects in the classes of categorical dependent variable from their measurement on one or more predictor variable. The goal of decision trees is to predict or explain responses on a categorical dependent variable.

6. UNSUPERVISED LEARNING (CLUSTERING ALGORITHM)

Clustering is a typical type of unsupervised learning method which helps to identify classes or clusters for a set of data objects. It is used to group given set of objects such that objects that are similar to each other are in same group. Clustering uses software measurement datasets which are either defective or non-defective for analyzing software quality. Clustering is a technique that divides data into number of clusters depending upon some criteria. In this case data is split in two clusters depending upon whether they are defective or non-defective. Then relevant algorithms for clustering are applied on the datasets. Software defect prediction models seek to predict defective and non-defective modules. We can also analyze software quality by applying clustering algorithms.

6.1 K-means Clustering

It is a method which is used to partition n observation into k number of clusters where each observation belongs to cluster with nearest mean [2]. In this study, K-Means clustering algorithm is being used for predictive models to predict faulty/non faulty modules. K-means classify data in to different k groups (where k is a positive integer). Data grouping is done by minimizing sum of squares of distances between data and their cluster centroid. This approach is an iterative scheme which allows classification of data in to faulty and non-faulty modules based on Euclidean distance between data point.

6.2 Hierarchical Clustering

Hierarchical Clustering is a tree of clusters which are also known as dendrogram [22].

Bottom up (Agglomerative)

- Start with a single point (singleton).
- Then add more than two appropriate clusters recursively.
- Stop when desired k number of clusters is achieved.

Top down (Divisive)

- Start with a big cluster.
- Then divide into number of smaller clusters recursively.
- Stop when desired k number of clusters is achieved.

6.3 Make Density Based Clustering

In Density-Based Clustering, many partitioning methods cluster the objects on the bases of distance between objects. These types of partitioning methods can find arbitrary shaped clusters. Here the given clusters are grown as long as the number of objects or density or data points in the neighborhood exceeds some threshold. Such methods can be used to filter our noise or outliers.

7. EXPERIMENTAL SETUP

We used three machine learning algorithms and clustering algorithm from Weka tool[6] for defect prediction. We use 10 MDP datasets, where DEFECT is the predicted variable which tells whether a module has been found to contain one or more faults or not. Various measures are used in classification and clustering to evaluate the performance.

7.1 Classification Algorithm

The predictions that are made in typical defect prediction studies are usually assessed using confusion matrix related performance measures, such as Recall, RAE, RMSE, ROC and precision. Confusion matrix is a matrix that acts as a visualization which is used typically in machine learning as shown in Fig1. The confusion matrix consists of two rows and two columns that consists of True negatives, True positive, False negative and False positive. These are defined as follows: True negatives (TN) are simply the modules which are fault-free. True positives (TP) are the modules that are classified correctly as faulty modules. False negatives (FN) are faulty modules classified incorrectly as fault-free modules. False positives (FP) are the fault-free modules which are incorrectly labeled as faulty.

Predicted

Observed		**NO**	**YES**
	NO	TN(True negative)	FN(False negative)
	YES	FP(False Positive)	TP(true Positive)

Fig. 1. Confusion Matrix

Performance evaluators in Classification.

The following set of performance evaluators are being applied to find the results:

- *Recall*: Recall is ratio of modules correctly predicted as defective to number of entire module TP/(TP+FN)
- *Precision*: It determines the ratio of actual defective files compared to the number of files flagged as defective. TP/(TP+FP)
- *F-Measure*: 2*R*P/(R+P)
- *ROC*: It is tool for comparing capabilities of classification model. It plots true positive rate on Y-axis and false positive rate on X-axis.
- *Mean absolute error(MAE)*: Mean Absolute Error is the average of difference between actual and predicted value in all test cases.
- *Root Mean Square Error(RMSE)*: Root Mean Square Error is a measure of differences between values that are actually observed from thing which is being modeled or estimated and values predicted by a model or estimator.
- *Relative Absolute Error(RAE)*: It takes the total absolute error and normalizes it by dividing by the total absolute error of the simple predictor.

Accuracy: Accuracy is number of modules correctly predicted to total number of modules.

$$Acc = (TP+TN)/(TP+FP+FN+TN)$$

The performance of each of the above classification algorithm is evaluated using the software metrics data set taken from NASA database [1]. These datasets are assessed by running them through a 10-way cross validation over the J48, Random Forest and Naive Bayes. The 10-fold cross validation process randomly splits the data into 10-approximately equal fold and uses 9 of these for training the framework and 1 for testing. In this process each fold is used the same number of times for training and one time for testing. Different data set contains different number of attributes which are used in classification algorithms and based on these the rules are generated. Experimental results are reported in Table 2.

The classification was done on freely distributed tool available online WEKA machine learning toolkit [6].In this experiment machine learning algorithms were used for learning on the MDP data set[1] . The learned model is applied on test data. The outputs of various machine learning algorithms are compared on the basis MAE, RAE, RMSE, ROC values, weighted average precision, recall analysis and F-measure. The results of prediction accuracy of models by using machine learning algorithms J48, Naive Bayes and Random Forest is shown in Table 2. Each value presented in the table is the run10-fold cross-validation outcomes.

Table 2. Performance of prediction models based on various dataset

	Model	Recall	Precision	F-Measure	ROC	MAE	RMSE	RAE	Accuracy
CM1	**J48**	.855	**.836**	**.844**	.594	**0.1722**	0.3645	**79.63%**	85.46%
	NB	.823	.832	.827	.694	0.1784	0.4164	82.51%	82.26%
	RF	**.858**	.823	.836	**.725**	0.1849	**0.3235**	82.50%	**85.75%**
KC1	**J48**	.842	.813	.821	.674	0.2069	0.3661	78.88%	84.16%
	NB	.824	.816	.820	.791	**0.1759**	0.4135	**67.06%**	**82.44%**
	RF	**.851**	**.837**	**.842**	**.807**	0.1863	**0.3274**	71.03%	85.06%
KC3	**J48**	**.805**	**.790**	**.797**	.624	0.2264	0.426	76.08%	**80.5%**
	NB	.785	.773	.779	.636	**0.2176**	0.4549	**73.13%**	78.5%
	RF	.780	.750	.762	**.698**	0.253	**0.3805**	85.01%	78%
MC1	**J48**	.994	.992	.992	.827	0.0106	0.0783	72.28%	99.36%
	NB	.941	.990	.963	**.892**	0.0592	0.2404	100.27%	94.11%
	RF	**.996**	**.995**	**.995**	.891	**0.0082**	**0.0664**	**55.58%**	**99.55%**
MC2	**J48**	.630	.617	.621	.585	0.393	0.5779	86.59%	62.99%
	NB	**.732**	**.728**	**.707**	**.717**	**0.2755**	0.5191	**60.70%**	**73.22%**
	RF	.630	.613	.618	.633	0.3882	**0.4863**	85.52%	62.99%
PC1	**J48**	.909	**.893**	**.899**	.719	0.1201	0.2863	80.70%	90.90%
	NB	.883	.892	.887	.768	**0.1156**	0.3377	**77.63%**	88.27%
	RF	**.914**	.889	.897	**.806**	0.1209	**0.2595**	81.20%	**91.43%**
PC2	**J48**	.987	.980	**.984**	.448	**0.0208**	0.112	100.80%	98.73%
	NB	.955	**.984**	.968	**.878**	0.0445	0.2046	100.48%	95.45%
	RF	**.989**	.980	.984	.657	0.0181	**0.105**	**87.64%**	**98.86%**
PC3	**J48**	.858	.843	.849	.655	**0.1622**	0.3572	**74.24%**	85.77%
	NB	.358	.847	.414	.743	0.6347	0.7719	100.49%	35.82%
	RF	**.869**	**.850**	**.857**	**.814**	0.1704	**0.3027**	77.99%	**86.73%**
PC4	**J48**	.896	**.896**	.896	.773	**0.1127**	0.303	**50.64%**	89.56%
	NB	.869	.857	.862	.825	0.1339	0.3492	60.16%	86.91%
	RF	**.901**	.893	**.896**	**.908**	0.1397	**0.2679**	62.78%	**90.06%**
PC5	**J48**	.974	.971	.972	.805	0.0326	0.1522	56.75%	97.35%
	NB	.963	.966	.964	.834	0.0369	0.1912	64.13%	96.30%
	RF	**.975**	**.974**	**.975**	**.950**	**0.0302**	**0.13**	**52.56%**	**97.52%**

The best model for each data set is highlighted with boldfaced print. Results on classification can be visualized using bar graph given in Fig. 2.

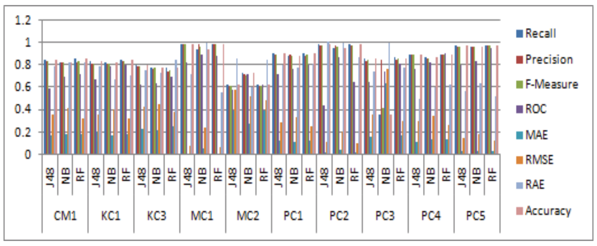


Fig. 2. Results of various parameters on NASA MDP Dataset using Classification models

For analyzing results of Fig 2, we have taken different values of Recall, Precision, F-Measure, ROC, MAE, RMSE, RAE and Accuracy for all the datasets and compared them for three models. It can be observed that out of three algorithms, Random Forest exhibits highest values of Accuracy, Recall, ROC and F-Measure in maximum number of datasets. Random Forest also gives minimum amount of Root Mean Square Error in all the cases. So we can say that there are minimum numbers of defects in Random Forest.

Clustering Algorithm

Clustering algorithms are used for grouping the software modules according to the values of their software metrics. Clustering is a process that is used to construct different groups of objects. This is done in a way that objects which are more similar than others are in the same group. Like classification clustering is also used to segment the data. Here the data is segmented into groups which were not defined previously. Clustering models do not use a target like classification. From the machine learning perspective clustering can be viewed as unsupervised learning of concepts. For clustering quality, we used Time taken to build clusters, Cluster Instances (0 for non-defective and 1 for defective), No. of iterations, Within cluster sum of squared error, Incorrectly clustered instances and Log Likelihood. A comparative analysis of dataset on three algorithms (Simple K-means (KM), Hierarchical Clustering (HC), and Make Density Based Clustering (MDBC)) is given in Table3.

Table 3. Performance of Clustering algorithms based on various dataset

	Algorithm	Time Taken(sec)	Cluster Instances(0 & 1)	No. of iterations	Within cluster sum of squared error	Incorrectly clustered instances	Log Likelihood
CM1	KM	0.08	14%&86%	10	205.2676	17.7326%	-
	HC	0	99%&1%	-	-	12.2093%	-
	MDBC	0.13	19%&81%	10	205.2676	21.5116%	-122.89047
KC1	KM	0.39	17%& 83%	19	203.6514	18.7023%	-
	HC	0.03	100%& 0%	-	-	15.5057%	-
	MDBC	0.5	22%&78%	19	203.6514	20.3244%	-63.77387
KC3	KM	0.08	83%&18%	12	174.7175	22.5%	-
	HC	0	96%&5%	-	-	18.5%	-
	MDBC	0.06	78%&22%	12	174.7175	24%	-111.19398
MC1	KM	2.45	25%&75%	9	5120.6531	24.8141%	-
	HC	0.13	100%& 0%	-	-	0.7546%	-
	MDBC	1.72	23%&77%	9	5120.6531	23.0678%	-102.36315
MC2	KM	0.02	10%&90%	3	115.7140	30.7087%	-
	HC	0.02	97%&3%	-	-	31.4961%	-
	MDBC	0.03	13%&87%	3	115.7140	29.1339%	-129.82322
PC1	KM	0.13	34%&66%	6	324.2966	37.1542%	-
	HC	0.08	100%& 0%	-	-	7.7734%	-
	MDBC	0.09	36%&64%	6	324.2966	36.3636%	-139.90952
PC2	KM	0.38	10%&90%	9	492.6716	10.7256%	-
	HC	0.31	100%& 0%	-	-	1.0726%	-
	MDBC	0.28	60%&40%	9	492.6716	39.6845%	-121.15875
PC3	KM	0.24	70%&30%	7	414.2303	35.6444%	-
	HC	0.16	100%& 0%	-	-	12.5333%	-
	MDBC	0.17	64%&36%	7	414.2303	38.6667%	-149.61368
PC4	KM	0.61	64%&36%	16	703.9360	45.8899%	-
	HC	0.39	100%& 0%	-	-	12.7949%	-
	MDBC	0.39	38%&62%	16	703.9360	33.1665%	-126.11086
PC5	KM	2.82	38%&62%	5	3689.6454	35.2097%	-
	HC	1.94	100%& 0%	-	-	2.941%	-
	MDBC	2.29	39%&61%	5	3689.6454	35.892%	-126.46888

It can be analyzed from Table3 that Hierarchical Clustering is not very efficient clustering when compared to K-Means and MDBC since in maximum cases the value of cluster instances is 100%&0% which is not ideal for clustering. When it comes to K-means and MDBC they have same number of iterations and same values of sum of squared error in all the cases. So a comparison for different clustering algorithms is also given in Fig3 which predicts defect on the basis of Time Taken (sec) and Incorrect clustered instances.

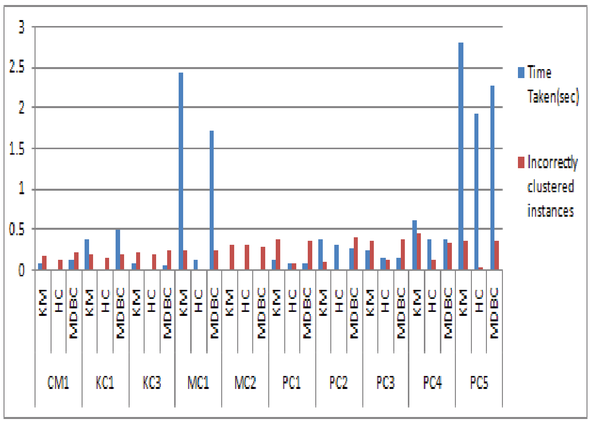


Fig. 3. Results of various parameters on NASA MDP Dataset using Clustering models

From Fig.3 it can be predicted that in maximum cases though K-means is taking more time but it is producing less number of incorrect clustered instances which is a more relevant factor in defect prediction. So when it comes to time we can say MDBC is producing better results but when it comes to incorrect clustered instance, we can say that simple K-means algorithm is exhibits relevant information.

8. CONCLUSION

Early detection of faults is very essential activity in software project development. Numbers of technique are available for this purpose. In this work three models were studied under both classifications as well as clustering algorithms for defect prediction. Results were found using NASA MDP datasets with Supervised and Unsupervised learning. Various datasets namely CM1, KC1, KC3, MC1, MC2, PC1, PC2, PC3, PC4 and PC5 were used to predict software defect using classifiers and clusters. Different measurement attributes like RAE, Accuracy, and RMSE etc in classifier (Table2) and Time Taken and Incorrect Cluster Instances in clusters (Table3) play an important role in giving correct predictions. On analyzing different classifiers, it is observed that best prediction results are obtained by Random Forest Algorithm since it has highest accuracy of 99.55% in case of MC1. Random Forest also exhibits highest values in Recall, F-Measure and ROC. It is observed from Fig.2 that Random Forest shows lowest number of Root Mean Square Errors in all the cases. When it comes to clustering it is suggested that K-means clustering algorithm is facile and simple algorithm as compared to other algorithms since it has minimum number of incorrect clustered instance and exhibits relevant information needed for clustering in minimum time which is shown in Fig.3. After analyzing various models of supervised and unsupervised learning we have predicted that defects are minimum in Random Forest(classification algorithm) and K-Means(clustering algorithm).

9. REFERENCES

[1] NASA IV &V Facility. Metric Data Program. Available from http://MDP.ivv.nasa.gov/.

[2] Jinxin Gao, David B. Hitchcock"James-Stein Shrinkage to improve K-means Cluster Anaysis"University of South California, Department of Statics November 30, 2009.

[3] B. B. and Victor R Basili, "Software defect reduction top 10 list," *IEEE*, vol. 15, no. 1, pp. 795–825, Jan 2001.

[4] N.E. Fenton and S.L Pfleeger, "Software Metrics, A Rigorous &Practical Approach", International Thomson Computer Press,London, 1997.

[5] M.A.Maloof, "On Machine learning ROC analysis and statistical tests of significance", 16th. International conference on Pattern recognition, IEEE, vol.2,pp.204-207, 2002.

[6] WEKA:http://www.cs.waikato.ac.nz/ml/weka.

[7] Bellini, P. (2005), "Comparing Fault-Proneness Estimation Models", In Proceedings of the 10th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS'05), China, pp. 205-214.

[8] Jiang, Y., Cukic, B. and Menzies, T. (2007), "Fault Prediction Using Early Lifecycle Data". In Proceedings of 18th IEEE Symposium on Software Reliability Engineering, IEEE Computer Society, Sweden, pp.237-246

[9] L. Guo, Y. Ma, B. Cukic, and H. Singh. Robust prediction of fault-proneness by random forests. In Proc. of the 15th International Symposium on Software Relaibility Engineering ISSRE'04, pages 417–428, 2004.

[10] T. Menzies, J. Greenwald, and A. Frank. Data mining static code attributes to learn defect predictors. *IEEE Transactions on Software Engineering*, 33(1):2–13, January 2007.
Availablefrom http://menzies.us/pdf/06learnPredict.pdf.

[11] Jiang, Y., Cukic, B., Menzies, T., Bartlow, N.: 'Comparing design and code metrics for software quality prediction'. Proc. Fourth Int. Workshop on Predictor Models in Software Engineering. PROMISE'08, New York, USA, 2008, pp. 11–18

[12] T. J. McCabe, “A complexity measure,” IEEE Transactions on Software Engineering, vol. 2, pp. 308–320, 1976.

[13] M. H. Halstead, “Element of software science,” in Proceedings of the 6th International Conference on Software engineering USA 1982.

[14] Han, J., & Kamber, M., “Data Mining: Concepts and Techniques”, San Francisco: Morgan Kaufmann Publishers,2001.

[15] Menzies, T., Ammar, K., Nikora, A., and Stefano, S., “How Simple is Software Defect Prediction?” Submitted to Journal of Empirical Software Engineering, October (2003).

[16] R. Hewett, and P. Kijsanayothin, “On modeling software defect repair time,” Empirical Software Engineering, vol.14, Apr. 2009, pp.165-186, doi:10.1007/s10664-008-9064-x.

[17] N. Seliya, T.M. Khoshgoftaar, and J. Van Hulsez, “predicting faults in high assurance software,” IEEE 12th International Symposium on High Assurance Systems Engineering, pp. 26–34, 2010.

[18] Amasyal_, M.F. (2008) New Machine Learning Methods and Drug Design Applications, Ph.D. Thesis, Y_ld_z Technical University _Istanbul, Turkey.

[19] S. Zhong, T. M. Khoshgoftaar and N. Seliya, “Analyzing Software Measurement Data with Clustering Techniques,” IEEE Intelligent Systems, vol.19, no.2, pp.20-27, Mar./Apr.2004.

[20] I. H. Witten and E. Frank. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, Los Altos, US, 2005.

[21] N. Sharma , A.Bajpai , R. Litoriya “Comparison the various clustering algorithms of weka tools”, International Journal of Emerging Technology and Advanced Engineering (ISSN 2250-2459, Volume 2, Issue 5, May 2012)

[22] Manish Verma, Mauly Srivastava, Neha Chack, Atul Kumar Diswar and Nidhi Gupta, “A Comparative Study of Various Clustering Algorithms in Data Mining”, International Journal of Engineering Research and Applications(IJERA) Vol. 2, Issue 3, May-Jun 2012, pp.1379-1384.