Classification of defective modules using object-oriented metrics

Satwinder Singh*
Centre for Computer Science Technology,
Central University of Punjab,
Bathinda, 151100, India
Email: satwindercse@gmail.com

Rozy Singla*
CSE Department,
Shankara Institute of Technology,
Jaipur, 302028, India
Email: rozysingla92@gmail.com
*Corresponding authors

Abstract: Software defect in today’s era is crucial in the field of software engineering. Most of the organisations use various techniques to predict defects in their products before they are delivered. Defect prediction techniques help the organisations to use their resources effectively which results in lower cost and time requirements. There are various techniques that are used for predicting defects in software before it has to be delivered, e.g., clustering, neural networks, support vector machine (SVM). In this paper two defect prediction techniques: K-means clustering and multi-layer perceptron model (MLP) are compared. Both the techniques are implemented on different platforms. K-means clustering is implemented using WEKA tool and MLP is implemented using SPSS. The results are compared to find which algorithm produces better results. In this paper object-oriented metrics are used for predicting defects in the software.

Keywords: object-oriented metrics; defect prediction; K-means clustering; artificial neural network; ANN; WEKA.


Biographical notes: Satwinder Singh is working as an Assistant Professor, Centre for Computer Science Technology, School of Engineering & Technology, Central University of Punjab, Bathinda. He has worked as an Assistant Professor in Dept. Comp. Sci. & Tech. in Baba Banda Singh Bahadur Engineering College, Fatehgarh Sahib from August 2006 to December 2015. He has worked as a Lecturer in Dept. Comp. Sci. & Tech in Rayat Institute of Engineering and Ifo. Tech. Railmajra, NawanShahar, from July 2004 to August 2006. His area of specialisation includes reengineering of software system, maintenance and fault prediction of object oriented systems, data mining.
1 Introduction

Standish Group carried a survey on cost of software projects. They concluded that the cost of software increases by 90% and the schedule of project exceeded by 222%. Data is collected from 8000 projects for this survey. According to this survey only 29% projects were successfully completed, 53% projects suffered from late delivery, over budget, etc. and 18% projects failed. After this survey it has been realised that it is important to measure the software early and necessary actions must be taken before the product is delivered. These actions may be bug finding, defect prediction, defect correction and so on. In software industry, a metrics program for software projects is usually deemed unnecessary, but when things go bad then the practitioners start to stress on these metrics programs. These metrics programs help to evaluate the performance of software before the delivery (Ramaswamy et al., 2013).

Most modern organisations are searching for a defect prediction model which can be used for any type of software but such a model is still under development. Generally the prediction models focus on the following aspects:

- finding the bugs in software system
- checking the reliability of the software against the time frame
- to grasp the effect of designing process over defects and failures.

If there is a well-established metrics program presents then cost and schedule of software project can be effectively predicted. The measured metrics give better detection during development phase of software.

One of the most popular techniques for defect prediction is ‘testing’. But it is too complicated and expensive when the size of the project grows. Clusters can be created to group the defects according to their types. If there is any relationship between software design metrics and defects’ properties, then it is possible to predict similar type of faults or defects in other parts of the software (Rawat and Dubey, 2012).

There are many methods used for defect prediction such as classification, statistical procedures, machine learning methods (artificial neural network (ANN) (Singh and Salaria, 2013), genetic algorithm (GA), etc.) and so on. Defect prediction is the most challenging part in software development life cycle. Classification is the most popular method for predicting the defects in the software. Classification is used to predict discrete or unordered labels and defect prediction is used to predict continuous valued functions. Defect prediction plays an essential role in software quality and software reliability. Defect prediction increases the cost of software development. It may not eliminate all the defects but minimises the number of defects and their impact on the reliability of the software (Sharma et al., 2012).

Fault prediction is a mechanism used in software development life cycle to reduce the software failure and is carried out mostly during initial planning to identify fault-prone
modules. Fault prediction not only increases the quality of monitoring during software development but also gives suggestions for suitable verification and validation approaches that eventually lead to improvement of efficiency and effectiveness of fault prediction (Fenton and Neil, 1999).

Present day software development is mostly based on object-oriented (OO) paradigm. The quality of OO software can be best assessed by the use of software metrics. A number of metrics have been proposed by researchers and practitioners to evaluate the quality of software (Rahman et al., 2012).

Various prediction models are used such as support vector machines (SVMs), Bayesian network, neural network and clustering.

The K-means clustering is a useful algorithm in generating good quality outcome for various realistic applications.

The K-means clustering intends to categorise \( n \) objects into \( k \) clusters in which each object belongs to the cluster with the nearest centroid serving as a model of the cluster. It is a partitional clustering approach which is simple and easy to understand. Each cluster produced, using K-means clustering, has a centroid, i.e., centre point. Every object is categorised into the cluster with the nearest centroid. The number of clusters must be specified.

There are different clustering algorithms available, such as: basic sequential algorithmic scheme, complete linkage clustering, DBSCAN, expectation–maximisation algorithm, fuzzy clustering, hierarchical clustering, mean shift, nearest-neighbour chain algorithm, single linkage clustering and spectral clustering.

2 Literature survey

Song et al. (2011) focuses on the classification of software components according to their defect-proneness, i.e., to classify them as fault-prone or non-defect-prone. They argued on how the attributes are used to build predictors and which particular attributes are used for defect prediction. They focused on which, i.e., the selection of attribute dataset used for learning or training. They designed a framework that includes a scheme evaluation and defect prediction. In scheme evaluation various learning or training strategies are used. In defect prediction stage, the best suited learning scheme is selected to build the prediction model. They used ROC (receiver operating characteristic) as performance measurement parameter to calculate AUC (area under curve). They have used 12 different learning schemes to predict defect-prone modules. They come up with the result that different learning schemes are selected for different datasets.

Fenton and Neil (1999) presented a critical review of software prediction techniques. In this review they show that size, complexity, testing and quality metrics can be used for estimating the defects. There is a direct dependency between the size metrics and defects and defects can be described as a function of size. One class of testing metrics that generates good results for predicting defects are the so-called test coverage measures. Prediction can also be done with the help of process quality metrics. The simplest process quality metrics is SEI capability maturity model (CMM) ranking. There are various other techniques that can be used for prediction such as neural network and clustering, regression which uses these metrics. They represented some issues affecting the software engineering community with respect to defect prediction. As the relationship between defects and failures are unknown, size and complexity metrics cannot be used as a sole in
the defect prediction. They proposed a prediction technique which is based on Bayesian belief network (BBN). A BBN is a graphical network which represents probabilistic relationship among variables such as relationship between defects and software metrics. The advantage of BBN is that it can predict events based on uncertain data and ability to represent and manipulate complex models. They give theoretical comparison which is not tested by them.

Ma et al. (2007) proposed a fault prediction strategy using a modified random forest method. They used five NASA datasets which vary in size. These datasets contain a little number of defect samples in the training set. They used balanced random forest as fault prediction methodology. They compare their results with data mining software packages like WEKA, See5 and SAS. Random forest algorithm builds a classification tree based on impurity (heterogeneity) function such as entropy-based function or Gini-index function. They used PD (probability of detection), PF (probability of false alarms), accuracy, TNR (true negative rate), G-mean and F-measure values for comparison. The results show that balanced random forest is better than random forest for classification.

Singh and Salaria (2013) used Software metrics such as file-level, class-level, component-level, method-level, process-level and quantitative values-level metrics to find the software defects. Various methods such as statistics, machine learning and machine learning along with statistical methods are used for calculating defects. Researchers can also apply a machine learning approach on real-time software systems for defect prediction. They used Ant 1.7 dataset from PROMISE repository for the experiment. They divide data as 85% data for training and 15% data for testing. The data is trained with Levenberg-Marquardt (LM) algorithm resulting in 88% accuracy.

Jain and Dubes (1988) defined the main steps of $K$-means algorithm as:

1. Select an initial partition with $K$ clusters; repeat steps 2 and 3 until cluster membership stabilises.
2. Generate a new partition by assigning each pattern to its closest cluster centre.
3. Compute new cluster centres. The $K$-means algorithm depends upon three user-specified parameters: number of clusters $K$, cluster initialisation and distance metric.

Choosing value of $K$ is most censorious. There are different extensions of $K$-means algorithm. The enlarged heuristics are tackled by some of these extensions that involve the minimum cluster size and merging and splitting clusters. There are two popular alternatives of $K$-means in pattern recognition literature that are ISODATA proposed by Ball and Hall (1965) andForgy (1965).

3 Experiment

3.1 Neural network based multi-layer perceptron model (NN MLP)

There are many techniques which can be used for defect prediction such as naive byes, LogReg, random forest, nearest-neighbour, SVM, machine learning. In our research, feed forward neural network is used for predicting the defects.
Feed forward neural networks, trained with a back-propagation learning algorithm, are the most popular neural networks. They are applied to a wide variety of problems. A FFNN consists of neurons, which are ordered into layers.

The first layer is called the input layer, the last layer is called the output layer, and the layers between are hidden layers. The used FFNN model is shown in Figure 1. Each neuron in a particular layer is connected with all neurons in the next layer.

**Figure 1** A multi-layer feed forward network

The connection between the $i$th and $j$th neuron is characterised by the weight coefficient $w_{ij}$. The weight coefficient reflects the degree of importance of the given connection in the neural network. The output of a layer can be determined by equations

$$a = x_1 w_1 + x_2 w_2 + x_3 w_3 + \cdots + x_n w_n$$

(1)

In this research seven neurons are used at input layer and three neurons are used at hidden layer. The seven inputs are object-oriented metrics which are: EcnF (Henderson-Sellers, 1996), DIT (Chidamber and Kemerer, 1994), NOC (Chidamber and Kemerer, 1994), Coh (Henderson-Sellers, 1996), WMC (Chidamber and Kemerer, 1994), CBO (Chidamber and Kemerer, 1994), RFC (Henderson-Sellers, 1996).

In this research, following activation functions are used:

- Hyperbolic tangent function
- Identity function

Hyperbolic tangent function is used as activation function for hidden layer and identity function is used as activation function for the output layer.
3.1.1 Hyperbolic tangent function (tanh)

It takes two real-valued arguments and transforms them to a range (–1, 1).

The equation used for tanh is

\[ a = \tanh(n) = \frac{2}{1 + e^{-2n}} - 1 \]  

(2)

3.1.2 Identity function

It takes one input and returns value \( n \).

The equation for purelin is

\[ a = \text{Identity}(n) = n \]  

(3)

3.2 K-means clustering algorithm

Flow chart is shown in Figure 2 for K-means clustering algorithm. In this clusters are created and centroids are calculated. Objects are classified by calculating their distance from centroids using a distance function.

**Figure 2** Flow chart for K-means clustering

3.3 Performance evaluation parameters

The following subsections give the basic definitions of the performance parameters used for fault prediction.
The confusion matrixes are categories into four categories (Table 1):

- true positives (TP) are the number of modules correctly classified as faulty modules
- false positives (FP) refer to non-faulty classes incorrectly labelled as faulty classes
- true negatives (TN) correspond to non-faulty modules correctly classified as such
- finally, false negatives (FN) refer to faulty classes incorrectly classified as non-faulty classes.

These are the performance parameter used to measures the classification techniques.

<table>
<thead>
<tr>
<th>Non-faulty</th>
<th>Faulty</th>
</tr>
</thead>
<tbody>
<tr>
<td>True negative (TN)</td>
<td>False positive (FP)</td>
</tr>
<tr>
<td>False negative (FN)</td>
<td>True positive (TP)</td>
</tr>
</tbody>
</table>

3.3.1 Precision

It is used to measure the degree to which the repeated measurements under unchanged conditions show the same results. It is also called as positive predictive value. Precision refers to the closeness of two or more measurements to each other. Value near one is good. Precision is related to random errors.

\[
\text{Precision} = \frac{TP}{FP + TP} \tag{4}
\]

3.3.2 TP rate (recall)

The true positive rate measures the proportion of faults correctly specified as faulty modules in the dataset and is complementary to the false negative rate. It is also called the sensitivity or recall.

\[
\text{True positive rate} = \frac{TP}{TP + FN} \tag{5}
\]

3.3.3 FP rate

The false positive rate specifies the tendency to predict the non-faulty modules as faulty.

\[
\text{False positive rate} = \frac{FP}{FP + TN} \tag{6}
\]

where FP is number of false positives and TN is number of true negatives.

3.3.4 Area under curve (AUC)

The area under receiver operating characteristics curve presents the association of true positive rate and false positive rate.
3.4 Dataset details

System chosen for defect prediction analysis is Licq (an instant messaging or communication client for the UNIX). Licq bug database was collected from the GitHub\textsuperscript{1} community (Table 2). Licq is a small system with 280 classes. One Mozilla Seamonkey Version 1.0.1 is open source system was used for analysis in this study. Whereas, the bug database required for each version was collected from the Bugzilla\textsuperscript{2} system. All errors (bugs) that have been found in the lifetime of the Mozilla Firefox project are stored in the Bugzilla database. Bugzilla database has detailed information that includes the release number, error severity and summary of errors. The descriptive analysis of both the datasets are shown in Tables 3 and 4.

<table>
<thead>
<tr>
<th>Name</th>
<th>No. of classes</th>
<th>No. of defect classes</th>
<th>% defects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Licq</td>
<td>280</td>
<td>126</td>
<td>45</td>
</tr>
<tr>
<td>Seamonkey 1.0.1</td>
<td>4103</td>
<td>47</td>
<td>1.145</td>
</tr>
</tbody>
</table>

Table 2 Dataset details

<table>
<thead>
<tr>
<th>NOC</th>
<th>RFC</th>
<th>DIT</th>
<th>WMC</th>
<th>CBO</th>
<th>LCOM</th>
<th>LCOMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.7071</td>
<td>16.5964</td>
<td>1.5929</td>
<td>24.2571</td>
<td>4.607</td>
<td>452.7214</td>
</tr>
<tr>
<td>Median</td>
<td>0.0000</td>
<td>7.0000</td>
<td>1.0000</td>
<td>4.0000</td>
<td>3.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>Std. deviation</td>
<td>4.16074</td>
<td>52.59194</td>
<td>1.39326</td>
<td>126.54807</td>
<td>12.00818</td>
<td>3936.0356</td>
</tr>
<tr>
<td>Variance</td>
<td>17.3120</td>
<td>2765.912</td>
<td>1.9410</td>
<td>16014.414</td>
<td>144.196</td>
<td>1.549E7</td>
</tr>
<tr>
<td>Minimum</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Maximum</td>
<td>57.0000</td>
<td>720.0000</td>
<td>5.0000</td>
<td>1920.0000</td>
<td>195.0000</td>
<td>45002.0000</td>
</tr>
<tr>
<td>Percentiles 25%</td>
<td>0.0000</td>
<td>3.0000</td>
<td>0.0000</td>
<td>2.0000</td>
<td>1.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>50%</td>
<td>0.0000</td>
<td>7.0000</td>
<td>1.0000</td>
<td>4.0000</td>
<td>3.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>75%</td>
<td>0.0000</td>
<td>14.7500</td>
<td>3.0000</td>
<td>12.7500</td>
<td>6.0000</td>
<td>8.0000</td>
</tr>
</tbody>
</table>

Table 3 Descriptive statistical analysis of Licq dataset

<table>
<thead>
<tr>
<th>NOC</th>
<th>RFC</th>
<th>DIT</th>
<th>WMC</th>
<th>CBO</th>
<th>LCOM</th>
<th>LCOMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>1.0846</td>
<td>29.7714</td>
<td>2.0041</td>
<td>245.1311</td>
<td>11.0943</td>
<td>44.4619</td>
</tr>
<tr>
<td>Median</td>
<td>0.0000</td>
<td>12.0000</td>
<td>2.0000</td>
<td>3.0000</td>
<td>5.0000</td>
<td>7.0000</td>
</tr>
<tr>
<td>Std. Deviation</td>
<td>17.05050</td>
<td>53.71393</td>
<td>1.79840</td>
<td>1523.63884</td>
<td>218.328</td>
<td>16269.03</td>
</tr>
<tr>
<td>Variance</td>
<td>290.720</td>
<td>2885.186</td>
<td>3.234</td>
<td>2321475.303</td>
<td>218.328</td>
<td>16269.03</td>
</tr>
<tr>
<td>Minimum</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Maximum</td>
<td>1078.00</td>
<td>770.00</td>
<td>10.00</td>
<td>54950.00</td>
<td>174.00</td>
<td>3250.00</td>
</tr>
<tr>
<td>Percentiles 25%</td>
<td>0.0000</td>
<td>4.0000</td>
<td>1.0000</td>
<td>0.0000</td>
<td>3.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>50%</td>
<td>0.0000</td>
<td>12.0000</td>
<td>2.0000</td>
<td>3.0000</td>
<td>5.0000</td>
<td>7.0000</td>
</tr>
<tr>
<td>75%</td>
<td>1.0000</td>
<td>31.0000</td>
<td>3.0000</td>
<td>48.0000</td>
<td>14.0000</td>
<td>37.0000</td>
</tr>
</tbody>
</table>

Table 4 Descriptive statistical analysis of Seamonkey 1.0.1 dataset
Table 5 defines various parameters used in K-means clustering algorithm for defect prediction. These parameters are helpful to control the operation of K-means clustering algorithm.

Table 5 Parameters used in K-means clustering

<table>
<thead>
<tr>
<th>Name</th>
<th>Functionality</th>
<th>Value set</th>
</tr>
</thead>
<tbody>
<tr>
<td>distanceFunction</td>
<td>The distance function is used to evaluate the instances of dataset</td>
<td>Euclidean distance</td>
</tr>
<tr>
<td>dontReplaceMissingValues</td>
<td>Substitute omitted values worldwide through mean/mode</td>
<td>False</td>
</tr>
<tr>
<td>maxIteration</td>
<td>It sets the maximum number of iterations that will occur during the whole clustering/classification process</td>
<td>500</td>
</tr>
<tr>
<td>numClusters</td>
<td>This option let us define the number of clusters needed in output</td>
<td>2</td>
</tr>
<tr>
<td>preserveInstancesOrder</td>
<td>This option can preserve the order of instances of dataset</td>
<td>False</td>
</tr>
<tr>
<td>Seed</td>
<td>It defines the number of seed that will be used in clustering procedure</td>
<td>10</td>
</tr>
</tbody>
</table>

4 Results and conclusions

4.1 Results using MLP

The proposed neural network model, i.e., MLP model gives TP rate up to 11%. It gives a precision rate 71% for Licq dataset (Table 6). The proposed model gives a perfect FP rate for Seamonkey 1.0.1 dataset and 4% for Licq dataset (Table 6). The highest AUC is given by Seamonkey 1.0.1 dataset, i.e., 0.826. Licq gives a value of 0.447 for Licq dataset. The ROC curves for Licq and Seamonkey 1.0.1 datasets are shown in Table 6 and Figure 3.

Table 6 Results using MLP

<table>
<thead>
<tr>
<th>Dataset</th>
<th>TP rate (recall)</th>
<th>FP rate</th>
<th>Precision</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Licq</td>
<td>0.115</td>
<td>0.04</td>
<td>0.7142</td>
<td>0.447</td>
</tr>
<tr>
<td>Seamonkey 1.0.1</td>
<td>0.0212</td>
<td>0</td>
<td>0.25</td>
<td>0.826</td>
</tr>
</tbody>
</table>

Figure 3 Results using MLP (see online version for colours)
4.2 Results using K-means clustering

4.2.1 Cluster centroids

The clusters are generated using K-means clustering algorithm are shown in Table 7. Table 7 represents centroid values of each cluster for Seamonkey1.0.1 dataset and Table 8 represents centroid values of each cluster for Licq dataset.

Table 7 Cluster centroids for Seamonkey 1.0.1 dataset

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Full data</th>
<th>Cluster 0</th>
<th>Cluster 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Centroid values</td>
<td>No. of instances</td>
<td>Centroid values</td>
</tr>
<tr>
<td>EncF</td>
<td>0.1414</td>
<td></td>
<td>0.0893</td>
</tr>
<tr>
<td>DIT</td>
<td>2.0041</td>
<td></td>
<td>2.1942</td>
</tr>
<tr>
<td>NOC</td>
<td>1.0846</td>
<td></td>
<td>1.2075</td>
</tr>
<tr>
<td>Coh</td>
<td>0.1793</td>
<td>4103</td>
<td>0.0749</td>
</tr>
<tr>
<td>WMC</td>
<td>44.4619</td>
<td></td>
<td>49.1558</td>
</tr>
<tr>
<td>CBO</td>
<td>11.0943</td>
<td></td>
<td>12.1447</td>
</tr>
<tr>
<td>RFC</td>
<td>29.7714</td>
<td></td>
<td>32.9839</td>
</tr>
</tbody>
</table>

Depending upon the minimum distance from centroid the cluster has been formed. The instance nearest to the centroid is assigned to the corresponding cluster. In this way the clusters are formed. Further, these centroid values are calculated using the Euclidean distance formulation as follow.

Euclidean distance between a point \( X (X_1, X_2, \text{ etc.}) \) and a point \( Y (Y_1, Y_2, \text{ etc.}) \)

\[
D = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}
\]  

(7)

A new centroid is generated till a new cluster is produced from the new centroid. When the same cluster is generated from the new centroid as formed by previous centroid, the final cluster is formed and no any further iteration takes place. Also the final centroid value is obtained.
The true positive rate states the rate of faults correctly specified as faulty modules in the dataset. In Seamonkey 1.0.1 dataset has predicted faulty modules most correctly, i.e., 0.83 and the probability of being wrong is 0.17.

The FP rate is false positive rate based on buggy and non-buggy classes. The Set5 has the highest false positive rate, i.e., 0.717. It is evaluated over the Seamonkey 1.0.1 dataset. This dataset shows that the cohesion has the highest tendency to predict the non-faulty modules as faulty. So the cohesion must be taken care to reduce the false positive rate and better results can be generated by the datasets.

The precision is defined as the ratio of number of modules correctly classified to be faulty to the total number of modules. As seen in Table 9 and Figure 4 the Seamonkey 1.0.1 has the highest precision of 0.979. Depending on true positive rate the precision is also maximum in case of the Seamonkey 1.0.1 dataset. Here it is specified that the coupling and complexity has to be taken care of, for maximising the predicted positive cases.

Table 9  Results using $K$-means clustering

<table>
<thead>
<tr>
<th></th>
<th>TP rate (recall)</th>
<th>FP rate</th>
<th>Precision</th>
<th>ROC area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seamonkey 1.0.1</td>
<td>0.823</td>
<td>0.717</td>
<td>0.979</td>
<td>0.547</td>
</tr>
<tr>
<td>Licq</td>
<td>0.536</td>
<td>0.49</td>
<td>0.529</td>
<td>0.523</td>
</tr>
</tbody>
</table>

Figure 4  Results using $K$-means clustering (see online version for colours)

The area under receiver operating characteristics curve provides us the relationship of true positive rate and false positive rate. ROC area is maximum for Seamonkey 1.0.1 dataset which is 0.55.

5 Conclusion

In this research paper, various parameters: TP rate, FP rate, precision and AUC are used to compare the given classification techniques. Among these four parameters, the AUC is the most widely used to evaluate the performance of classification technique. Table 10 represents the comparison of ANN and $K$-means clustering algorithm.

These results show that the MLP has better AUC values as compared to $K$-means clustering algorithm. The $K$-means clustering algorithm has improved results in case of recall or TP rate. MLP gives value 0.7142 of precision for Licq which is better than
K-means clustering algorithm which gives the value of precision 0.529 for Licq. K-means clustering algorithm results are more precise for Seamonkey 1.0.1 dataset.

Table 10  
Comparison between classification techniques

<table>
<thead>
<tr>
<th>Dataset</th>
<th>TP rate (recall)</th>
<th>FP rate</th>
<th>Precision</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>0.115</td>
<td>0.04</td>
<td>0.7142</td>
<td>0.447</td>
</tr>
<tr>
<td>Seamonkey 1.0.1</td>
<td>0.0212</td>
<td>0</td>
<td>0.25</td>
<td>0.826</td>
</tr>
</tbody>
</table>

K-means

<table>
<thead>
<tr>
<th>Dataset</th>
<th>TP rate (recall)</th>
<th>FP rate</th>
<th>Precision</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Licq</td>
<td>0.536</td>
<td>0.49</td>
<td>0.529</td>
<td>0.523</td>
</tr>
<tr>
<td>Seamonkey 1.0.1</td>
<td>0.823</td>
<td>0.717</td>
<td>0.979</td>
<td>0.547</td>
</tr>
</tbody>
</table>

After studying these results we conclude that a good prediction system is required for predicting the software defects at an early stage of software development life cycle. Using MLP technique in ANN, the value of AUC is better than others. MLP is based on a machine learning approach. So, it is found that machine learning models are mostly used and provides the better results. These methods can be used for cross-company projects also. The proposed model provides better accuracy and TNR. This model gives low values for MSE. The K-means algorithm is most simple and well-known algorithm which can be easily implemented.

References


**Notes**

1www.github.com
2www.bugzilla.org